Automating a manual process of scheduling jobs to productions machines in a pharmaceutical company

Guðrún Sjöfn Axelsdóttir

2010

MSc thesis in Decision Engineering

Author: Guðrún Sjöfn Axelsdóttir
Id no: 290875-3159
Supervisors: Dr Eyjólfur Ingi Ásgeirson and Dr. Hlynur Stefánsson

Tækni- og verkfræðideild
School of Science and Engineering.

Sjálfvirk leið fyrir handvirkt ferli í niðurröðun verka í fyrirfram skilgreinda framleiðsluáætlun hjá lyfjafyrirtæki

Útdráttur

Þessi rannsókn skoðar niðurröðun verka hjá lyfjafyrirtæki. Þrjár aðferðir eru bornar saman við að raða verkefnum niður; bestun, gráðugt algrími og núverandi handvirk leið sem lyfjafyrirtækið notar í dag og unnið var með raunveruleg gögn frá lyfjafyrirtækinu. Markmiðið með rannsókninni var að sjá hvort hægt væri að gera þetta ferli sjálfvirkt. Niðurstöður sýna að hægt er að notast við bestun og gráðugt algrími til að leysa þetta vandamál sjálfvirkt.

Automating the manual process of scheduling jobs in prepared plans for a pharmaceutical company

Abstract

The study looks at scheduling jobs for a pharmaceutical company into prepared plans. Comparison is made between three methods to assign the jobs to a prepared plan, optimization, a greedy algorithm and the current manual assignment in use today at the pharmaceutical company. The comparison was based on real data from the pharmaceutical company. The goal is to see if the process can be automated. Results show that both optimization and a greedy algorithm can be used to solve this problem automatically.

Key words: Optimization, scheduling, flexible Flowshop, pharmaceutical industry

The undersigned hereby certify that they recommend to the School of Science and Engineering at Reykjavík University for acceptance this thesis entitled “Automating the manual process of scheduling jobs in prepared plans for a pharmaceutical company” submitted by Guðrún Sjöfn in partial fulfillment of the requirements for the degree of Master of Science.

_____

Supervisor, Dr Eyjólfur Ingi Ásgeirsson

_____

Supervisor, Dr Hlynur Stefánsson

_____

Examiner, Dr Páll Jensson

List of Tables **page**

List of Figures **page**

1 Introduction

Scheduling is inherent to man in its basic form and is essential for organization of time. Without it chaos emerges. The goal of scheduling as a decision making process in the production environment is to optimize production by taking into account the allocation of resources and setting priority levels for different tasks. Single criteria scheduling has one main objective which may be the minimization of the number of tasks completed after their due date or minimization of time spaced between tasks worked on the same machine. The use of scheduling is broad-based, not only in manufacturing and service industry but also in rescue operations and executions of computer programs [8][1].

This article will look at scheduling in terms of optimizing production for a pharmaceutical company. The problem will be divided into basic elements as often defined by the theory. A description of the objective function is included when explaining the elements. Elements are added step by step to the problem and explained how the complexity increases. The problem is NP-hard, so unless P=NP there is no polynomial time algorithm that finds an optimal solution [4].

This study is inspired by a study by Stefansson et.al [10,11], where the same problem is studied. The study focused on solving the whole problem both creating a production plan and putting the jobs into the production plan. The problem that Stefansson et.al. had was stability issues, the production plans were ever changing and the solution time of the problem was also too long for the pharmaceutical company. The goal of this study is to find a realistic, quick and practical solution for putting jobs into a prepared plan that is practical and easy to use in a real world environment.

1.1.1 Short practical description

The study is directed to solve a real life problem originating from a pharmaceutical company. The pharmaceutical company must have a competitive edge to survive, which means that the company must cut down time to market and deliver a fast response to customers with high service level and reliability. Production cost must be kept at minimum and margins increased by maximizing the utility of assets at the same time. The pharmaceutical company has an order driven multistage production process in a single plant to make tablets. The production is divided in to four stages, granulation, compression, coating and packing. Scheduling is needed to optimize the production progress as for each stage there are several machines with different properties and capabilities that can produce the product. Figure 1 shows a typical decision process for a single product.

[1] In preface pages vii-viii

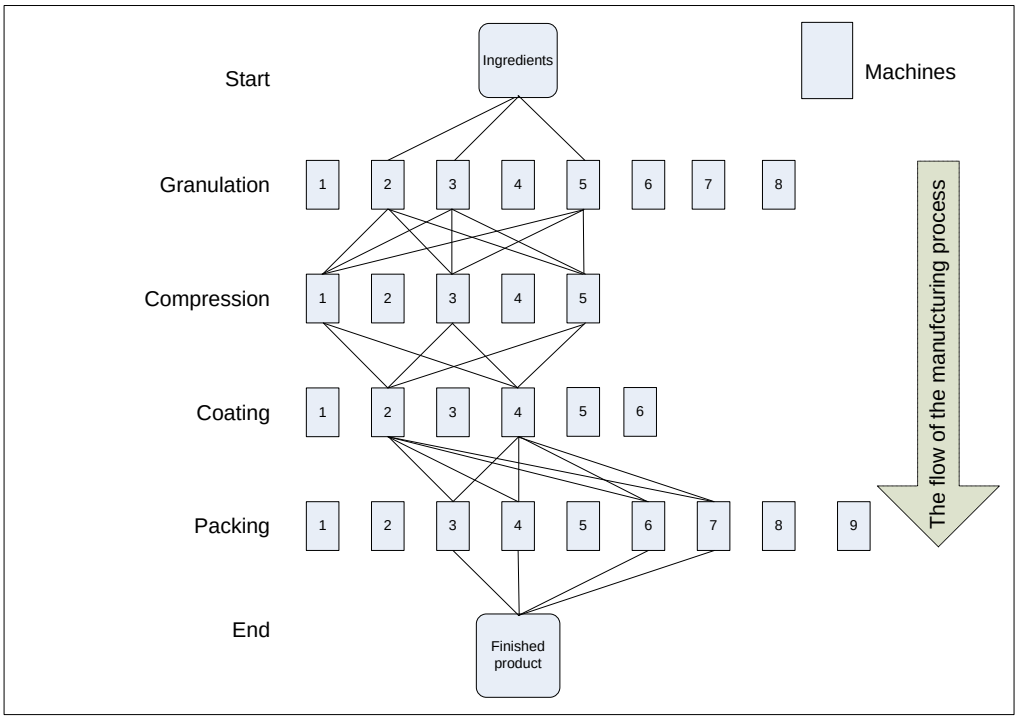

Figure 1: The complexity of the decision of choosing a path for a single product in the production process. For example if the production process can start from 8 machines on stage 1, 5 machines on stage 2, 6 machines on stage 3 and 9 machines on stage 4 then there are a total of 2160 different production paths possible for the product.

The goal of the pharmaceutical company is to respond efficiently to ever changing customer demand but at the same time maximize the plant's utilization. Relatively simple planning and scheduling is rendered useless to increased complexity of these processes combined. Scheduling will be used to decide when and how to complete products of jobs.

Superior utilization of resources, increased flexibility and reduced response time must be weighed against minimizing production cost and matching production to demand planning. The economic performance of the plant is optimized and best possible profit margin ensured. With scheduling the economic imperatives of the plan are translated in to the sequence of actions to be executed in the plant.

