Bachelor Thesis

Title: A combinatorial planning and scheduling problem: model and heuristic solution

Gregor Fischer
1403086

under the supervision of
Senior Lecturer Dipl.-Ing. Dr.techn Josef L. Haunschmied

Institute of Statistics and Mathematical Methods in Economics
TU Wien

18.09.2017

1 Introduction

For my bachelor thesis I worked 14 weeks for MCP, a company that sells and develops software that solves planning and scheduling problems for production enterprises. My task there was to develop a solution to calculate an optimized production plan for a potential customer. My bachelor thesis is about my work on the solution for the company. The solution of the problem is a sequence of giant paper rolls, so called motherrolls, with each motherroll having products assigned to them. In order to use an exact algorithm or a heuristic I developed a binary linear programming model, that uses binary variables to encode wether a certain product is on a certain motherroll. The initial model turned out to have to many constraints even for a heuristic algorithm to calculate a solution, but it showed the problems with modeling the situation of the customer. Namely encoding the different properties of products on each motherroll. This lead to a different approach, which is very common in operations research when there are too many combinations in a combinatorial optimization problem. The second model used given patterns of products and is similar to a cutting stock problem: minimize total quantity of motherrolls with given patterns, constraints encoding demand fulfillment. Patterns guarantee that products on the same motherroll have the same properties. Yet the second model doesn't deal with all problems. First of all it is not at all clear how to create patterns, since there are a lot of things to consider with combining products on patterns i.e. due dates, demand and properties of products. Second, the pattern model ignored the change of product properties between motherrolls, which is an important part of the problem: changing product properties on the machine leads to costs and setup time. Minimizing setup time was not the main goal of the problem, but there should not be too much setup time. Adding constraints to encode the change of product properties would have led to as many constraints as in the first model. In the solution I implemented for MCP I split the problem into three phases, where in each phase I dealt with different decisions of the production process. I wanted to use the model with the patterns, therefore I dealt with the setup times in the first phase by grouping products together with respect to product properties and due dates. The first phase basically creates jobs, where a job is producing a group of products, that have the same properties, in certain amounts. How to fulfill this jobs is done in the third phase with the pattern model. In phase two the sequence of this created jobs is optimized with respect to the setup time between each jobs. The model of phase two is similar to a job shop scheduling model (Knopp, Dauzere-Peres, Yugma 2017): Binary variables encode if a group is done before or after another group, constraints make sure the structure given by the solution of phase one stays the same. The second and third phase work independent from one another. Phase 3 could also be Phase 2 and the other way around. Each phase was solved with a heuristic algorithm. Although there are lots of things that can be improved, the total algorithm performed well, with given test data it was able to calculate a solution with little waste in five to ten seconds.

In the second section I will describe the problem, namely the different decisions and goals the customer has in his production process. In the third chapter I will provide the exact mathematical model. In the fourth chapter the problem of the exact model is stated and the model with patterns is formulated. In the last three chapters I will describe the solution, that I actually developed for MCP, namely the three phase approach where in each phase different decisions for the production plan are made. Each of the three chapters 5 to 8 will describe a phase containing the mathematical model behind the phase and a description of the heuristic that was used. In chapter nine I will describe the performance of my solution, having tested it with test data, and draw a conclusion from my work at MCP.

2 Problem description

The company produces the packaging for boxed drinks, which it sells to all kinds of beverage producing companies. In the production process they imprint paper rolls, so called motherrolls, where on each motherroll there are tracks of products imprinted next to each other. Later in the production process each motherroll is cut into these tracks. These tracks are then basically rolls of drink packs that can run through a filling system of a beverage company. Each product can have different variations in the following attributes:

- papertype
- size
- opening type
- dot area
- trapping rate
- colors

Products that are combined on the same motherroll must not differ in any of these attributes, they can also share utmost 7 different colors. Whenever one of these properties changes from one motherroll to the next , there is a setup time on the machine imprinting the motherrolls. The more properties change the more setup time is needed. Every motherroll has the same dimensions. Both the number of tracks per motherroll and the number of packs per track depend only on the size of the product. The goal of the production is to find the best combinations of products on motherrolls in order to have minimal waste, and have little setup time. Two different products can't be scheduled on the same track of a motherroll. The orders from the customers are given in terms of packs (not tracks) and they usually come 21 days before the delivery date. The company has the option to deliver the order between 7 days before and 7 days after the delivery date. Similar idea with the amount, the company can deliver between 90 % and 110 % of the ordered amount.

3 An exact model

The idea of this model is to consider the planning horizon as a sequence of motherrolls. In the following case I suppose the planning horizon is 21 days since the company receives each order 21 days before the delivery date, so it makes sense that they want to plan 21 days ahead in order to be able to tell if they want to accept the order. The company can produce about 50 motherrolls per day, so we have a sequence of $21 * 50 = 1050$ motherrolls. There is a one to one correspondence between a subsequence of motherrolls and a day. The first 50 motherrolls belong to day one, motherroll 51 - 100 belong to the second day and so on. The problem with this idea is that the number of motherrolls that can be produced on a day depends on how much setup time there is on that day. Of course one could include this fact in the model and have the number of motherrolls per day depend on the setup time of that day, but then one would loose that one to one correspondence between motherrolls and days. I solved this problem by only allowing a distinct amount of setup time per day in order to be certain that there can be at least 50 motherrolls produced per day.

