Joint replenishment problem with shifting time-lines

R.P.C. Zoun (384414)

July 3, 2016

Supervisor: Dekker, R.   Second assessor: Mulder, J.

Bachelor Thesis
Econometrie en Operationele Research

Erasmus University Rotterdam
Erasmus School of Economics

Abstract

In this paper, we do research to an extension of the joint replenishment problem. In our cases, we look to shifting time-lines of the products. In the solution of the standard joint replenishment problem, all the products are included in the first order. However, in our problems this is not always the case. It could be more optimal to start the time-line of a product on a later order. Where the time-line of a product begins at the first replenishment of that product and ends with the last one between the time-interval. We show that shifting is beneficial and will obtain a lower total average cost. It is also improving the fill rate of the trucks and lowers the average number of trucks. Besides the proposed methods to solve the joint replenishment problem with truck limitation and shifting time-lines, we also give an order heuristic which give an optimal solution with a low computation time. The solution of all the methods are compared and the influence of different values for the several variables are investigated.

Contents

1 Introduction

Managing your inventory is a difficult task. To make orders for the replenishment of your products, you need to know when and how much you need of a product. To make this more complex, we like to look at joint replenishment. When an company has multiple product, it is cost saving when you order them together. This can lead to a decrease in transport costs. This kind of problems is referred as joint replenishment problems (*JRP*).

In this paper we will use an extension, we consider the case that it is possible to shift with the time-lines of the products. The time-line of a product is the time from the first replenishment of a product till the last one. If we solve the joint replenishment problem, all the replenishment's will start immediately at the first time-period. This could be non-optimal, we can let the time-lines of several product begin on a later time-period. This could lower the total required number of trucks.

First, we will give a review over the existing and interesting literature. We discuss some methods and ideas to use for our research. After this, we will start with showing a formulation of the *JRP* with the corresponding parameters and variables. We will focus on the JRP with truck limitations. This means that we do not consider constraint on the budget or storage capacity like is done in some other literature.

Next we construct a method to optimize the solution further. We will use shifting time-lines and we will evaluate the consequences on the optimal solution and the corresponding functions. We will give a formulation to solve the shifting time-lines problem and give an example to give some more insight in the methodology. After this, we will give some methods to solve the *JRP* with truck limitations and shifting time-lines. We will modify the algorithm of Goyal, with help of our shifting time-lines formulation and we will extend this with an iteration method. At last, we will give a heuristic to solve the problem without a constant basic cycle time.

At the end, the results of the different methods are discussed and compared. We will evaluate the effects of the several variables and show the effects of the shifting time-lines on the decision variables. We will also give the results of the heuristic and compare them with the shifting algorithms. We will evaluate the average number of trucks used in the solution of the methods and we compare the outcomes of different cases.

With our research we will try to answer the question: Which benefits are there if we take in to account shifting time-lines, and what is the effect on the optimal solution?

2 Literature review

In the last 30 years there is a lot written about the joint replenishment problem. They try to determine an econometric policy to handle the *JRP* in a smart and quick way and resulted in a solution that was near by the optimal one. Khouja and Goyal (2008) gave a review over the research that has been done and show some heuristics, methods and extensions of the *JRP*. The research is most often based on the algorithm of Goyal (1974). Which came up with a solution that determines the optimal order frequency for all the items in the *JRP*. Van Eijs (1993) modified this algorithm and created a good manner to solve the basic *JRP*, which considers a situation with constant known demand over a long period of time. This is a method for the deterministic side of the problem, which also will be considered in this paper. There are also methods for the situation when the demand is not constant. In Lee and Chew (2005) they considered stochastic demands and in Minner and Silver (2005) they have demands that are Poisson distributed. However, we will not discuss this side of the *JRP* in this paper.

For a long period of time, we have tried to find the optimal solution of the *JRP*. Khouja and Goyal (2008) concluded that this took a lot of time and that it is more interesting to add extensions to the *JRP*, so we will get a more realistic model. The policy developed by Goyal (1974) is a strict cyclic policy. This means that we assume that the lowest reorder interval is equal to one and every order contains products. Porras and Dekker (2008) suggested a correction factor for empty replenishment that will occur when the lowest reorder interval is bigger than one. This is called a cyclic policy and holds for all integer values of the reorder intervals. If there are no empty replenishment's the model with the correction factor will be equal to the normal form. So, it will always give an equal or better solution than the standard form.

In the *JRP* we want to minimize the total cost. The literature considers two types of costs. We have the major costs, which are cost that are made per order. We also have the minor costs, which are made when a product is included in an order. There are a lot of experiments been done, where the relation between the optimal solution and the height of the major costs is examine. It has been proven that different methods are optimal for different values of the major costs and also the ratio between the major and minor costs has a big influence on the optimal policy. In our paper we will obtain a solution for different major costs and more interesting, we will look to the situation when the minor costs are zero.

In the standard *JRP* there are no constraints on the storage and transport capacity and on the budget. These were added by Hoque (2006), where the transport capacity is the most interesting one. In the paper there is a limit per item. Which is not really a real life implication. It is more interesting to investigated the situation that there is a limitation on the whole truck. For this transformation they made use of the average number of trucks in the system. We will use this method in our paper, however we will not consider the storage and budget constraints.