Problems are often entwined within these two processes and distinguishing between a planning and a scheduling problem for all practical purposes impossible. Therefore in real life companies do not try to solve this problem optimally, instead the problem is simplified.

The production plant setup is identical to the setup in work of Stefansson et.al. [10,11]. Production in the plant either has three or four stages as not all pills are coated. Every batch goes through the production process in the same order, granulation, compression, coating and packaging. If a product does not require coating the processing time in stage 3 is set to zero. Each step of the production is assigned to a different campaign so for a coated pill job is assigned to four campaigns. Even though the pharmaceutical company has several machines for mixing a single machine is assigned to each drug because of strict standards for pharmaceutical companies. The other three stages of the production has several machines that the pills can be worked on, even more than one machine at once. Compression, stage 2, is where the dough from stage one is compressed into pills. Coating, stage 3, is where some of the production's pills are coated. And finally stage 4, packaging, is where

the pills are put in pages. Every job has sequence dependent setup time, so the time that it takes to set up machine for product A is dependent on what product was on the machine before.

1.1.2 Problem elements and complexity

The most basic element of the problem under consideration in this study is a flow shop problem. Flow shop *(F)* is a scheduling problem with n jobs and m machines, each job has to be processed on each one of the m machines in the production line. All jobs have to follow the same route, i.e., they have to be processed first on machine 1, then on machine 2 and so on. After completion on one machine a job joins the queue at the next machine. Flow shop problems are difficult even for relatively small instances. Minimizing the total completion time in a two machine flow shop with average complete time, i.e., $F2||\sum C_j$ is strongly NP-hard. [5]

Flexible flow shop *(FF)* is more general than flow shop. Instead of m machines, there are c stages in series and at each stage there are a number of identical machines in parallel. Each job has to be processed first at stage 1, then at stage 2, and so on. The flexible flow shop problem is more general than pure flow shop which is strongly NP-hard, so flexible flow shop is at least strongly NP-hard.

The machines in each stage in our problem are not identical and are thus classified as unrelated machines in parallel *(R).* At every stage there are different machines in parallel and machine i can process job j at speed v_{ij}.

Sequence dependent setup time s_{jki} is dependent on the product that is scheduled on the machine and the previous product processed on the machine. That is means that the setup time that is incurred between the processing of jobs j and k on machine i is dependent on both jobs j and k and on machine i.

Due date (d_j) related objective functions are common in manufacturing companies. Due dates are the dates of delivery, promised to the customer. To delivery of the products to embed due dates in the object function tardiness $T_j = max(ET_j - d_j, 0)$ is used, ET_j is the time that the product is finished. Tardiness measures how many units of time the completion time are past the due date. In our case the units of time are days.

When unrelated machines in parallel $(Rm)$with sequence dependent setup times (s_{ijk}) and total weighted tardiness $(Rm|s_{ijk}|\sum \omega_i T_i)$[2] are combined the problem is strongly NP-hard [8]

The problem can be written as $(FFcRm|s_{ijk}|\sum \omega_i T_i)$ which represent the flexible flow shop with unrelated machines in parallel with sequence depended setup times and total tardiness. Since both FFc and $(Rm|s_{ijk}|\sum \omega_i T_i)$ are strongly NP-hard, the combination of those two problems is at least strongly NP-hard.

1.2 Known solution methods

When problems are NP-hard solutions are often found using meta-heuristics or if the problem instance is sufficiently small, mixed integer programming can be used. Meta is a Greek for “higher-

[2] *The problem that Pinedo describes in page 143 is $(Qm|s_{ijk}|\sum \omega_i T_i)$, where Q_m is for machines in parallel with different speeds, R_m is a generalization of Q_m so $FFcRm|s_{ijk}|\sum \omega_i T_i)$ is at least NP-hard.*

level" and heuristics solution methods are algorithms that are running time and/or solution quality is not guaranteed [2]. Meta-heuristics are heuristics that can be used to solve a very general class of computational problems and include methods such as simulated annealing, evolutionary algorithm and swarm intelligence. Meta-heuristics often "evolve" into more detailed and more specific methods when focused on a particular problem. Meta-heuristics is a way to find reasonably good solutions in a relatively short time. The downside is that in meta-heuristics it is impossible to determine if the solution is an optimal solution or how far from the optimum the solution is.

Flexible flow shop is often solved using LPT (or longest remaining processing time first (LRPT)) which can be optimal for a single stage but when the stages are multiple this is no longer true. A drawback with this method is the fact that the later stages can be idle for a long time [8].

Unrelated machines in parallel with sequence dependent setup times is a problem which is hard to solve optimally, even for relatively small problems. Heuristics such as neural networks or meta heuristics have been used to find solutions to this problem [1, 9].

In recent work on the problem of solving flexible flow shop with unrelated parallel machines and dual criteria by Jungwattanakit et. al.[6] they show that for constructive algorithms, job insertion based algorithms outperform other constructive algorithms and for iterative algorithms, genetic algorithms perform better than simulated annealing.

2 Solution approach

To optimize production in a pharmaceutical plant several factors must be taken into account. Each production stage is operated in batch mode with a number of multi-purpose production equipment at each stage. Due to large sequence set-up and cleaning times required to switch between batches containing products from different product families the plant produces products in large lots called campaigns. Switching between batches within the same product family significantly reduces set up and clean-up time, which in turn lowers production cost of the campaign. A campaign is an ordered set of batches containing products from the same product family produced consecutively by the same machine. With the predefined campaigns, the number of possible production routes and times is however greatly reduced. No changes were made to the pharmaceutical campaigns in this study. Instead the focus was on the scheduling of the jobs within the campaigns. Each product has a number of different feasible production routes through the plant and as the number of product families is over 40 and product variations are more than 1000, the process of planning and scheduling the production in an optimal way becomes extremely complicated.

Every job has to go to several campaigns, one campaign for each stage of the production and in the right order. Prior to this study the pharmaceutical company had manually inserted every job to the campaigns by several employees. In this study a comparison is made between the manual way, a mixed linear integer optimization model and a greedy algorithm. The focus of the study is to see if an automatic way of inserting jobs in preplanned campaigns based on forecasted demand will prove more efficient and /or at all relevant to the routine enforced by the pharmaceutical company today. Care must be taken to make sure that the jobs assigned to each campaign are in the right stage, of the right type and that the job starts after its predecessor has started.

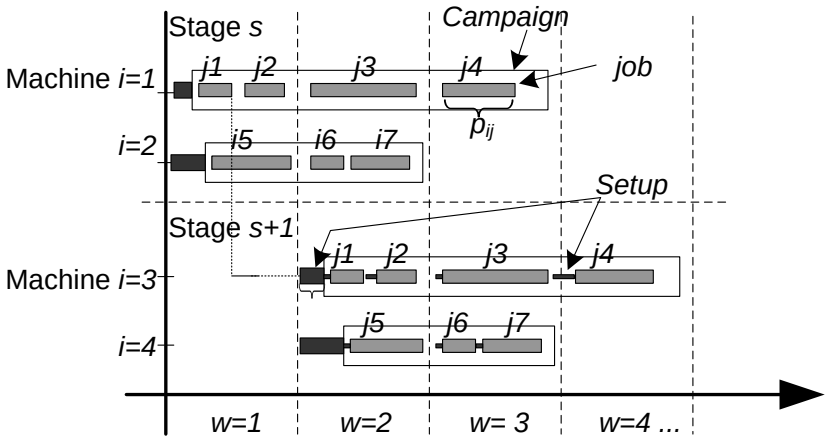

Figure 2: The figure illustrates the structure of campaigns and jobs at different production stages and also explains the use of various parameters in the models. Setup times between campaigns are included in the models proposed for level 1 and 2 and in addition, setup times between individual jobs within each campaign are included in the model proposed for level 3.