3.1 Parameter

x_j	ordered amount of product j
$I = \{1, ..., 1050\}$	set of motherrolls in the plannig horizon
J	set of products in the planning horizon
$K = \{1, ..., 7\}$	set of tracks on motherroll
$I_j \qquad (\subset I)$	set of motherrolls that can be used to produce product j
S	set of all sizes
D	set of all dot areas
T	set of all trapping rates
V	set of all opening types
P	set of all paper types
C	set of all colors
$C(j)$	set of all colors used for product j
$N_{col}(j)$	number of colors used for product j
$v(j)$	opening type of product j
$s(j)$	size of product j
$d(j)$	dot area of product j
$t(j)$	trapping rate of product j
$p(j)$	paper of product j
a_s	number of packs on a track (depending on size)
$k_1, ..., k_6$	costs for changes of the machine configuration
b_s	number of tracks per motherroll (depending on size)
k_M	costs for one motherroll
k_R	costs for loss of revenue

3.2 Variables

$M_{i,j,k}$	binary variable (b.v) that encodes if product j is on track k of motherroll i

3.3 Alternatives variables

X_j	amount we produce of product j
$\hat{M}_i$	b.v that encodes if motherroll i is used
$\alpha_{i,c}$	b.v that encodes if color c is used on motherroll i
$\beta_{i,v}$	b.v that encodes if opening type v is used on motherroll i
$\gamma_{i,s}$	b.v that encodes if size s is used on motherroll i
$\delta_{i,d}$	b.v that encodes if dot area d is used on motherroll i
$\epsilon_{i,t}$	b.v that encodes if trapping rate t is used on motherroll i
$\zeta_{i,p}$	b.v that encodes if paper type p is used on motherroll i
$E_{0,i}$	number of colors that have changed on motherroll i
$E_{1,i}$	b.v that encodes if any color has changed on motherrroll i
$E_{2,i}$	b.v that encodes if a change of the opening has occured on motherroll i
$E_{3,i}$	b.v that encodes whether a change in size occurs on motherroll i
$E_{4,i}$	b.v that encodes if a change of the dot area has occurred on motherroll i
$E_{5,i}$	b.v that encodes if a change of the trapping rate has occured on motherroll i
$E_{6,i}$	b.v that encodes if a change of the paper has occurred on motherroll i
$e_{1,i,c}$	b.v that encodes if color c has changed on motherroll i
$e_{2,i,v}$	b.v that encodes if opening type v has changed on motherroll i
$e_{3,i,s}$	b.v that encodes if size s has changed on motherroll i
$e_{4,i,d}$	b.v that encodes if dot area d has changed on motherroll i
$e_{5,i,t}$	b.v that encodes if trapping rate t has changed on motherroll i
$e_{6,i,p}$	b.v that encodes if paper p has changed on motherroll i

3.4 Objective function

The target function models:

- costs for the motherrolls used in the production
- costs for setup changes on the machine
- costs for revenue loss if we produce less than what we could sell to the customer

$$min \quad TF \quad = \sum_{i \in I} \left(k_M * \hat{M}_i + \sum_{l=1}^{6} k_l * E_{l,i} \right) + \sum_{j \in J} k_R * (1.10 * x_j - X_j) \quad (1)$$

s.t.

3.5 Constraints for connecting the variables $\hat{M}_i$ and $M_{i,j,k}$

A motherroll is used, as soon as one of it's tracks is used.

$$M_{i,j,k} \leq \hat{M}_i \quad \forall i, j, k \tag{2}$$

If we use a motherroll, at least one of its tracks has to be used

$$\hat{M}_i \leq \sum_{(k,j) \in K \times J} M_{i,j,k} \qquad \forall i \tag{3}$$

3.6 Constraints for properties of products on motherrolls

If a product is put on a motherroll, all the colors of the product are used on that motherroll.

$$M_{i,j,k} * N_{col}(j) \leq \sum_{c \in C(j)} \alpha_{i,c} \quad \forall i, j, k \tag{4}$$

Only seven different colors can be used on the same motherroll

$$\sum_{c \in C} \alpha_{i,c} \leq 7 \quad \forall i \tag{5}$$

If a product is put on a motherroll, the motherroll has to have the opening type for that product

$$M_{i,j,k} \leq \beta_{i,v(j)} \quad \forall i, j, k \tag{6}$$

All products on a motherroll have to have the same opening type

$$\sum_{v \in V} \beta_{i,v} \leq 1 \quad \forall i \tag{7}$$

If a product is put on a motherroll, the size of the product has to be used on that motherroll

$$M_{i,j,k} \leq \gamma_{i,s(j)} \quad \forall i \tag{8}$$

All products on a motherroll have to have the same size

$$\sum_{s \in S} \gamma_{i,s} \leq 1 \quad \forall i \tag{9}$$

If a product is put on a motherroll, the dot area for that product has to be used on that motherroll