Recent articles show us that the research has moved to solving real-life aspects of the *JRP*. An example of this is the adjustment of defective items, which was done by Paul et al. (2014). When the products are transported, they could be damaged and useless. You could take this into account when we place the order for the replenishment of the products. However, our opinion is that it will not have a huge impact on our optimal solution. We could also consider different requirements for the products in our *JRP*. Because this are rare cases, we will not consider them in our paper. At least we want to point out another interesting addition, Salameh et al. (2014) discussed the case of substitution products. We will not discuss this, because the lack of time. Nevertheless, it is quit interesting.

The *JRP* is NP-Hard, which was proven by Arkin et al. (1989). This has a huge impact on the computation time. To solve the problem we need to use the algorithm of Goyal (1974) to solve the problem in a limited time. However, the literature gives also other possible heuristics. Silver (1976) gave a simple heuristic for determining the optimal operating policy. In this method the multiplier per item depends on the multipliers of the other items with a lower value. This means that the lowest multiplier has a huge impact on the solution. This is not always optimal. Also Wildeman et al. (1997) came with a different approach. The problem with solving the *JRP* is long running times for continues time-periods. They used Lipschitz optimization and had running times that increases linear in the number of items. Another way to lower the computation time is constructing a lower and upper the basic cycle time. This was already presented by Goyal (1974) and was later improved by Viswanathan (1996).

Another method to solve the *JRP* is to use a genetic algorithm(*GA*), which is done by

Ongkunaruk et al. (2016). Like in our case, the *GA* uses integer number of trucks. There are a number of trucks available and the *GA* tries to fill these trucks minimizing the total costs. The cost of an empty truck will be zero, so the *GA* will also minimizing the used number of trucks. The *JRP* of Ongkunaruk et al. (2016) has also constraints on defective items and the screening of products. Our opinion is that in reality this happen, however we see this as two separate problems. In Ongkunaruk et al. (2016) they also look to the case that some of the products can not transported together. This because requirements of the products, for example temperature. We do not think this is an important extension, an it is very hard to shift with time-lines if we implement this constraint. If we omit the constraint that are not necessary for our case, the *GA* is a logical and clear method to use. Our biggest concern is that the computation time will be to large. That is why the algorithm of Goyal (1974) is more practical.

The literature gives also some new ideas to consider for determining the optimal replenishment policy. Goyal and Satir (1989) came up with the idea to make the multipliers not longer integer, this will create different replenishment cycles at the same time. Because of the time-limit of our research, we will not investigate this. However, this could be interesting for further research. Another idea was to look only at multipliers with a value of the power of two. Which was done by Lee and Yao (2003). It has been proven by Jackson et al. (1985) that the solution will not be more than 6% away from the optimal minimum costs. We will discuss this case in our paper to evaluate the effect on the computation time and the optimal solution.

3 Problem Definition

In the paper of Hoque (2006) the joint replenishment problem is extended by adding storage and transport capacities and budget constraints. In this paper we will not consider the constraints for the storage and budget. We like to focus on the transport capacities and optimize the required number of trucks to minimize the total costs. First we introduce the variables and parameters for the formulation of our JRP problem.

3.1 Parameters and variables

n := number of products

S := major ordering cost

T := time between two orders, called basic cycle time (**decision variable**)

w := total number of pallets per truck

c := cost per truck

j := designated product, with j ∈ {1,2,...,n}

D_j := annual demand rate for product j

S_j := minor ordering cost for product j

h_j := unit inventory holding cost per unit per year for product j

$H_j := \frac{h_j D_j}{2}$

w_j := fraction of pallet space for one product

m_j := number of full truck loads for product j

T_j := time between two consecutive orders of product j , called the reorder interval

$k_j := \frac{T_j}{T}$ (**positive integer**)

3.2 Assumptions

In our problem definition, we made the following assumptions.

1: The lead time is equal to zero

2: The demand is constant

3: Shortages are not allowed

4: Quantity discount is not possible

5: Transportation times are insignificant and hence ignored

6: There is a k_j equal to one

3.3 Objective function

We want to minimize the total costs for all the replenishment's of every product. We have costs for an order, called major costs. The second part consists of the holding costs and the cost per truck for every product *j*, called minor costs. In this paper we make an adjustment on the formulation used in Hoque (2006), which is given by

Hoque JRP

$$\text{minimize} \quad \frac{S}{T} + \sum_{j=1}^{n} \left(\frac{S_j + m_j c}{T_j} + H_j k_j T \right)$$

with

$$m_j = \frac{w_j k_j T D_j}{w} \tag{1}$$

In this formulation we don't have the case that the sum over all the m_j is always a positive integer value. In fact, most of the time the variable will be not integer. This is not a realistic situation, in practice they will round up the sum of all the m_j. Of course, we can't only drive a peace of a truck. To make this adjustment we follow the method in Guijarro, they make use of the average number of trucks used in the total cycle time. The only difference is that we will round up the number of trucks per time-period and take the average of all those integers. We introduce a new variable *t*, which indicates the time-period. A time-period is the moment when an replenishment occur. The total number of time-periods is equal to the least common multiple (*lcm*) of all the reorder intervals. This means that the number, equal to the amount of time-periods, is divisible by all the k_j. This will create a full reorder cycle for every product. The first time-period is equal to 1 and the interval from the first time-period till the time-period equal to the *lcm* will be referred to the time-period cycle. We also define a new binary variable $O_{jt}(k_j)$ which indicates if an order of time-period *t* contains a certain product with reorder interval multiplier k_j.