2.1 Current manual solution method

At the time of the research the pharmaceutical company collected a list containing one week of jobs and then scheduled the jobs into campaigns. The employees use a graphical interface which allows them to drag and drop the jobs into the campaigns. They are only allowed to put the jobs into the campaigns that are of the same type, with enough room and such that the previous stage of the job is in a campaign that starts before the selected one. The employees also make sure that the right ingredients are in stock before scheduling the job. The jobs cannot be processed if the right ingredients are not available. For simplification this part of the process was ignored.

2.2 Mixed integer job assignment model

A mixed integer model is proposed to place the jobs in the predefined campaigns. The objective of the mixed linear optimization model is to find an assignment of jobs into campaigns that minimizes the total tardiness of the jobs. Each campaign has one product line (type) at one stage of the manufacturing process that split the jobs (j) in parts, the same amount of parts that the stages are for each product. Every job (j) has been replicated by the number of stages that the job has, it depends on the type of production line how many stages the job has.

The objective function is to minimize the total tardiness ($\sum_{j=1}^{n} T_j$). Here the due date is the requested delivery date from the costumer. This however is not the due date that the pharmaceutical company works with; they work with confirmed delivery, which is generated after they have put the job into the schedule. Of course the pharmaceutical company wants to have these dates as close together as possible. For comparison, requested delivery was the chosen endpoint for all the three methods instead of confirmed delivery.

Indexes: $k = \ campaign\ number$
$k' = \ campaign\ number$
$j = job$
$j' = job$
$l = stage$

Sets:

$$K: the\ set\ of\ campaigns, k \in K$$
$$n: the\ number\ of\ jobs, j \in n$$
$$L: the\ number\ of\ stage, l \in L$$

Constants:

Campaigns:

$$C_k: avalable\ time\ in\ campaign\ k$$
$$ST_k: start\ time\ for\ Campigni\ k$$
$$ET_k: end\ time\ of\ campaign\ k$$
$$Ctype_k: production\ type\ of\ campaign\ k$$
$$Cl_k: stage\ that\ campaign\ k\ is$$

Jobs:

$$d_j: due\ date\ of\ job\ j$$
$$Jtype_j: production\ type\ of\ order\ j$$
$$P_{jk}: processing\ time\ of\ a\ type\ j\ job\ on\ campaign\ k$$
$$Jl_j: stage\ for\ order\ j$$

Variables:

$$x_{jk} = \begin{cases} 1\ if\ order\ j\ is\ produced\ in\ campaign\ k \\ 0\ otherwice \end{cases}$$
$$y_j = \begin{cases} 1\ if\ there\ is\ no\ available\ campaign\ for\ order\ j \\ 0\ otherwice \end{cases}$$
$$T_j: tardiness\ of\ \ job\ j$$
$$D_k: extra\ time\ for\ campaign\ k$$

The objective function is subject to finding campaigns for each job j and they have to be of the same family group,($x_{pk} = 0\ \forall c, p: S_{ps} \neq S_k\ og\ Ptype_p \neq Ctype_k$). The campaign has to have capacity to manufacture the product in the job line ($\sum_{p=1}^{n} P_p \cdot x_{pk} \leq C_k\ \forall k$). Care must be taken to ensure that the process of a job has started on stage l before the processing on stage $l+1$ ends ($ET_k \cdot x_{jk} \leq ST_k x_{j'k} + M \cdot y_{j'}\ \forall k, j: type_j = type_k\ j \prec j'$), see figure 3

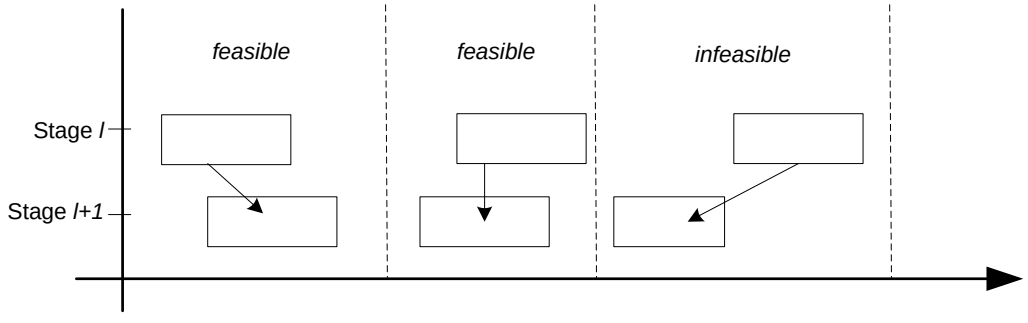

Figure 3: Illustration of a feasible and infeasible way for assigning jobs into campaigns on different stages of production.

.

Objective function:

The goal is to minimize the total sum of tardiness, $\sum_{j=1}^{m} T_j$. The assumption is that all jobs have unit weights. If there are jobs that do not find a campaign, a variable y_j is assigned for that. The variable y_j is 1 only if job j has no campaign and for this undesirable event a punishment constant α,

$\alpha \sum_{j=1}^{m} y_j$ is needed. In some cases the campaigns have extra capacity. To handle that, a variable D_k was added to the model, were D_k is the extra time campaign k is given. Here the punishment constant is β, $\beta \sum_{k=1}^{K} D_k$.

Constrains:

The time past the due date is found with the tardiness variable T_j

$$T_j \geq \sum_{k=1}^{K} ((ET_k - d_j) \cdot x_{kj}) \ \forall j,k: Jl_j = Cl_k = L \qquad (1)$$

Where ET_k is the end of campaign k and d_j is due date of the job j. Jl_j and Cl_k is the stage for job j and campaign k, were the concern with the due date is only at the final stage.

To make sure that the campaigns have enough room for the jobs that is assigned to the campaign:

$$\sum_{j=1}^{n} P_{jk} \cdot x_{jk} \leq C_k + D_k \forall k \qquad (2)$$

where is p_{jk} the processing time for job j in campaign k. C_k is the available space in campaign k, D_k is the extra time for campaign k.

To put jobs in the right campaigns

$$x_{jk} = 0 \ \forall k,j: Jl_j \neq Cl_k \ og \ Jtype_j \neq Ctype_k \qquad (3)$$

Here x_{jk} can only be equal to 1 when the job is at the same level and at the same production type as the campaigns.

Manufacture in the right job:

$$ST_{k'} \cdot x_{j'k'} \leq ET_k x_{jk} + M \cdot y_j \ \forall k,j: Jl_j = Cl_k \ , j' \prec j \qquad (4)$$

Where j' is the predecessor of job j, $ET_{k'}$ is the end time of campaign k' and ST_k is the start time of campaign k, M is a large number and y_j is 1 only if there is not a campaign for job j.

All the jobs j must either be assigned to a campaign or the job put in the list of unsigned jobs

$$\sum_{k=1}^{K} x_{jk} + y_j = 1 \ \forall j \qquad (5)$$

Where x_{jk} is 1 only if job j is manufactured in campaign k and 0 otherwise and y_j is 1 only if no campaign for job j is found. This constraint makes sure that job j is either assigned to a campaign or put on a list of unassigned jobs.