$$M_{i,j,k} \leq \delta_{i,d(j)} \quad \forall i, j, k \tag{10}$$

All products on a motherroll have to have the same dot area

$$\sum_{d \in D} \delta_{i,d} \leq 1 \quad \forall i \tag{11}$$

If a product is put on a motherroll, the trapping rate of the product has to be used on that motherroll

$$M_{i,j,k} \leq \epsilon_{i,t(j)} \quad \forall i, j, k \tag{12}$$

All products on a motherroll have to have the same trapping rate

$$\sum_{t \in T} \epsilon_{i,t} \leq 1 \quad \forall i \tag{13}$$

If a product is put on a motherroll, the paper for that product has to be used on the motherroll

$$M_{i,j,k} \leq \zeta_{i,p(j)} \quad \forall i, j, k \tag{14}$$

All products on a motherroll have to have the same paper supplier

$$\sum_{p \in P} \zeta_{i,t} \leq 1 \quad \forall i \tag{15}$$

3.7 Constraints for encoding the change of properties of motherrolls

The next block of constraints connects the variables encoding the use of certain product properties on the motherrolls to the variables encoding the change of this properties.

A change of color c on the i-th motherroll happens if and only if

$(\quad \alpha_{i-1,c} + \alpha_{i,c} = 1 \quad) \quad mod \quad 2.$

We want to express

$e_{1,i,c} = (\alpha_{i-1,c} \oplus \alpha_{i,c})$

where $\oplus$ stands for XOR. The following inequality constraints express that relationship with linear constraints:

$$\begin{aligned} e_{1,i,c} &\leq \alpha_{i-1,c} + \alpha_{i,c} \\ e_{1,i,c} &\geq \alpha_{i-1,c} - \alpha_{i,c} \\ e_{1,i,c} &\geq \alpha_{i,c} - \alpha_{i-1,c} \\ e_{1,i,c} &\leq 2 - \alpha_{i-1,c} - \alpha_{i,c} \\ &\forall i, c \end{aligned} \tag{16}$$

$e_{1,i,c}$ denotes if color c has changed on the i-th motherroll, it has the following connection to the variable $E_{1,i}$ that denotes how many colors have changed on the i-th motherroll:

$$E_{0,i} = \sum_{c \in C} e_{1,i,c} \quad \forall i \tag{17}$$

The next constraint connects the variables encoding how many colors change ($E_{0,i}$) to the variables encoding if any color changes ($E_{1,i}$).
$E_{0,i} > 0 \quad \Leftrightarrow \quad E_{1,i} = 1$
$E_{0,i}$ can utmost be seven, since only seven different colors are allowed on a motherroll.

$$7 * E_{1,i} \geq E_{0,i} \geq E_{1,i} \tag{18}$$

The variables encoding a change of the opening type v and the variables and the variables encoding the use of that opening type have a similar relationship:
$e_{2,i,v} = (\beta_{i-1,v} \oplus \beta_{i,v})$

$$\begin{aligned} e_{2,i,v} &\leq \beta_{i-1,v} + \beta_{i,v} \\ e_{2,i,v} &\geq \beta_{i-1,v} - \beta_{i,v} \\ e_{2,i,v} &\geq \beta_{i,v} - \beta_{i-1,v} \\ e_{2,i,v} &\leq 2 - \beta_{i-1,v} - \beta_{i,v} \\ &\forall i, v \end{aligned} \tag{19}$$

There is also a similar relationship between $e_{2,i,v}$ and $E_{2,i}$:

$$E_{2,i} = \frac{1}{2} * \big(\sum_{v \in V} e_{2,i,v} \big) \quad \forall i \tag{20}$$

The factor $\frac{1}{2}$ is there because we double count the changes of the opening type. Since there is only one opening type per motherroll, utmost one opening type can change. Therefore the sum above can only be either 0 or 2.

The same idea goes for all other properties of the product.

Size:

$$\begin{aligned} e_{3,i,s} &\leq \gamma_{i-1,s} + \gamma_{i,s} \\ e_{3,i,s} &\geq \gamma_{i-1,s} - \gamma_{i,s} \\ e_{3,i,s} &\geq \gamma_{i,s} - \gamma_{i-1,s} \\ e_{3,i,s} &\leq 2 - \gamma_{i-1,s} - \gamma_{i,s} \\ &\forall i, s \end{aligned} \tag{21}$$

$$E_{3,i} = \frac{1}{2} * \big(\sum_{s \in s} e_{3,i,s}\big) \quad \forall i \tag{22}$$

Dot area:

$$\begin{aligned} e_{4,i,d} &\leq \delta_{i-1,d} + \delta_{i,d} \\ e_{4,i,d} &\geq \delta_{i-1,d} - \delta_{i,d} \\ e_{4,i,d} &\geq \delta_{i,d} - \delta_{i-1,d} \\ e_{4,i,d} &\leq 2 - \delta_{i-1,d} - \delta_{i,s} \\ & \forall i, d \end{aligned} \tag{23}$$

$$E_{4,i} = \frac{1}{2} * \big(\sum_{d \in D} e_{4,i,d}\big) \quad \forall i \tag{24}$$