$\lceil X \rceil$ will give the rounded up value of a value X. Now the average number of trucks is given by

$$B = \frac{\sum_{t=1}^{lcm} \lceil \sum_{i=1}^{n} O_{jt}(k_j) m_j \rceil}{lcm(k_1, ..., k_n)} \tag{2}$$

We can now define the total cost with truck limitations, which depends on the basic cycle time and all the reorder interval multipliers. We will also differ here from the method used by Guijarro. We will distinguish the cost per order S, like administration costs, and the cost per truck c. The average number of trucks B will correspond with the cost per truck. We want to minimize the total cost as objective function for our *JRP*:

$$TC(T, k_1, ..., k_n) = \frac{Bc + S}{T} + \sum_{j=1}^{n} (\frac{S_j}{T_j} + \frac{1}{2} H_j k_j T) \tag{3}$$

3.4 Formulation

Differently done by Hoque (2006), we don't take into account the storage and budget constraints. We only have the constraints that all the multipliers k_j needs to be positive integers and that $T_j = k_j T$ holds for every product j. We can implement this in the objective function, by filling in T. We can also do this for H_j for more specification. The final formulation:

JRP with truck limitations

$$\begin{aligned} \text{minimize} \quad & \frac{Bc + S}{T} + \sum_{j=1}^{n} (\frac{S_j}{k_j T} + \frac{1}{2} h_j D_j k_j T) \\ \text{subject to} \quad & k_j \in \mathbb{N}, \ j \in \{1, 2, ..., n\} \end{aligned}$$

4 Shifting time-lines

In the algorithm of Goyal (1974) the optimal multipliers k_j with j = 1,..,n are determined. These multipliers give the reorder interval per product and the replenishment's of one order need to fit in all the truck that are used at that time. It can be better to shift some of the reorder time-lines of the products. This could result in a decrease of the required number of trucks.

We consider the following simple example:

Table 1: Example for shifting time-lines

		number of trucks needed					
Product j	k_j	t=1	t=2	t=3	t=4	t=5	t=6
1	1	0.50	0.50	0.50	0.50	0.50	0.50
2	2	0.50		0.50		0.50	
3	2	0.50		0.50		0.50	
Total		1.50	0.50	1.50	0.50	1.50	0.50
Trucks		2	1	2	1	2	1

In this example the values of the reorder multipliers are 1 and 2. The least common multiple will be equal to 2. We give some more time-periods to make the method more clear. In this case we have three times the time-period cycle. For every cycle we need 3 trucks, so in total we need 9 truck for this solution. However, we can shift the time-line of the second product to reduce the total number of trucks. The result is shown in table 2

Table 2: Example for shifting time-lines

		number of trucks needed					
Product	k_j	t=1	t=2	t=3	t=4	t=5	t=6
1	1	0.50	0.50	0.50	0.50	0.50	0.50
2	2		0.50		0.50		0.50
3	2	0.50		0.50		0.50	
Total		1.00	1.00	1.00	1.00	1.00	1.00
Trucks		1	1	1	1	1	1

The total number of trucks needed for every time-period cycle is now decreased to a value of 2. In total for the whole example the number of trucks decrease from 9 to 6. If we have more products or time-periods, the total number of trucks can be reduced with a larger amount.

4.1 Shifting time-lines problem

The above method minimizes the number of trucks over a given period of time. We could use this idea to lower the average number of trucks B of equation 2, which will have an effect on the total optimal costs. To do this, we need that the reorder interval multipliers $\{k_1, ..., k_n\}$ are fixed. Because we know the k_j for every product, the only variable will be the decision when the replenishment is included in an order. To find the optimal shifting, we calculated the total number of trucks used in the whole time-period cycle. To provide this, we define the following parameters and variables:

t^{max} := least common multiple(*lcm*) of $\{k_j,...,k_n\}$

t := time period, with $t \in \{1, ..., t^{max}\}$

N_t := number of trucks at time period t **(integer)**

x_{jt} := number of trucks at time period t **(real positive)**

$$y_{jt} = \begin{cases} 1 & \text{if product j is included in the order at time-period t} \\ 0 & \text{otherwise} \end{cases}$$

$$\text{where } t \leq k_j \qquad\qquad \forall j$$

We can now solve this problem with the following formulation:

Shifting time-lines problem formulation

$$\begin{aligned}
\text{minimize} \quad & \sum_{t=1}^{t^{max}} N_t \\
\text{subject to} \quad & \sum_{j=1}^{n} x_{jt} \leq N_t && \forall t \\
& x_{jt} = x_{j(t+k_j)} && \forall j, t \in \{1, 2, ..., t^{max} - k_j\} \\
& x_{jt} = m_j y_{jt} && \forall j, t \in \{1, 2, ..., k_j\} \\
& \sum_{t=1}^{k_j} y_{jt} = 1 && \forall j \\
& x_{jt} \geq 0 && \forall j, t \\
& y_{jt} \in \{0, 1\} && \forall j, t \in \{1, 2, ..., k_j\} \\
& k_j \in \mathbb{N} && \forall j
\end{aligned}$$

The objective function minimize the total number of trucks in the total period of time. So we summon over all the time periods till t^{max}. This maximum value of t is calculated by finding the least common multiple of all the k_j. This is required because the maximum value needs to be a multiple of the reorder intervals of the products. If this is not the case, it is possible to remove one replenishment of a product by starting the time-line on a later time-period. The last replenishment in the last reorder interval could be higher than the maximum value of t and will not take place, this will give a wrong impression of the minimum costs. If all the reorder multipliers are even or a power of two, then we can take the maximum reorder interval as the maximum value of t.