If job j has a predecessor j' and j' does not have campaign, the job j has no campaign:

$$y_{j'} \leq y_j \ \forall j : j' \prec j \qquad (6)$$

Here j' is a predecessor to job j and if the predecessor has $y_{j'}$ =1 than the job $j's$ y_j variable has to be 1 as well.

The upper bound for D_k

$$D_k \leq C_k \cdot \theta_k \qquad (7)$$

Where θ_k is the ratio which is allowed for extra time in campaign k

Figure 4 shows the mixed integer model.

Objective function:

$$min \sum_{j=1}^{m} T_j + \alpha \sum_{j=1}^{m} y_j + \beta \sum_{k=1}^{K} D_k$$

s.t.

$$T_j \geq \sum_{k=1}^{K} ((ET_k - d_j) \cdot x_{kj}) \ \forall j : Jtype_j = Ctype_k, Jl_j = Cl_k = L$$

$$\sum_{j=1}^{n} P_{jk} \cdot x_{jk} \leq C_k + D_k \forall k$$

$$x_{jk} = 0 \ \forall k, j : Jl_j \neq Cl_k \ og \ Jtype_j \neq Ctype_k$$

$$ST_{k'} \cdot x_{j'k'} \leq ET_k x_{jk} + M \cdot y_j \ \forall k, j : Jl_j = Cl_k \ , j' \prec j$$

$$\sum_{k=1}^{K} x_{jk} + y_j = 1 \ \forall \ j$$

$$y_{j'} \leq y_j \ \forall j : j' \prec j$$

$$D_k \leq C_k \cdot \theta_k$$

$$x_{jk} \in \{0,1\}^{n \cdot L \cdot K}$$

$$y_j \in \{0,1\}^{n \cdot L}$$

$$D_k \geq 0$$

$$T_j \geq 0$$

Figure 4: Optimization model, with the object function and the constraints of the model. The model finds th optimal way to put the jobs into campaigns while minimizing the tardiness of the jobs and put as many jobs as possible into campaigns.

The optimization was solved with the MPL Modeling System 4,2k and Gurobi 2.0.2. The number of jobs was around 100 each week and the campaigns where more than 300. Even though the number of jobs is low, the optimization had between 150-265 thousand constraints and 100-155 thousand variables.

2.3 Greedy algorithm

The greedy algorithm tries to find a campaign for jobs and is only concerned with the job at hand. Here the algorithm stops processing a job as soon as it finds a campaign to put the job in. That excludes optimization and only focuses on finding a solution. The way to influence the solution is by sorting the campaigns and the jobs. For example jobs can be sorted by requested delivery or by size. Then the greedy algorithm iterates through the list and the jobs are assigned to campaigns. This is similar to what is done in reality, except the employee can choose the priority for each job using size, due date or other variables. Figure 5 shows a pseudo code for the algorithm while Figure 6 shows a flowchart describing the algorithm.

Step 1

$J \leftarrow \{ordered\ set\ of\ jobs\}$
$K \leftarrow \{ordered\ set\ of\ campaigns\}$
$Cl \leftarrow \{Stage\ for\ campaign\ k\}$
$ST \leftarrow \{Start\ time\ for\ campaign\ k\}$
$ET \leftarrow \{End\ time\ for\ campaign\ k\}$
$p \leftarrow \{Processing\ time\ \ of\ jobs\ j\ in\ campaign\ k\ \}$
$L \leftarrow \{Number\ of\ stages\}$

Step 2, Find a campaign for job j

For each $j \epsilon J$

Set $l = 1$
Set $starttemp = 0$

While $l \leq L$

Set $ProcessingTime = \infty$

For each $k \in K$

If $(l = Cl_k)$ and $(Ptype_j = Ctype_k)$ and $(starttemp \leq ET_k)$

Set job j into campaign k
Set $starttemp = ST_k$
Set $l = l + 1$
Set $ProcessingTime = p_{jk}$

If $(ProcessingTime = \infty)$

Set job j into the list of unscheduled jobs
break

Figure 5: A pseudo code for the greedy algorithm, which takes an ordered set of jobs and place them into campaigns.

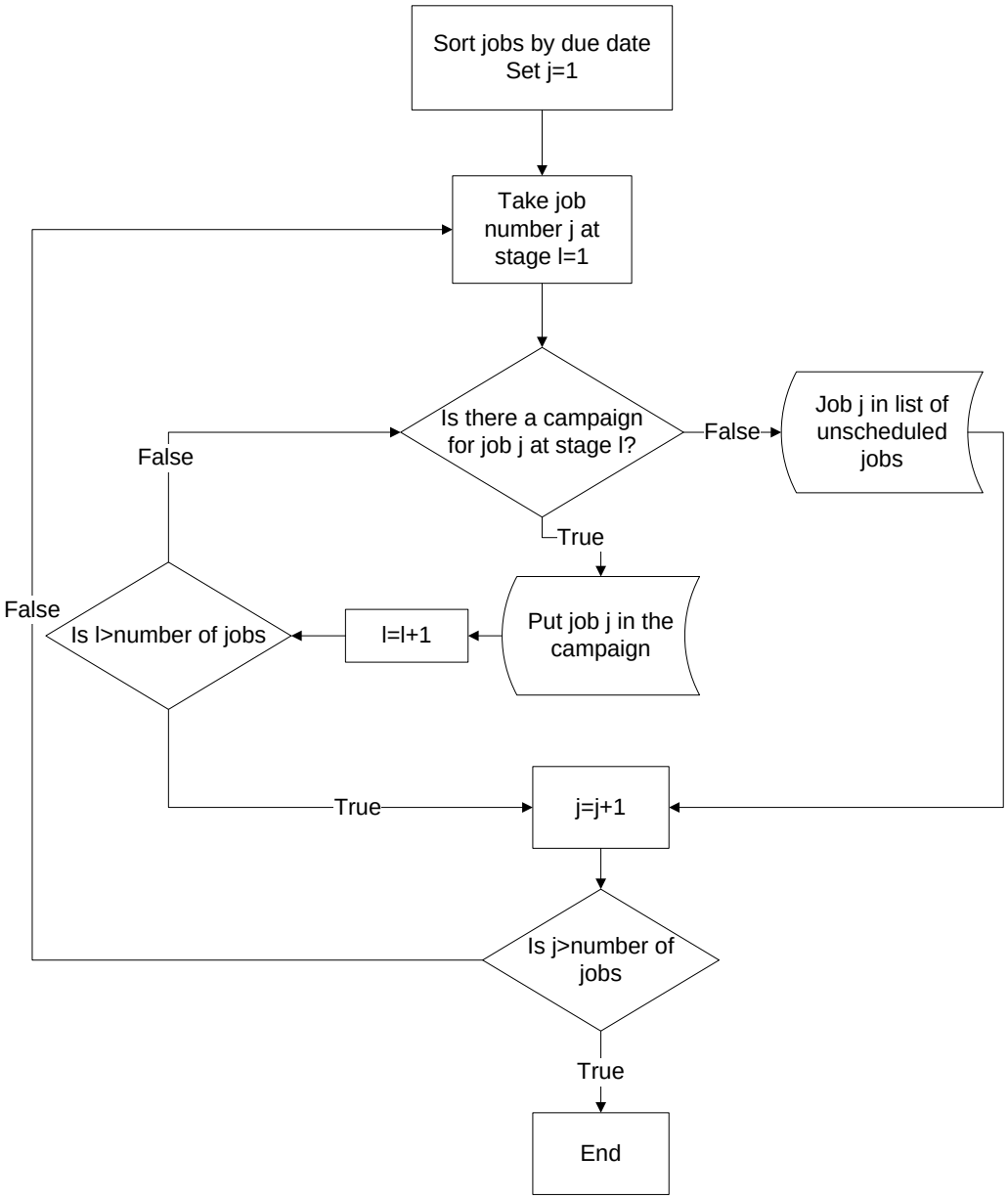

Figure 6: A flow chart of the greedy algorithm, the focus is to assign jobs to campaigns. As soon as a room for the job in a campaign is found the job is assigned to that campaign.