Trapping rate:

$$\begin{aligned} e_{5,i,t} &\leq \epsilon_{i-1,t} + \epsilon_{i,t} \\ e_{5,i,t} &\geq \epsilon_{i-1,t} - \epsilon_{i,t} \\ e_{5,i,t} &\geq \epsilon_{i,t} - \epsilon_{i-1,t} \\ e_{5,i,t} &\leq 2 - \epsilon_{i-1,t} - \epsilon_{i,t} \\ & \forall i, t \end{aligned} \tag{25}$$

$$E_{5,i} = \frac{1}{2} * \big(\sum_{t \in T} e_{5,i,t}\big) \quad \forall i \tag{26}$$

Paper type:

$$\begin{aligned} e_{6,i,p} &\leq \zeta_{i-1,p} + \zeta_{i,p} \\ e_{6,i,p} &\geq \zeta_{i-1,p} - \zeta_{i,p} \\ e_{6,i,p} &\geq \zeta_{i,p} - \zeta_{i-1,p} \\ e_{6,i,p} &\leq 2 - \zeta_{i-1,p} - \zeta_{i,p} \\ & \forall i, p \end{aligned} \tag{27}$$

$$E_{6,i} = \frac{1}{2} * \big(\sum_{p \in P} e_{6,i,p}\big) \quad \forall i \tag{28}$$

There is a limit on changes of the configuration on the machine per day

$$\begin{aligned}\sum_{i=1}^{50}\sum_{j=1}^{6} w_j * E_{j,i} &\leq const \\ &... \\ \sum_{i=1001}^{1050}\sum_{j=0\ j\neq 1}^{6} w_j * E_{j,i} &\leq const\end{aligned} \tag{29}$$

The w_j can be interpreted as weights. Some changes might be more restrictive than others.

3.8 Constrains for when and how much can be produced

It depends on the size how much tracks on a motherroll can be used

$$\sum_{(j,k)\in J\times K} M_{i,j,k} \leq \sum_{s\in S} b_s * \gamma_{i,s} \quad \forall i \tag{30}$$

Depending on their due dates, products can only be produced in a certain time window

$$M_{i,j,k} = 0 \quad \forall i \in I_j^c \quad \forall j,k \tag{31}$$

The next constraint calculates how much is produced of each product

$$X_j \quad = \sum_{(i,k)\in I_j\times K} M_{i,j,k} * a_{s(j)} \tag{32}$$

We have to produce (90% - 110%) times the ordered amounts

$$0,9 * x_j \leq X_j \leq 1,1 * x_j \quad \forall j \tag{33}$$

The next constraint guarantees that we really have a sequence on motherrolls, meaning that the second role can only be used if the first one is used etc.

$$\begin{aligned}\hat{M}_1 \geq \hat{M}_2 \geq \hat{M}_3 ... &\geq \hat{M}_{50} \\ \hat{M}_{51} \geq \hat{M}_{52} ... &\geq \hat{M}_{100} \\ &... \\ \hat{M}_{1001} \geq \hat{M}_{1002} ... &\geq \hat{M}_{1050}\end{aligned} \tag{34}$$

Note that Rolls 51, 100, 151, ..., 1001 can always be used because these are the first rolls for each day in the planning horizon.

All non used motherrolls of one day should have the same setup as the last used motherroll of that day. Because that way we can compare the the last used motherroll of one day to the first used motherroll of the next day with our existing constraints.

$$\begin{aligned}
5000 * (\hat{M}_i - \hat{M}_{i+1}) &\leq 5000 - \sum_{m=1}^{6} \sum_{n=i+1}^{50} E_{m,n} \quad \forall i = 1, ..., 49 \\
5000 * (\hat{M}_i - \hat{M}_{i+1}) &\leq 5000 - \sum_{m=1}^{6} \sum_{n=i+1}^{100} E_{m,n} \quad \forall i = 51, ..., 99 \\
&\qquad ...
\end{aligned} \tag{35}$$

4 Model with patterns

Encoding the different properties used on a motherroll is very expensive. Take constraint (6) from the exact model for example. If we have a planning horizon of 21 days (1050 motherrolls) and 100 products we have 700.000 constraints just for encoding the used opening type on a motherroll. Constraints for encoding the properties used on a motherroll are needed for two reasons:

- to make sure all products on a motherroll have the same properties (same size, same opening etc.)
- to model the change of a setup between two motherrolls

The following approach at least solves the first point:

First create for example 1000 patterns for how to organize different products on motherrolls, then try to create a production plan that respects due dates with this patterns using as few motherrolls as possible. All patterns will be valid, meaning the products on one pattern have the same properties and share utmost seven different colors

4.1 Parameter

$D = \{1, ..., 21\}$	set of days in the planning horizon
J	set of products in the planning horizon
$P = \{1, ..., 1000\}$	set of possible patterns
$D(p)$	subset of days where pattern p can be used
$\hat{D}(p)$	subset of days where using pattern p is penalized
P_j	subset of patterns that produces product j
$a_{p,j}$	amount of packs of product j produced by pattern p
c_p	penalty cost for using pattern p in $\hat{D}(p)$
x_j	amount of packs ordered of product j
X_j	amount of packs we produce of product j
M	maximum number of motherrolls we can use per day

4.2 Variables

$\alpha_{p,d}$	amount of pattern p used on day d

4.3 objective function

$$min \quad OF := \sum_{p \in P} \sum_{d \in D} \alpha_{p,d} + \sum_{p \in P} \sum_{d \in \hat{D}(p)} c_p * \alpha_{p,d} \tag{36}$$

s.t.