In the first constraint the number of trucks per time-period needs to be higher than the required truck space. This truck space could be non-integer. Because we can't drive a piece of a truck the total number of trucks N_t will always be the rounded up value of the total required truck space. The second constraint covers that the replenishment of the products are cyclic. The truck space of a product at a time-period needs to be equal to the trucks space a cycle later. So their will always be k_j time-periods between two replenishment's. When there is no replenishment in a time-period, the truck space will be zero and from the second constraint follows that the time-period of a cycle later will also be zero. The constraint is for all time-periods until $t^{max} - k_j$, otherwise the algorithm will try to find the x variables for time-periods higher than t^{max}.

To provide that the replenishment's will have the right value, we have the two remaining constraints. The time-line needs to start in one of the time-periods of the first reorder interval of the product. This means before the value of k_j. This is covered by the fourth constraints. When the time-line of a product begins at a time-period the value of the corresponding y will be 1. This means that the corresponding x will be equal to the truck space m_j. When the value of y is zero, the value of x will also be zero. We do not need a variable y for every time-period, because the second constraint already covers that the replenishment's are cyclic.

After a few tests we concluded that the computation time of this method is not very large, we can implement the formulation into the algorithm of Goyal (1974). Where the total cost will decrease with help of the above formulation. We can adjust the average number of trucks B with the solution of the shifting time-lines problem. We have

$$B = \frac{\sum_{t=1}^{t^{max}} N_t}{t^{max}}$$

4.2 Example shifting formulation

We consider the following example to give some more insight of the shifting algorithm. For this example we consider five products. Every product has his own demand rate D_j. Which is respectively {8.1, 3.0, 0.8, 0.8, 0.3}. We also have the variables w_j which indicates how many pallets we need per product. In this example we have the following vector for w: {0.2, 0.2, 0.4, 0.4, 0.8}. So, for product 1, we can place five products on a pallet. The total number of pallets in a truck has a maximum W of 24. We assume that from earlier calculations we obtain an optimal basic cycle time T of 7 and the optimal reorder interval multipliers vector k: {1, 2, 3, 2, 6}. This means that we have a least common multiple of 6. So, we will consider 6 time-periods in our example, after this the time-period cycle will repeat itself.

To find the optimal shifting policy the shifting algorithm will place the replenishment of the product on the best time-period. First, we will calculate the size of the replenishment of each product m_j if they are included in an order. We can use equation 1 and fill in the values of the variables given above. The will result in the vector m: {0.47, 0.35, 0.26, 0.19, 0.42}. Now we can compute the number of trucks for each time-period. In table 3 the 'normal' situation is shown, there is still no shifting involved.

Table 3: Example shifting formulation, still without shifting

		Time-period					
Product j	k_j	1	2	3	4	5	6
1	1	0.47	0.47	0.47	0.47	0.47	0.47
2	2	0.35	0	0.35	0	0.35	0
3	3	0.28	0	0	0.28	0	0
4	2	0.19	0	0.19	0	0.19	0
5	6	0.42	0	0	0	0	0
Total		1.71	0.47	1.01	0.75	1.01	0.47

For the six time-periods we need in total 9 trucks. We see clearly that this policy would not be optimal, because in the third and fifth time-period we use a truck for only 1 percent of the total available truck-space. The shifting algorithm will find the optimal starting time-periods for the time-lines of the products. If we look to our example, we see that we can not shift the first product. However, all the other products could be shifted. The solution of the algorithm is given as follow

Table 4: Example shifting formulation, with shifting

		Time-period					
Product j	k_j	1	2	3	4	5	6
1	1	0.47	0.47	0.47	0.47	0.47	0.47
2	2	0	0.35	0	0.35	0	0.35
3	3	0	0	0.28	0	0	0.28
4	2	0.19	0	0.19	0	0.19	0
5	6	0	0	0	0	0	0.42
Total		0.66	0.82	0.94	0.82	0.66	1.52

We only need 7 trucks for all the time-periods. The minimum was 6, so that is a small differences. Table 4 shows that, in the optimal solution, a lot of products do not start at the first time-period. This shows the effect of the shifting time-lines.

5 Methods

We propose several methods to calculated the optimal solution of the *JRP* with shifting time-lines. We will evaluate the outcomes of the different methods with each other and with the solution of the *JRP* with only the truck limitations. We will also discuss the correctness of the used methods and the corresponding computation times. First we will solve the problem with a modified version of the algorithm of Van Eijs (1993), which was already a modified algorithm of Goyal (1974). In the second chapter we show how we can calculate the optimal solution by iterating over different values for the reorder interval multipliers.

5.1 Modified algorithm of Goyal

Because we have a problem which includes shifting time-lines, the total costs are no longer linear for T. This means that the algorithm of Goyal (1974) will not be optimal anymore. We will modified some functions of this algorithm, so it will represent our case. The method that we use will be a heuristic because it could be not optimal.