2.4 The test cases data

The data was collected from a manufacturing company in the pharmaceutical industry. The data was collected before and after the manual planning process was executed at the plant. The data included information on what jobs were unscheduled, due date for the jobs and information about the campaigns. Also information was gathered about how the scheduled jobs were assigned in the previous week. For simplification, the end of the campaign was defined as the end of the job and details such as work hours of the plant and how many hours per day the machines were running were excluded. Data was collected for three weeks of scheduling in the company. As mentioned before the number of jobs was around 100 each week and the campaigns where over 300.

The current manual method used by the pharmaceutical company to assign the jobs to campaigns was used as a benchmark for the other methods.

To evaluate the quality of the three different solutions, measurable parameters were used. Tardiness T_j. The main goal was to minimize sum of tardiness. The end time of a job was defined as the end time of the campaign that the job was processed in. Lateness L_j, is how many days job j finished before or after its due date, $L_j = ET_j - d_j$.

The unit penalty U_j of job j is $U_j = \begin{cases} 1 \; if \; ET_j > d_j \\ 0 \; otherwise \end{cases}$. Unit penalty was used to count how many jobs are tardy. The jobs that finish before their due date have indicator $V_j = \begin{cases} 1 \text{ if } \mathrm{ET_j} - \mathrm{d_j} \leq 0 \\ 0 \text{ oterwise} \end{cases}$, V_j was used to count the jobs that are ready on-time. The end time (ET_j) is only defined for j if job j finds a campaign at the final stage. Because jobs have to be processed in campaigns, not all jobs find a campaign. The set of jobs that have no campaigns, which is unscheduled, is Z.

Table 1 shows how the actual schedule for the three weeks. These results of the actual schedule are a benchmark for the other methods.

Week	$\sum_{j=1}^{n} T_j$	$\frac{\sum_{j=1}^{n} L_j}{n}$	$\frac{\sum_{j=1}^{n} T_j}{\sum_{j=1}^{n} U_j}$	$\max(T_j)$	$\sum_{j=1}^{n} U_j$	$\sum_{j=1}^{n} V_j$	$\lvert Z \rvert$	$\frac{\sum_{j=1}^{n}(U_j + V_j)}{n}$	$\frac{\sum_{j=1}^{n} V_j}{n}$
Week 1	437	-10. 9	25. 4	59	19	37	29	66%	44%
Week 2	1471	24. 4	46. 6	158	15	33	50	49%	34%
Week 3	426	-14. 9	26. 6	55	16	51	67	50%	38%

Table 1: Actual solution from the industrial partner. The first column shows the overall tardiness. Then second column shows the average of lateness. The third column shows average tardiness of the jobs and column 4 is the largest tardiness. Columns 5 and 6 contain the number of jobs that are ready after and before there due date. Column 7 shows the number of jobs that did not find a campaign for all of their stages. Column 8 and 9 contain the percentage of jobs that found campaigns and the number of jobs that are ready before their due date.

Week one is poor for comparison since the pharmaceutical company added campaigns to their solution to meet increasing demand and those campaigns were not part of the study data.

3 Results

Tables 2-3 contain the results of the optimization and the greedy methods where the campaign capacity was set equal to the plant maximum capacity.

Week	$\sum_{j=1}^{n} T_j$	$\frac{\sum_{j=1}^{n} L_j}{n}$	$\frac{\sum_{j=1}^{n} T_j}{\sum_{j=1}^{n} U_j}$	$\max(T_j)$	$\sum_{j=1}^{n} U_j$	$\sum_{j=1}^{n} V_j$	$\lvert Z \rvert$	$\frac{\sum_{j=1}^{n}(U_j + V_j)}{n}$	$\frac{\sum_{j=1}^{n} V_j}{n}$
Week 1	357	-22,75	22,31	85	16	35	34	60%	41%
Week 2	1371	0,19	76,17	245	18	36	44	55%	37%
Week 3	459	-28,38	28,69	69	16	60	58	57%	45%

Table 2: The optimization with factory at maximum capacity. The first column shows the overall tardiness. Then second column shows the average of lateness. The third column shows average tardiness of the jobs and column 4 is the largest tardiness. Columns 5 and 6 contain the number of jobs that are ready after and before there due date. Column 7 shows the number of jobs that did not find a campaign for all of their stages. Column 8 and 9 contain the percentage of jobs that found campaigns and the number of jobs that are ready before their due date.

Week	$\sum_{j=1}^{n} T_j$	$\frac{\sum_{j=1}^{n} L_j}{n}$	$\frac{\sum_{j=1}^{n} T_j}{\sum_{j=1}^{n} U_j}$	$\max(T_j)$	$\sum_{j=1}^{n} U_j$	$\sum_{j=1}^{n} V_j$	$\lvert Z \rvert$	$\frac{\sum_{j=1}^{n}(U_j + V_j)}{n}$	$\frac{\sum_{j=1}^{n} V_j}{n}$
Week 1	384	-34,69	25,60	85	15	34	36	58%	40%
Week 2	1557	12,68	81,95	245	19	25	54	45%	26%
Week 3	366	-42,29	28,15	55	13	46	75	44%	34%

Table 3: The results for the greedy algorithm with factory at maximum capacity. The first column shows the overall tardiness. Then second column shows the average of lateness. The third column shows average tardiness of the jobs and column 4 is the largest tardiness. Columns 5 and 6 contain the number of jobs that are ready after and before there due date. Column 7 shows the number of jobs that did not find a campaign for all of their stages. Column 8 and 9 contain the percentage of jobs that found campaigns and the number of jobs that are ready before their due date.

In Tables 4 and 5 the algorithms are run again but this time with 80% of the plant´s maximum capacity. In reality problems will arise and the plant cannot maintain maximum capacity at all times. By using only 80% of maximum capacity the plant has the option to push through production when needed.

Week	$\sum_{j=1}^{n} T_j$	$\frac{\sum_{j=1}^{n} L_j}{n}$	$\frac{\sum_{j=1}^{n} T_j}{\sum_{j=1}^{n} U_j}$	$\max(T_j)$	$\sum_{j=1}^{n} U_j$	$\sum_{j=1}^{n} V_j$	$\lvert Z \rvert$	$\frac{\sum_{j=1}^{n}(U_j + V_j)}{n}$	$\frac{\sum_{j=1}^{n} V_j}{n}$
Week 1	471	-15,2	26,2	85	18	31	36	58%	36%
Week 2	1272	-5,9	74,8	245	17	35	46	53%	36%
Week 3	561	-22,6	33,0	69	17	55	62	54%	41%

Table 4: The results for the optimization with the factory at 80% of maximum capacity. The first column shows the overall tardiness. Then second column shows the average of lateness. The third column shows average tardiness of the jobs and column 4 is the largest tardiness. Columns 5 and 6 contain the number of jobs that are ready after and before there due date. Column 7 shows the number of jobs that did not find a campaign for all of their stages. Column 8 and 9 contain the percentage of jobs that found campaigns and the number of jobs that are ready before their due date.