Following constraint encodes what we produce with the patterns we use:

$$\sum_{p \in P_j} \sum_{d \in D(p)} \alpha_{p,d} * a_{p,j} = X_j \quad \forall j \tag{37}$$

We only want to produce what we can sell to the costumer:

$$0.9 * x_j \leq X_j \leq 1.1 * x_j \tag{38}$$

There is a limit of how many motherrolls we can use per day

$$\sum_{p \in P} \alpha_{p,d} \leq M \quad \forall d \tag{39}$$

For each pattern there are days, where this pattern cannot be used. For example after the due date of a product on the pattern.

$$\alpha_{p,d} = 0 \quad \forall d \in D(p)^c \quad \forall p \tag{40}$$

The number of used patterns has to be an integer:

$$\alpha_{p,d} \in \mathbb{N} \quad \forall p, d \tag{41}$$

4.4 Discussion

This model is very simplified compared to the previous model, but it ignores the configuration changes on the machine completely. It only minimizes the number of motherrolls with a given set of patterns. Including constraints for encoding the properties used on patterns is as expensive as in the other model.

The idea is to use this is as the third phase of a three phase approach. In the first phase I deal with the setup changes by grouping products by their properties and assigning similar products to the same day, respecting due dates of course. The third phase would minimize the motherrolls used on one day with a similar model like the model above.

5 My solution for the problem

In the next three chapters I will describe how I solved the problem. There are many decisions for the production plan. Besides the actual organisation on the motherrolls the company has to decide to produce when and which product, and to deliver when and which product. So I decided to split the problem in phases, where in each phase I handle different decisions. The model with the patterns deals with the problem, that all products on one motherroll have to have the same properties, but it does not model the set up changes from one configuration to the next one. The model with the patterns would be nice, if there were no this set up times in the problem. This was the motivation for the following approach:
In phase one we deal with the setup times and take care of the decision of which products are produced on which day. The goal of the first phase is to assign products to days in terms of motherrolls such that the total setup time of the whole planning horizon is minimal.

6 Phase 1

The decision in the first phase is how many motherrolls of each product are assigned to each day. There are no actual patterns of motherrolls calculated in this phase. I assume that every product is put on a motherroll alone. This most simple solution provides an upper bound. The solution of this phase should just provide a grouping of different products to days such that setup time is minimal. Also the decision of what percentage of the order we deliver should not be taken in Phase 1, but in a later phase. So for each product I suppose that there are as many motherrolls to assign, as needed to be able to fulfill the maximum percentage of the order.

6.1 Measure for setup times

Later in the objective function we want to minimize the setup times. Therefore the model needs a measure for the setup times, or equivalent a measure for the heterogeneity of products on the same day. This measure can be different, however in this model I suppose that the setup time is additive. Therefore the measure of heterogeneity will be some linear combination of the number of used colors, used trapping rates, used dot areas etc.

6.2 Parameter

J	set of all products
D	set of all days in the planning horizon
D(j)	set of all days where motherrolls of product j can be assigned to
C	set of all colors
C(j)	set of all colors of product j
V	set of all opening types
M	set of all dot areas
T	set of all trappingrates
P	set of all papertypes
S	set of all sizes
X_j	number of motherrolls we need to schedule of product j
m_α	measure coefficient for colors
m_β	measure coefficient for opening types
m_γ	measure coefficient for sizes
m_δ	measure coefficient for dot areas
m_ϵ	measure coefficient for trapping rates
m_η	measure coefficient for paper types
Max_d	maximum number of motherrolls there can be scheduled on day d

6.3 Variables

$x_{j,d}$	amount of motherrolls of product j scheduled in day d

6.4 Alternative variables

$\alpha_{c,d}$	b.v. that encodes if color c is used on day d
$\hat{\alpha_d}$	number of colors used on day d
$\beta_{v,d}$	b.v. that encodes if opening v is used on day d
$\hat{\beta_d}$	number of different openings used on day d
$\gamma_{s,d}$	b.v that encodes if size s is used on day d
$\hat{\gamma_d}$	number of different dot sizes used on day d
$\delta_{m,d}$	b.v that encodes if dot area m is used on day d
$\hat{\delta_d}$	number of different dot areas used on day d
$\epsilon_{t,d}$	b.v that encodes if trapping rate t is used on day d
$\hat{\epsilon_d}$	number of different trapping rates used on day d
$\eta_{p,d}$	b.v that encodes if paper type p is used on day d
$\hat{\eta_d}$	number of different paper types used on day d

6.5 Objective function and constraints

$$min \quad OF = \sum_{d \in D} \mu(\hat{\alpha_d}, \hat{\beta_d}, \hat{\gamma_d}, \hat{\delta_d}, \hat{\epsilon_d}, \hat{\eta_d}) \quad = \quad \sum_{d \in D} m_\alpha * \hat{\alpha_d} + m_\beta * \hat{\beta_d} + ... + m_\eta + \hat{\eta_d} \tag{42}$$

s.t.