To optimal solution of the *JRP* depends on the basic cycle time T and the reorder interval multipliers $k_1, ..., k_n$. Goyal (1974) derived functions for these decision variables. We will evaluate which functions can be used in our case and which are not applicable for our situation. From equation 3 we can derive the function, for fixed $k_1, ..., k_n$ and fixed B, of the optimal basic cycle time T^*, which is given by

$$T^* = \left[\frac{2(Bc + S + \sum_{j=1}^{n} \frac{S_j}{k_j})}{\sum_{j=1}^{n} k_j D_j h_j} \right]^{\frac{1}{2}} \tag{4}$$

If we combine equations 3 and 4, we get a total average cost per time-period (*TRC*) of

$$TRC = \left[2(Bc + S + \sum_{j=1}^{n} \frac{S_j}{k_j})(\sum_{j=1}^{n} k_j D_j h_j) \right]^{\frac{1}{2}} \tag{5}$$

However, B is a function of T. So if we take the derivative of equation 3 to T, we also need to take into account the derivative of B to T. This is impossible, because we have rounded up values. We need to apply a heuristic and consider the non-rounded up values for deriving the derivative. We will now get an optimal basic cycle time T^* of

$$T^* = \left[\frac{2(S + \sum_{j=1}^{n} \frac{S_j}{k_j})}{\sum_{j=1}^{n} k_j D_j h_j} \right]^{\frac{1}{2}} \tag{6}$$

because

$$B^* = \frac{B}{T} = \frac{\frac{\sum_{t=1}^{lcm} \sum_{i=1}^{n} O_{jt}(k_j) w_j k_j T D_j}{lcm(k_1, ..., k_n) w}}{T} = \frac{\sum_{t=1}^{lcm} \sum_{i=1}^{n} O_{jt}(k_j) w_j k_j D_j}{lcm(k_1, ..., k_n) w}$$

Because the function B^* does not depend on T, the derivative will be zero and that is why we do not find this part back in equation 6. We will combine this new function with equation 3 to obtain the optimal total average cost per time-period, which is given by

$$TRC = \left[2(S + \sum_{j=1}^{n} \frac{S_j}{k_j})(\sum_{j=1}^{n} k_j D_j h_j) \right]^{\frac{1}{2}} + B^* c \tag{7}$$

When we have a fixed T, Goyal (1974) derived a formulation to obtain the reorder interval multipliers separately by minimizing the variable costs per product:

$$k_j(T)(k_j(T) - 1) < \frac{\frac{2S_j}{D_j h_j}}{T^2} \leq k_j(T)(k_j(T) + 1) \tag{8}$$

Because we will shift the time-lines of the products, equation 8 will not be valid in our case. It could be optimal to choose another value for K_j, which results in a lower number of trucks and therefore lower total costs. This depends on the demand of every product and we can not create a formula for this phenomena. That is why we will still use equation 8, because the solution will not far from optimum.

To minimize the total computation time, the algorithm of Goyal (1974) uses a maximum and minimum value for the optimal basic cycle time. T^{max} is given when all the reorder interval multipliers are equal to one. We can calculate this with equation 6, so we will get $T^{max} = T^*(1, 1, ..., 1)$. This is the same as in the standard *JRP*. However, the function for the minimum of the basic cycle time will change. We can derive a function for T^{min} as follow:

$$T^* TRC^* = 2(S + \sum_{j=1}^{n} \frac{S_j}{k_j}) + B^* T^* c$$

We have $TRC^* \leq TRC$, the optimal total average cost will always be lower than the cost of a different solution.

$$T^* \geq \frac{2(S + \sum_{j=1}^{n} \frac{S_j}{k_j}) + B^* T^* c}{TRC}$$

We can obtain the minimum of the basic cycle time from the above formulation and we will get

$$T^{min} = \frac{2S + c}{TRC} \tag{9}$$

5.2 Iterative

Because the solution of the modified algorithm of Goyal (1974) is not always optimal, we like to propose an other method to solve the *JRP* with truck limitations and shifting time-lines. In some cases the number of trucks that we can save by using shifting time-lines, could be higher than for other values of the reorder interval multipliers. This means that equation 8 is not valid anymore. However, the k_j from this equation will be close to the optimal solution. The solution that we got from the modified algorithm will be not further than $\frac{c}{T*}$ of the optimal total average costs. We like to show this with the following example:

Table 5: Example minimum difference of the optimal solution

	Time-period						
	1	2	3	4	...	LCM - 1	LCM
Trucks before	1+ϵ	1+ϵ	1+ϵ	1+ϵ	...	1+ϵ	1+ϵ
Trucks after	1	1	1	1	...	1	2

In the example we have every time-period $1 + \epsilon$ number of trucks. This means that we need at every order 2 trucks. The second truck will be almost empty, so this could be more optimal. Because our method to shift with the time-lines of the products, we could obtain situation two. All the little ϵ of the time-period is now at the last time-period (or another one), so we have a decrease of (LCM-1) number of trucks. We can not decrease the number of trucks anymore, because if we remove the needed truck space at one time-period, we also need to add it at another one. This is the most extreme case and will not happen in practice. That is why we will only look to solution nearby the optimal solution of the modified algorithm. We will only iterate over the k_j that are in the interval -1 and +1 of the multiplier calculated with 8. So if $k_1^*, ..., k_n^*$ are the reorder interval multipliers of the optimal solution from the modified algorithm of Goyal. We have the following

$$k_j^{min} = \begin{cases} k_j^* - 1 & \text{if } k_j^* > 1 \\ 1 & \text{otherwise} \end{cases} \tag{10}$$

and

$$k_j^{max} = k_j^* + 1 \tag{11}$$

The iterative algorithm is an addition to the solution of the modified algorithm of Goyal. The iterative algorithm will try to find a lower total cost nearby the optimal solution that already exist. The algorithm will check all the possible combinations and will return the solution with the minimized cost. The full algorithm is as follow

Iterative algorithm

We define the vector $k^* = k_1^*, ..., k_n^*$, which store the best reorder interval multipliers found so far. We define the value TRC^*, which stores the best total average costs found so far.