Week	$\sum_{j=1}^{n} T_j$	$\frac{\sum_{j=1}^{n} L_j}{n}$	$\frac{\sum_{j=1}^{n} T_j}{\sum_{j=1}^{n} U_j}$	$\max(T_j)$	$\sum_{j=1}^{n} U_j$	$\sum_{j=1}^{n} V_j$	$\lvert Z \rvert$	$\frac{\sum_{j=1}^{n}(U_j + V_j)}{n}$	$\frac{\sum_{j=1}^{n} V_j}{n}$
Week 1	529	-29,3	31,1	85	17	29	39	54%	34%
Week 2	1378	10,0	158,0	245	17	24	57	42%	24%
Week 3	435	-34,4	31,1	69	14	42	78	42%	31%

Table 5: The results for the greedy algorithm with the factory at 80% of maximum capacity. The first column shows the overall tardiness. Then second column shows the average of lateness. The third column shows average tardiness of the jobs and column 4 is the largest tardiness. Columns 5 and 6 contain the number of jobs that are ready after and before there due date. Column 7 shows the number of jobs that did not find a campaign for all of their stages. Column 8 and 9 contain the percentage of jobs that found campaigns and the number of jobs that are ready before their due date.

Figure 7 and 8 show how the campaigns and jobs are scheduled in those campaigns for the two methods. To simplify, the start time of the jobs was defined as the start time of the campaigns. The widest boxes are campaigns and the black smaller ones are the jobs which are put into campaigns.

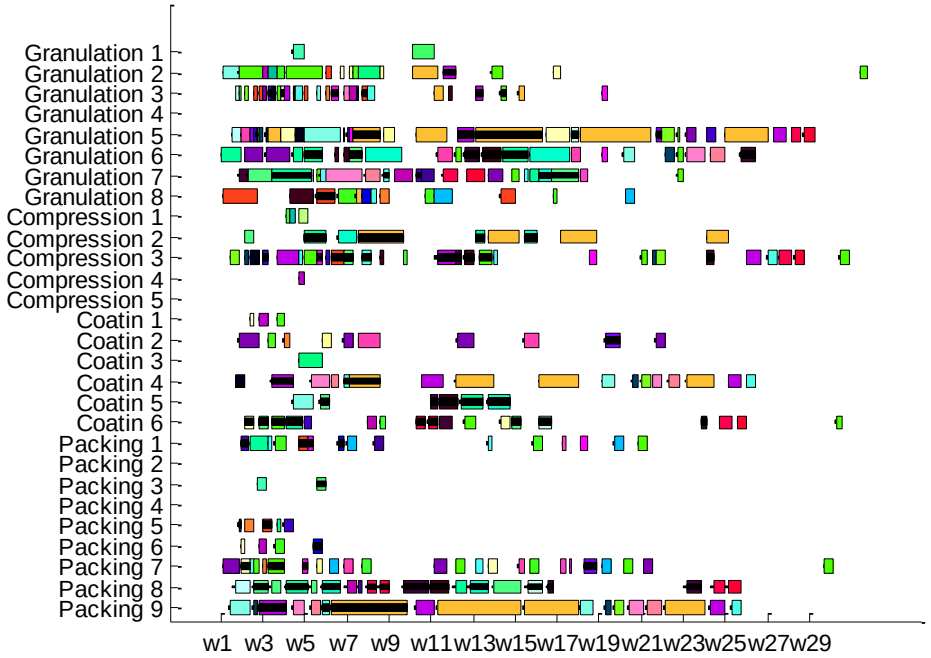

Figure 7: A Gantt chart of the solution for week 1 for the optimization with 80% capacity; showing the campaigns and the jobs. The campaigns that have no job assignment from this week are in light colors, but the campaigns into which jobs are assigned have a black stripe in the middle of the campaign.

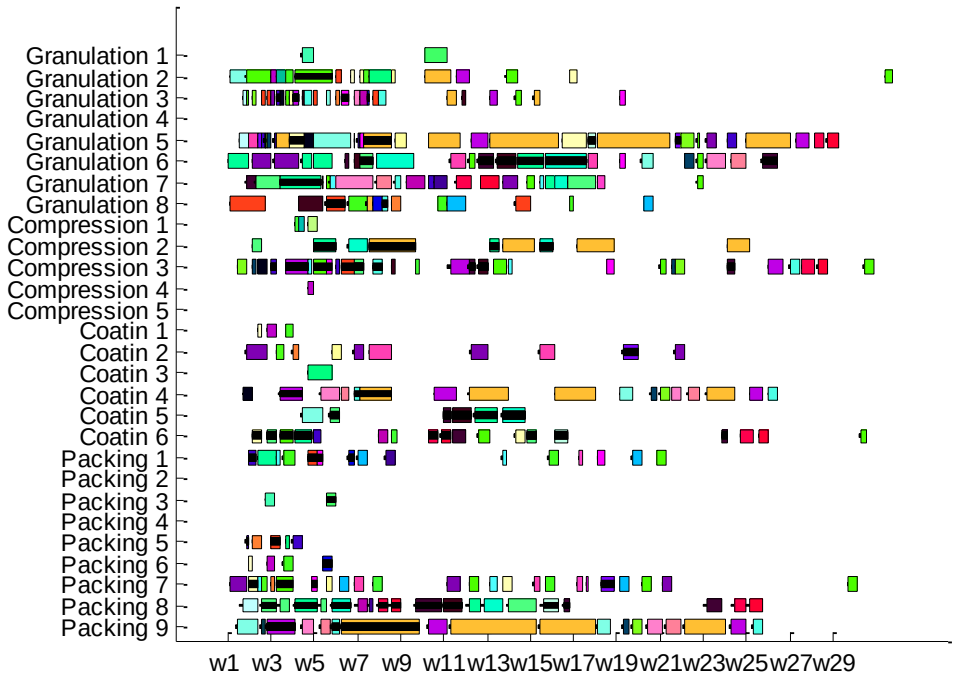

Figure 8: A Gantt chart of the solution for week 1 for the greedy algorithm with 80% capacity; showing the campaigns and the jobs. The campaigns that have no job assignment from this week are in light colors, but the campaigns into which jobs are assigned have a black stripe in the middle of the campaign.

To see if there is any difference between the three methods the average overall tardiness and standard deviation of the jobs that found a campaign at the final stage was calculated.

Comparison of tardiness	$\bar{T} = \frac{\sum_{j=1}^{n} T_j}{n}$	$s_T = \frac{\sum_{j=1}^{n} (T_j - \bar{T})^2}{n-1}$	n
Optimization	9.61	18.45	49
Greedy	11.50	20.23	46
Actual	8.45	15.25	56

Table 6: The average and standard deviation of Tardiness of jobs that found a campaign for all the three method at week 1, where the optimization and greedy algorithm were run with 80% capacity.

Comparison of percent of job finished	$\bar{R} = \frac{\sum_{j=1}^{n} (U_j + V_j)}{n}$	$s_T = \frac{\sum_{j=1}^{n} ((U_j + V_j) - \bar{R})^2}{n-1}$	n
Optimization	0.57	0.50	84
Greedy	0.55	0.50	84
Actual	0.67	0.47	84

Table 7: The average and standard deviation over precentage of job that found a campaign for all the three method at week 1, where the optimization and greedy algorithm were run with 80% capacity.