On each day there is a limit on set up changes

$$\mu(\hat{\alpha_d}, \hat{\beta_d}, \hat{\gamma_d}, \hat{\delta_d}, \hat{\epsilon_d}, \hat{\eta_d}) \leq const(d) \quad \forall d \in D \tag{43}$$

The next constraint ensures, that all needed motherrolls are scheduled for each product

$$\sum_{d \in D} x_{j,d} = X_j \quad \forall j \in J \tag{44}$$

There is a limit on how many motherrolls can be scheduled on one day

$$\sum_{j \in J} x_{j,d} \leq Max_d \quad \forall d \in D \tag{45}$$

The next few constraints connect the variables $x_{j,d}$ to the variables encoding the used properties ($\alpha_{c,d}$, $\beta_{v,d}$ ...) In the following constraints M is a very big number.

If we schedule a product on a day, all colors of that product have to be used on that day

$$\alpha_{c,d} \leq x_{j,d} \leq M * \alpha_{c,d} \quad \forall c \in C(j) \quad \forall j, d \tag{46}$$

If we schedule a product on a day, the opening type of that product has to be used on that day

$$\beta_{v(j),d} \leq x_{j,d} \leq M * \beta_{v,d} \quad \forall j, d \tag{47}$$

Likewise for all the other product properties like size, dot area, trapping rate and opening type

$$\gamma_{s(j),d} \leq x_{j,d} \leq M * \gamma_{s(j),d} \quad \forall j, d \tag{48}$$

$$\delta_{m(j),d} \leq x_{j,d} \leq M * \delta_{m(j),d} \quad \forall j, d \tag{49}$$

$$\epsilon_{t(j),d} \leq x_{j,d} \leq M * \epsilon_{t(j),d} \quad \forall j, d \tag{50}$$

$$\eta_{p(j),d} \leq x_{j,d} \leq M * \eta_{p(j),d} \quad \forall j, d \tag{51}$$

The last block of constraints connetcts the variables $\alpha_{c,d}$ to the variable $\hat{\alpha_d}$. Since $\alpha_{c,d}$ encodes the usage of color c on day d, and $\hat{\alpha_d}$ is the total number of colors used on day d, $\hat{\alpha_d}$ is simply the sum of all $\alpha_{c,d}$. The same for all the other variables encoding the product properties on a motherroll.

$$\hat{\alpha_d} = \sum_{c \in C} \alpha_{c,d} \tag{52}$$

$$\hat{\beta}_v = \sum_{v \in V} \beta_{v,d} \tag{53}$$

$$\hat{\gamma_s} = \sum_{s \in S} {}_{s,d} \tag{54}$$

$$\hat{\delta_m} = \sum_{m \in M} \delta_{m,d} \tag{55}$$

$$\hat{\epsilon_t} = \sum_{t \in T} \epsilon_{t,d} \tag{56}$$

$$\hat{\eta_p} = \sum_{p \in P} \eta_{p,d} \tag{57}$$

6.6 A heuristic approach

In this section I will describe the heuristic that was used to solve the problem.

Due to the fact, that products on a motherroll can not differ in any property, there are basically groups of products such that only products within the same group can be combined on motherrolls. Not groups in the mathematical sense, but simply a collection of products having specific properties like this opening, this papertype, this dot area and so on. The first step of the heuristic algorithm is to take the order list and identify the group of each order. Then foreach group you have a list of orders, which is then to be sorted increasing in due dates of the order. Then for each group: go through each element of this sequence of orders and assign the order with the following rule:

Take all the possible days where the order can be scheduled. For each day calculate the increase of the heterogeneity measure if the order was scheduled on that day. An empty day gets some constant as its increase of heterogeneity instead of the actual increase. If an empty day would get the actual increase a product would only be scheduled on an empty day if it shares no properties with any products on the other days. This is not good since I want to have similar products scheduled on the same day. If a product is not similar to any already scheduled product, the algorithm shell schedule it on an empty day. Once this calculation is made for every day, schedule as much motherrolls as possilbe on the "best" day. If there are motherrolls left, schedule as much as possible in the second best day and so on until there are no more motherrolls left to schedule or no more possible days . In the second case put the order into a list. This list of unscheduled orders is also an output of the algorithm, which can rerun the algorithm with looser restrictions on when an order can be scheduled on a day.

7 Phase 2

In Phase one products have been assigned to days. The solution of phase one provides the information of what groups of products are scheduled on each day. In phase two the goal is to optimize the order of the groups for each day, such that the total setup time is minimized. The model will use binary variables to encode whether a group is to be done before another group or the other way around. The problem here is, that a group may occur more than once in the planning horizon. I solve this problem by assigning every group a different coding even though some of them are technically the same group.