Step 1: *Initialization*

```
Solve the JRP with truck limitations and shifting time-lines, with the method of
chapter 5.1 and obtain the optimal vector $k_{goyal}$
$k^* = k_{goyal}$
$TRC^* = TRC(k^*)$, with equation 5
$T^* = T(k^*)$, with equation 6
```

Step 2: *Create interval*

```
$k_j^{max} = k_j^* + 1$
if( $k_j^* > 1$ )
    Set $k_j^{min} = k_j^* - 1$
else
    Set $k_j^{min} = 1$
end
```

Step 3: *Iterative*

```
for(every combination of all $k_j^{new}$, with $k_j^{min} \leq k_j^{new} \leq k_j^{max} \quad \forall j$)
    newK = $\{k_1^{new}, ..., k_n^{new}\}$
    newTRC = TRC(newK), with equation 5
    if( newTRC < $TRC^*$ )
        $k^*$ = newK
        $TRC^*$ = newTRC
        $T^*$ = T(newK), with equation 6
    end
end
```

Step 4: *Solution*

```
$k^{opt} = k^*$
$TRC^{opt} = TRC^*$
$T^{opt} = T^*$
```

5.3 Order heuristic

At last, we give a heuristic for determining when the orders take place and which products in which amount are included in the order. For this order heuristic, we do not have a constant basic cycle time. So, we also do not have reorder interval multipliers anymore. We expect that the total costs of this heuristic will be higher than of the other methods, however the computation time will be very fast. In this method, the variable t is not a time-period, it is the current time in our time-interval. The time-interval will be from zero till t^{max}. For this heuristic we define the following extra variables:

t^{next} := the next time, when there is a new order.

Δt := time between the last order and the next one, so $t^{next} - t$.

v := vector with the stock level for every product at the current time.

λ := order size rate.

Every product starts with an initial stock level. This could also be zero for every product, but we prefer to use non-zero initial stock levels. In the order heuristic, we determine when a new order takes place. To determine the value of t^{next}, we iterate over all the products and find the lowest value for $t + \frac{v_j}{D_j}$ for product j, where t is the current time. So, we determine when the first product is out of stock. After this, we will update the stock levels, which are given by

$$v_j^{new} = v_j - \Delta t D_j \tag{12}$$

Now we have the stock levels before the order. To determine which products are included in the order and in which amount, we follow some steps. First, we have the product that is out of stock. This product will be in the order and the amount will be determined by the equation λD_j. After this, we determine how many trucks we need for this replenishment, this is the rounded up value of $\frac{\lambda D_j}{W}$. If the last truck is not totally full, we like to include other products in the order till the whole truck is full. To determine which product this should be, we again use the method to find the first product with a stock out. There could appear two situations. It could be that the room left in the truck is enough to transport the whole order amount of the determined product, in this case we include this order amount. In the other case, the room in the truck is to small, then we put as many items of the determined product in the truck. We repeat this until there is no place left in the truck. An important remark is that we do not include products that already have a high stock level. This will result in unnecessary expensive holding costs. So, we define a vector LB, which has a lower bound of the stock level for every product, that needs been reached before we can include that product in an order. If we have done all this, we update the stock levels of the products we just order and the current time will be equal to the time of this last order.

We will repeat this whole procedure until we reach t^{max}. At the end, we can determine the total made costs. We have three parts of the total costs. First, we have the setup costs of all the products. So, when a product is included in an order, we need to add up the setup costs of this product to the total costs, also when it is just for one item. Secondly, we have the holding costs. We calculate this costs with the holding rate times the main stock level. We calculate this immediately after we determine t^{next}, the equation is given by

$$\text{holding costs} = \frac{1}{2}\sum_{j=1}^{n} h_j \Delta t(v_j + v_j^{new}) \tag{13}$$

If we determine these costs after every order, we will obtain the total holding costs. The last part of the total costs are the major setup costs and the truck costs. The truck costs are just the number of trucks used in total of all the orders times the cost per truck. The major setup costs are the number of orders times the cost per order. The major costs will have a big influence on the total costs, because we expect a lot more orders.

we have the total costs of the whole period from zero till t^{max}, to obtain the total average costs we just divide the total costs by t^{max}. At the end, we can find an optimal value for the order size rate λ, if you do not have one beforehand, by iterating over an self chosen interval for this variable. Because the total average costs function is not convex for the order size rate, we need to pay attention that we do not end up in a local minimum. We propose the following method, we take the basic cycle time from the algorithm of Goyal with shifting as starting point. Then iterate over all the values from zero till two times this basic cycle time. We choose the order size rate that gives the lowest total average costs.