3.1.1 Solution time for each method

The greedy method solves the problem in only a fraction of a second. The optimization using MPL solved the problem in less than one minute. Today the process is done manually by four employees for each campaign with three or four campaigns needed to make an end product. It takes each employee more than half an hour per week to allocate campaigns for jobs.

3.2 Comparison

Figure 9 compares results for the three methods with the plant running at maximum capacity. The columns show the percentage of jobs that each method manages to finish. In week 1, we see that the actual solution does the best, even better than the optimization. This is because the workers at the pharmaceutical company made new campaigns as required for the jobs that the company got. Therefore the actual solution had more resources than the optimization.

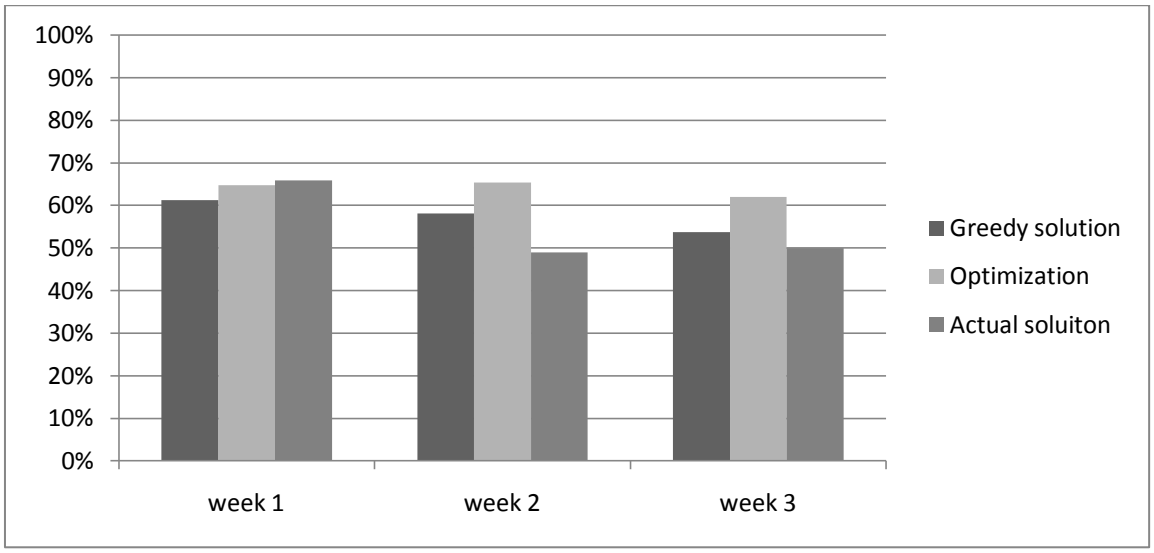

Figure 9: The difference in the three methods of the percentage of the jobs that are on schedule. This is when the methods are run at the factory´s maximum capability (from table 2 and 3). The darkest grey is for the greedy algorithm, the lightest

is for the optimization and the third color represent the manual process. In week 1 the employees of the plant added more campaigns to the schedule so the comparison in week 1 is not fair.

Since plants can never run at maximum capacity all the time we compare the result to 80% of maximum capacity, figure 10.

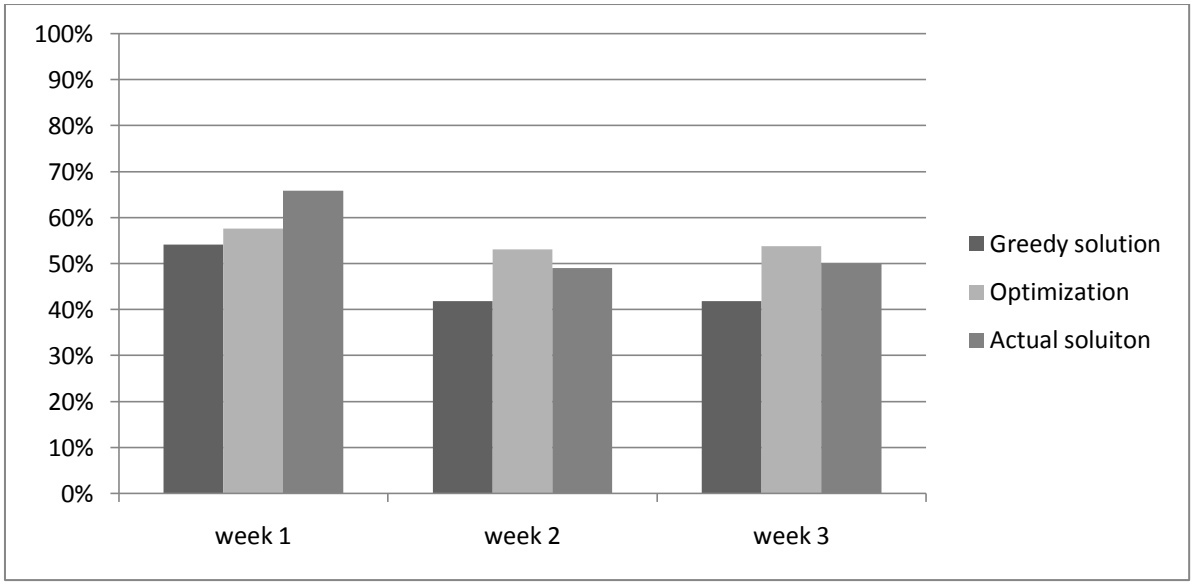

Figure 10: The difference in the three methods of the percentage of the jobs that are on schedule. This is when the methods are run at the factory´s maximum capability (from table 2 and 3). The darkest grey is for the greedy algorithm, the lightest is for the optimization and the third color represent the manual process. In week 1 the employees of the plant added more campaigns to the schedule so the comparison in week 1 is not fair.

In comparison there is an insignificant difference between 80% and 100% of maximum capacity. The reason for this is that the numbers of jobs applicable to campaigns were still relatively large. This in turn explains why optimization has similar results as the other two solutions. Furthermore, the campaigns strongly restrain the optimization since the optimization can only assign jobs to predetermined campaigns, rather than determining the schedule from scratch. The jobs are scheduled once a week and not changed after that[3], which makes the problem semi-online whereas once the jobs have been scheduled for the week, the schedule is not revised even with further information on demand. The greedy solution works well in this setting and is easy to program.

The working hypothesis is that there is no difference between methods in number of jobs completed. The hypothesis is rejected when 1-(confidence-level) is higher than the p-value. To see if there is a statistical difference between methods in week 1, the numbers of completed jobs were compared. In table 8, with 95% confidence level, the difference is insignificant and therefore no reason to reject the hypothesis. But at 90% confidence level the hypothesis is rejected and the greedy and actual way

[3] *The pharmaceutical company can change the job ordering, customers change and cancel their orders. In the current manual scheduling process when a big important job comes, the company tries to make room for that job, even removing other jobs out of the campaign.*

are not the same. Where the standard deviation is not the same the t test uses $s_{R_1-R_2} = \sqrt{\frac{s_1^2}{n_1}+\frac{s_2^2}{n_2}}$ and degree of freedom is $df = \frac{\left(s_1^2/n_1+s_2^2/n_2\right)^2}{\left(s_1^2/n_1\right)^2/(n_1)+\left(s_2^2/n_2\right)^2/(n_2)}$ [7].