7.1 Parameter

M	set of days in the planning horizon
J	set of all groups
G_j	group j
S_m	subset of groups scheduled in day m
$p_{i,j}$	setup time for switching from group i to group j, of course $p_{i,j} = p_{j,i}$

7.2 Variables

$$a_{j,i} = \begin{cases} 1 & \text{if group j is to be done before group i} \\ 0 & \text{else} \end{cases}$$

$$b_{j,i} = \begin{cases} 1 & \text{if group j is done immediately before group 1} \\ 0 & \text{else} \end{cases}$$

$$f_i = \begin{cases} 1 & \text{if group i is the first group in the sequence} \\ 0 & \text{else} \end{cases}$$

$$l_i = \begin{cases} 1 & \text{if group j is the last group in the sequence} \\ 0 & \text{else} \end{cases}$$

7.3 Model

$$min \quad OF = \sum_{i,j \in J \quad i<j} p_{i,j} * b_{i,j}$$

(58)

s.t

If group i is scheduled before group j, and group j is scheduled before group k then group i has to be scheduled before group k, so we want to express: $a_{i,j} = 1, a_{j,k} = 1 \quad \Rightarrow \quad a_{i,k} = 1$

$$a_{i,j} + a_{j,k} - 1 \leq a_{i,k} \quad \forall i, j, k \tag{59}$$

If group i is scheduled immediately before group j, group i has to be scheduled before group j.

$b_{i,j} = 1 \quad \Rightarrow \quad a_{i,j} = 1$

$$b_{i,j} \leq a_{i,j} \tag{60}$$

The last constraint also guarantees, that if j is after i it cant be immediately before i.

The next constraint makes sure, that every group has a group scheduled before it unless it is the first group.

$$f_i + \sum_{j \in J} b_{j,i} = 1 \quad \forall i \in J \tag{61}$$

Likewise every group has to have a group scheduled after it unless it is the last group.

$$l_i + \sum_{j \in J} b_{i,j} = 1 \tag{62}$$

Groups not scheduled on the first day cant be the first group in the sequence

$$f_i = 0 \quad \forall \quad i \notin S_1 \tag{63}$$

One group of day one has to be the first group

$$\sum_{i \in S_1} f_i = 1 \tag{64}$$

Groups not scheduled in the last day cant be the last group in the sequence.

$$l_i = 0 \quad \forall \quad i \notin S_{|M|} \tag{65}$$

One group of the last day has to be the last group in the sequence.

$$\sum_{i \in S_{|M|}} l_i = 1 \tag{66}$$

If group i is on an earlier day than group j, it has to be scheduled earlier in the sequence.

$$a_{i,j} = 1 \quad i \in S_m \quad j \in S_n \quad m < n \quad \forall n, m \in M \tag{67}$$

7.4 A heuristic approach

Following heuristic was used to solve the problem:

The idea of the heuristic is to consider the planning horizon as a sequence of days. Go through each day of this sequence and calculate all permutations of groups within that day. To each permutation apply a function that somehow measures the quality of that permutation and pick the permutation with the least value of that function. On each day the algorithm does not know what permutation it is going to pick on the next day, and it does not change the permutation it picked in previous days. In order to describe the function that was used in the heuristic I need a little bit notation.

$\mu(G_i, G_j)$	measure of heterogeneity between group i and group j
$P_{j,d}$	permutation p of day d
$P_{j,d}(i)$	group scheduled in place i in permutation j of day d
P_d	permutation the algorithm chose for day d
n_d	number of groups in day d

The next function is a measure of setup time of a permutation within a day.

$$H(P_{j,d}) = \sum_{i=1}^{n_d-1} \mu(P_{j,d}(i), P_{j,d}(i+1)) \tag{68}$$

This function was used in the heuristic:

$$Q(P_{j,d}) = \mu(P_{d-1}(n_{d-1}), P_{j,d}(1)) + H(P_{j,d}) + min_{k=1}^{n_{d+1}}\{\quad \mu(P_{j,d}(n_d), P_{j,d+1}(k))\} \tag{69}$$

The first term gives the setup time between the last group of the previous day and the first group of the current day. With the last term the algorithm looks optimistic to the next day assuming it will pick a permutation that starts with the group that fits best to the last group of the current day.

8 Phase three

In this phase the calculation of the actual configurations on the motherrolls happens. I use a pattern model similar to the one explained in Chapter four. Mathematically a pattern is a vector $\vec{p} \in \mathbb{R}^n$. The dimension is the number of products in the planning horizon, and the j-th element describes how many tracks of product j this pattern produces.

For example:

$$\begin{bmatrix} 1 \\ 2 \\ 1 \\ 0 \\ \vdots \end{bmatrix}$$

This would mean one track of product 1, two tracks of product 2 and one track of product 3. In phase one products have already been assigned to days, therefore only products of the same day will be combined on a pattern. There is a limit on how many patterns (motherrolls) can be used per day. This limit depends on the day, since on some days there is more setup time needed than on others. In the first and second phase it has not yet been decided what percentage of the order is going to be fulfilled. This decision will be made now, for each product there is a lower bound and an upper bound of tracks.