6 Results

In the last part of this paper, we like to show, compare and discuss our results. We used a big data-set for our main calculations. This data-set is of an original problem from Mexico and was also used by Oudenes. The data of this problem is given in appendix 24 and 25 and will be further referred as the main data-set. In all the problems that are solved we express T in days. First, we give an example of the shifting time-lines. We only use the first 20 products of the main data-set for this example, however we will run the methods on the whole data-set. Only for simplicity, we show only the results of a peace of the data-set. Because the main data-set is not sorted in any way, by picking the first 20 products we selected them randomly. In the second part, we will show the effects if we will change some of the different variables and at least we will compare the solutions of the different methods that were proposed in the last chapter.

6.1 Example Shifting time-lines

We like to examine the effect of taking into account the shifting time-lines. We can split this into two parts, namely the change in the optimal cost and the change in the optimal solution. In table 6 we can see that there are a lot of time-lines that are not beginning at the first time-period. The maximum multiplier has a value of 8, so we give the first eight time-periods to observe when the time-lines of all the products starts. When the product is included in an order at a time-period, this is given by a '1'. In table 6 is only given the first twenty products of the main data-set. The full table with all the products is given in the appendix. The major costs which corresponds to this solution are $S = 500$ and the cost per truck are $c = 1000$. There is space for 24 pallets per truck and the truck-space per product is given in the table in the appendix.

Table 6: Example begin shifting time-lines

		Time-period							
Product	Multiplier	1	2	3	4	5	6	7	8
1	4	0	0	1	0	0	0	1	0
2	5	1	0	0	0	0	1	0	0
3	1	1	1	1	1	1	1	1	1
4	4	0	1	0	0	0	1	0	0
5	1	1	1	1	1	1	1	1	1
6	2	0	1	0	1	0	1	0	1
7	3	1	0	0	1	0	0	1	0
8	7	1	0	0	0	0	0	0	1
9	1	1	1	1	1	1	1	1	1
10	5	0	0	0	1	0	0	0	0
11	7	0	0	0	1	0	0	0	0
12	2	1	0	1	0	1	0	1	0
13	2	0	1	0	1	0	1	0	1
14	8	0	0	0	1	0	0	0	0
15	7	0	0	0	0	0	0	1	0
16	2	0	1	0	1	0	1	0	1
17	1	1	1	1	1	1	1	1	1
18	2	1	0	1	0	1	0	1	0
19	1	1	1	1	1	1	1	1	1
20	1	1	1	1	1	1	1	1	1

When the multiplier of the product is 1, obviously we can not change the starting time-period of the time-line. If we leave out these products, we have 14 products left of the above example. Of those products, there are 9 products that are not included in the first order. If we look at all the products from the main data-set, we can conclude that, in the optimal solution, there are a significant number of products which are not replenished at the first order.

We can not detect a pattern in table 6 for the value of the multiplier of the product. We see that the products 4 and 13 both start at the first time-period. However, products 12 and 18 does not, while they have the same multiplier. There are more products with a multiplier of two that start at the first time-period than products with a multiplier of four and five. This is logical because with a multiplier of two, the first time-period is 50% of the possible starting points and for a multiplier of four this is 25%.

6.2 Comparing methods

We make use of different kind of data-sets to compare the several methods that are proposed earlier in this paper. Because the iterative method can only solve problems with a low number of products in an acceptable period of time, we can not run the main data-set with this method. If we solve the problem for the whole data-set, we will have 3^{83} iterations. Because the computation time of 2000 iterations is approximately one minute, the computation time of all the iterations will be too large, around 3.8^{30} years. We will first use the modified algorithm of Goyal and compare the case without shifting and with shifting of the time-lines. We have the standard situation with $S = 500, c = 1000$ and

$W = 24$. Table 7 contains the total costs and the corresponding optimal basic cycle time for both cases.

Table 7: Number of trucks per time-period

	Optimal T	Optimal TRC	Optimal B
Goyal without shifting	2.24	€5296.33	8.33
Goyal with shifting	2.25	€5150.03	8.03

Table 7 shows us that the optimal basic cycle time is almost equal for both situation. One of the reasons is that the functions for the optimal basic cycle time T and the reorder interval multipliers k_j are the same for both cases. Therefore, the decrease for the total costs is entirely on account of the number of trucks. The reduction is in total €146,30 per day, which corresponds with €53,399.50 per year. We will evaluate the reduced number of trucks in chapter 6.3.

After some evaluation and the knowledge of some shortcomings of the modified algorithm of Goyal, we conclude that the solution of this method is not optimal. To find a better solution we will use the Iterative method as described in chapter 5.2. Because we need to iterate over all possible cases, we need to consider an example with less number of products. For this example we have seven products with the following data, we constructed this data set ourselves:

Seven products data-set

D = {10.56, 20.57, 10.23, 50.23, 60.25, 20.69, 30.01}

h = {0.35, 0.35, 6.0, 6.0, 0.32, 0.32, 19.2}

s = {20, 20, 50, 50, 20, 40, 40}

w = {4, 5, 4, 5, 4, 5, 4}

W = 24

S = 500

c = 1000

This data-set will create a situation were there will be multiple k_j bigger than one. This case will show the effects of the used methods and will help us to compare the proposed methods. First, we will look to the optimal basic cycle time T and the corresponding total average costs TRC. This results are shown in table 8. The Iterative method is always with shifting, because it is an extension for the modified algorithm of Goyal with shifting. The last column gives the differences in percentage of the total average cost between the method and the algorithm of Goyal without shifting.