Is there a difference between the methods?	$\lvert R_1 - R_2\rvert$	$s_{R_1-R_2}$	$t = \frac{\lvert R_1 - R_2\rvert}{s_{R_1} s_{R_2}}$	df	$P - value$
Greedy vs. Opt	2.38%	7.70%	0.31	165.99	0.38
Opt vs. Act	9.52%	7,50%	1.27	165.73	0.10
Greedy vs. Act	11.90%	7,52%	1.58	165.65	0.06

Table 8: Comparison of the values from table 7. Our hypothesis was that there is no difference in percentage of jobs finished in week 1 at 80% maximum capacity and there is no reason to reject the hypothesis at 95% confidence level. Act stands for the way this was done at the pharmaceutical company and Opt is for optimization

In week 2 and 3 the hypotheses that the greedy and optimization find campaigns for same amount of jobs is rejected. In week 3 the optimization is also better than the actual way. In both weeks there is no reason to reject the hypothesis that the greedy solution and actual way had the same percentage of job in campaigns. It is also interesting at 95% confidence level, the result do not give a reason to reject the hypothesis that the greedy solution and actual solution are the same, for all the three weeks. The optimization has additional benefits because of the shadow price. Shadow price gives for example the information of the possible gain from adding more time to the campaigns in linear optimization. In this case we are using integer linear optimization, meaning the shadow prices do not give the same accurate information as in linear optimization [2].

	Week 2			Week 3		
Is there a difference between the methods?	$\lvert R_1 - R_2\rvert$	$s_{R_1-R_2}$	$P - value$	$\lvert R_1 - R_2\rvert$	$s_{R_1-R_2}$	$P - value$
Greedy vs. Opt	11.22%	7.12%	0.06	16.33%	6.04%	0.00
Opt vs. Act	4.08%	7.17%	0.28	15.31%	6.04%	0.01
Greedy vs. Act	7.14%	7.13%	0.16	1.02%	6.12%	0.43

Table 9: Comparison of percentage of jobs that finished in week 2 and 3, our hypothesis was that there is no difference in percentage of jobs finished in those weeks at 80% maximum capacity. Act stands for the way this was done at the pharmaceutical company and Opt is for optimization

Let us take a closer look at week 1 at 80% of maximum capacity. Is there a statically difference between these three methods of finding campaigns for the jobs? Table 9 shows calculated p-values of comparison student t test. The hypothesis is that there is no statistical difference between those three methods on the overall tardiness of the jobs that find campaigns. There is no reason to reject the hypothesis even with confidence level as low as 80%.

Is there a difference between the methods?	$\lvert\overline{T_1} - \overline{T_2}\rvert$	$s_{T_1}s_{T_2}$	$t = \frac{\lvert\overline{T_1} - \overline{T_2}\rvert}{s_{T_1}s_{T_2}}$	df	$P - value$
Greedy vs. Opt	1.89	3.98	0.471	90.80	0.31
Opt vs. Act	1.88	3.68	0.511	101.24	0.31
Greedy vs. Act	0.01	3.93	0.002	93.52	0.50

Table 10: Comparison of , our hypothesis was that there is no difference in average tardiness in week 1 at 80% of maximum capacity and there is no reason to reject the hypothesis. Act stands for the way this was done at the pharmaceutical company and Opt is for optimization

For the average tardiness in the other two weeks there was no reason to reject the hypothesis that the average tardiness for the jobs is the same in all the comparisons of the methods, with confidence level 80%. The reason for that is that the standard deviation is very high, twice the size of the difference in tardiness.

4 Conclusions and future work

The pharmaceutical company schedules its jobs into prepared manufacturing plans. In our study we focused on the way this is done today and how to make this process more automatic. For simplification we focused mostly on tardiness. We tested a greedy algorithm and optimization against the current manual way of the pharmaceutical company.

We found that the greedy algorithm was just as good as the company manual solution. The greedy solution is fairly easy to implement and the company could benefit from having this problem solved automatically or interactively between the employees and the automatic methods.

The optimization was statistically the best method to solve this problem and has the advantage of information on shadow price, but the shadow price is not perfect for integer linear optimization. However the implementation of the optimization is not as easy as the greedy solution which is the only drawback of this method.

Future work includes adding weights to the jobs. The weights would represent the size and/or the importance of the jobs. In the optimization the weight is put in the objective function so the modification which is trivial. In the greedy algorithm it takes further study to find the best way to sort the list after weights and due date. There should also be a check for availability of ingredients before scheduling the jobs, as the employees do in the pharmaceutical company. Further improvements could be made to include addition of campaigns or changes and to look more closely at the forecasting process used.

5 Acknowledgments

The author wishes to thank Sigrún B. Gunnhildardóttir, product manager of the pharmaceutical company, for helpful discussions on how the actual process works. Also the author would like to acknowledge financial support from the RU development fund.

6 References

[1]Aytung, H., Bhattacharyya, S., Koehler, G., & Snowdon, J. (1994). A Review of Machine Learning in Scheduling. *IEEE Transactions on Engineering, Vol. 6* , 165-171.

[2]Bertsimas, D., & Tsitsiklis, J. N. (1997). *Introduction to LINEAR OPTIMIZATION.* Massachusetts: Athena Scientific.

[3]Cormen, T. H., Leiserson, C. E., Rivest, R. L., & Stein, C. (2001). *Introduction to Algorithms, second Edition.* Boston: McGraw-Hill.

[4]Garey, M. R., & Johnson, D. S. (1979). *Computers and Intractability: A Guide to Theory of NP-Completeness.* San Fracisco: Freeman.

[5]Graham, R. L., Lawler, E., Lenstra, J., & Rinnooy Kan, A. (1979). Optimization and Approximation in Deterministic Sequencing and Scheduling: a Survey . *Annals of Discrete Mathematics* , 287-326.

[6]Jungwattanakit, J., Reodecha, M., Chaovalitwongse, P., & Werner, F. (2005). An Evaluation of Sequencing Heuristics for Flexible Flowshop Scheduling Problems with Unrelated parallel Machines and Dual Criteria. *Technical report, Faculty of Mathematice, Otto-vonGuericke-University, D-39016, Germany* , 1-23.

[7]Moore, D. S., & McCabe, G. P. (2006). *Inntroduction to the practice of statistics.* New Yourk: W.H. Freeman and Company.

[8]Pinedo, M. L. (2008). *Scheduling Theory, Algorithms, and Systems.* (T. Edition, Ed.) New York: Springer Science+Business media, LLC.

[9]Priore, P., de la Fuente, D., Gomez, A., & Puente, J. (2001). A Review of Machine Learning in Dynamic Scheduling of Flexible Manufacturin Systems. *Artificial Intelligence for Engineering Design, Analysis and Manufacturing* , 251-263, Vol. 15.

[10]Stefansson, H., Jensson, P., & Shah, N. (2009). A heuristical procedure for redducing the risk of delayed deliveries in make to order production. *the Journal of production Planning and Contorl* , Vol. 20, No. 4, 332-342

[11]Stefansson, H., Jensson, P., & Shah, N. (2006). An Efficient Procedure to Increase Robustness of Production Plans Embedded in an Integrate Multi-scle planning and Scheduling Approach. *Annual meeting of the American Institute of Chemical Engineer* , AIChE.


HÁSKÓLINN Í REYKJAVÍK
REYKJAVIK UNIVERSITY