8.1 Paramteter

P	set of patterns we want to use
P_d	subset of patterns of products of day d
m_d	max number of motherrolls(patterns) we can use on day d
$\vec{l}$	vector describing the least amount of tracks needed for each product
$\vec{u}$	vector describing the maximum amount of tracks needed for each product

8.2 Variables

α_p	number of pattern p used

8.3 Model

$$min \quad OF \quad = \sum_{p \in P} \alpha_p \tag{70}$$

s.t

For each day we have a limit on how many patterns (motherrolls) we can use.

$$\sum_{p \in P_d} \alpha_p \leq m_d \tag{71}$$

For each product we have to produce a number of tracks that is in between the range.

$$\vec{l} \leq \sum_{p \in P} \alpha_p \leq \vec{u} \tag{72}$$

As I already stated in the introduction this model is very similar to a one-dimensional cutting stock problem. Literature provides lots of different solving approaches to these kind of problems (Valerio de Carvalho, 1998; Braga, Alves, Marcedo, Valerio de Carvalho,2016).

9 Conclusion

Even though the exact model exists only for theoretical purposes and might appear useless for a real world application like this, it is definitely one of the most important elements. Developing an exact model not only translated the problem into the mathematical language, it also showed where the difficulties are. The problem with modeling the change of setup led to the three-phase-approach where in the first phase the algorithm takes care of the setup time and only in the third phase decides on actual arrangements of products on motherrolls.

While working for MCP, I also implemented code to create some test data in order to both test the performance of the algorithm and improve the algorithm. The algorithm was very fast, taking only 2 to 5 seconds to compute a solution for a 21-days planning horizon with 200 orders and 3 motherrolls per order on average. After each phase I displayed the output in order to be able to see the performance of the algorithm at each phase. The idea of the phase one heuristic algorithm turned out to work quite well. Running through the list of orders and schedule the orders with strict restrictions and output the unscheduled orders to rerun the algorithm again with looser restrictions has led to a very good grouping of the products. One problem at this stage was, that sometimes leftover orders could not be scheduled even with very loose restrictions on when an order can be scheduled on a day. Also it sometimes occurred that there was a huge difference in scheduled motherrolls between days, while on some days capacity was completely used up, others still had 10 to 20 motherrolls capacity left. My colleges and I came to the conclusion that this result was because of the just-in-time-scheduling-restriction. With the second phase implementation there were no issues. In the heuristic algorithm I assumed that on each day it will choose a permutation where the first group is the best fit to the last group of the previous day. This assumption proved to be reasonable, this was almost always the case. The third phase on the algorithm also performed quite well, above all if the problem could be solved without waste. If the problem couldn't be solved without waste, patterns including "waste" had to be generated. The algorithm became slow if it had to deal with "waste"-patterns. Of course it has to do with the colors how much waste there is because the colors decide what products in a group can be combined on a motherroll. So I experimented a little bit with the colors by generating different data. It was to observe that if the products had 4.5 colors on average the algorithm found a solution without waste every time. But when the products had 5.5 colors on average the algorithm always had to deal with waste.

A big benefit of splitting the problem up into phases and having independent algorithms for each phase is, that it can work together with the customer very well. The customer does not yet use any computer assistance on calculating a production plan. The workers do the production planning, who have a great experience with. The three pattern algorithm blends in with the humans, because they might decide to come up with a solution of a phase themselves and let the algorithm continue working on the other phases with their solution. For example: the workers could provide patterns and the computer could try to find a solution with this patterns.

I think my work for MCP has left a good foundation to build on a solution for their customer. An improvement of the current procedure could be to make the phases interact with one another. In the heuristic algorithm of the first phase products are assigned to days in terms of motherrolls. The number of motherrolls scheduled on each day represents an upper bound of needed motherrolls. At some point of this algorithm one can use the algorithm of phase three to actually calculate the currently needed motherrolls for each day. This would improve the phase-one-algorithm a lot, since it would know the actual capacities of days when scheduling products.
Also the dependency between heterogeneity of products on a day and the capacity of that day can be implemented in the algorithm. Furthermore in the first and second phase a heuristic algorithm like Simulated Annealing (Ozdamar, Padamallu, 2010; Anand, Saravanasanka ,Subbaraj, 2012) could be implemented, that uses the solution of the current algorithm as a starting point and improves it.

References

S. Knopp, S. Dauzere-Peres, C. Yugma (2017), "A batch-oblivious approach for complex Job-Shop scheduling problems", European Journal of operational research, Vol 263(1) pp 50 - 61

J. M. Valerio de Carvalho (1998), "Exact Solution of Cutting Stock Problems Using Column Generation and Branch-and-Bound", International Transaction in Operational Research. Vol 5 No. 1 pp 35 - 46

N. Braga, C. Alves, J. M. Valerio de Carvalho (2016), "Exact Solution of Combined Cutting Stock and Scheduling Problems"

Anand S., Saravanasankar S. Subbaraj P. (2012), "Customized simulated annealing based decision algorithm for combinatorial optimization in VSLI floorplanning model", Computational Optimization and Applications Vol 52(3) pp 667 - 689

O. Linet, P. C. Sekhar (2010), "New Simulated Annealing Algorithms for constrained optimization", Asia-Pazific Journal of Operatioinal Research Vol 27(03) pp 347 - 367