Table 8: Compare methods, basic cycle time and total average costs

	Optimal T	Optimal TC	gap in percentage
Goyal without shifting	1.23	€3637.61	-
Goyal with shifting	1.15	€3203.00	-11.95%
Iterative	1.19	€3149.53	-13.42%

We see that the optimal T is for all the methods nearly the same. However, there is more change than with the solutions of the main data-set. We see that the costs difference between the 'normal' method of Goyal without shifting and the Iterative method is quite large. There is a gap of almost €500. Also the cost difference between the methods which uses the modified algorithm of Goyal is more than €50. If we compare the reorder interval multipliers of all the methods with each other, we see that considering shifting will give a different solution.

Table 9: Compare methods, reorder interval multipliers

	k_j						
Method	1	2	3	4	5	6	7
Goyal without shifting	1	1	1	1	1	1	1
Goyal with shifting	3	2	1	1	2	3	1
Iterative	3	1	1	1	1	3	1

Table 9 shows that the solution for the k_j of Goyal without and with shifting is totally different. This means that the solution with the lowest cost before shifting is a different one after shifting. The shifting of the time-lines is more optimal for other values of the reorder interval multipliers. Also the Iterative method makes beneficial adjustments for some of the k_j, which lead to a lower optimal cost of €3149.53. An important remark is that the above example does not prove that the k_j will always be going up, it could be that the optimal solution contains lower k_j than those of the solution of the algorithm of Goyal.

6.3 Fill rate and average number of trucks

We need to evaluate another important aspect of the shifting time-lines solution. We can decrease the average number of trucks by optimizing the fill rate. This means that we want to minimize the difference between the number of trucks per order and the rounded up value of this. So, we define the fill rate as, how much the last truck of every order is filled. However, we like to know how much this is improved and if there is more room for improvement. We will look to the solution of the main data-set. So, we have $S = 500$, $c = 1000$ and $W = 24$. The least common multiple is equal to 840 and we will use Goyal with shifting to solve this problem. In table 10, we have the first 20 time-periods and the corresponding number of trucks.

Table 10: Number of trucks per time-period

	Time-period									
	1	2	3	4	5	6	7	8	9	10
Trucks	9.38	6.99	8.28	7.40	8.48	7.13	8.74	7.08	8.68	7.40
	11	12	13	14	15	16	17	18	19	20
Trucks	8.43	6.99	8.93	6.99	8.38	7.54	8.68	6.99	8.74	6.99

The important part of the number of trucks is the fill rate of the 'last truck'. For example, in the first time-period there are 9 full trucks. The tenth truck is full for 38%, So that is less than half a truck. If we look to all the time-periods, this happens quite often. We also see a difference in number of trucks between time-periods, all the values are between 7 and 10 trucks. The average number of trucks over all the 840 time-periods is 8,33. In table 11, we see the number of trucks per time-period after shifting.

Table 11: Number of trucks per time-period after shifting

	Time-period									
	1	2	3	4	5	6	7	8	9	10
Trucks	7.99	7.87	7.87	7.91	7.96	7.81	7.96	7.94	7.90	7.95
	11	12	13	14	15	16	17	18	19	20
Trucks	7.95	7.87	7.92	7.93	7.98	7.96	7.88	7.90	7.93	7.95

We see that all the values are between 7.75 and 8.00. Shifting of time-lines ensures an increase in the fill rate, which will reduce costs. A secondary positive effect is that the solution contains an equal number of trucks for every time-period. Every time-period requires eight trucks, which become handy when the company owns the trucks and not rent them. The average number of trucks are equal to 8.03. This means that we reduce the transport costs with $\frac{0.30*c}{T*}$ per time, where c is the cost per truck and T^* the optimal basic cycle time. In this problem we have a truck cost of €1000. If we express T in days, we can earn €134 per day. Which corresponds with €48910 per year. Therefore, we can conclude that optimizing the fill rate is a good strategy to reduce the total costs.

If we examine the results of the Seven products data-set we see the same pattern. In the table below, we compare the results of the three methods. We show the first 6 time-periods, because this is the largest time-period cycle of the three methods.

Table 12: Number of trucks per time-period, seven products data-set

	Time-period					
	1	2	3	4	5	6
Goyal	2.36	2.36	2.36	2.36	2.36	2.36
Goyal shifting	2.81	0.97	3.79	1.97	2.81	1.94
Iterative	1.94	1.94	2.94	1.94	1.94	2.94

We can see that the fill rate of the modified algorithm of Goyal without shifting is very low. The time-period cycle is equal to one time-periods, so we can see that the fill rate of the last truck is on average 36%. The average number of truck is equal to 3. The fill rate of the second method is already much higher. The method with shifting has on average a fill rate of 88.17% for the last truck. A huge difference, also the average number of trucks decreases to 2,5. Finally, the iterative method shows the finest results. With a time-period cycle of three periods, we can see that the fill rate of the last trucks is equal to 94%. Also the average number of trucks is further reduced till 2,33.

6.4 Different variable values

Last, we want to evaluate some influence of the several variables on the optimal solution. In this chapter we will review what happens with the total cost if we change some of the variables. First, we will look to changes in the costs variables and at the end we will try some different values for the truck capacity. We have the standard situation with $S = 500$, $c = 1000$ and $W = 24$. We will change every time one variable and let the other variables on the original value. First, we looked at different kind of values for the order costs S. The results are shown in table 13

Table 13: Different values for the order costs S, basic cycle time and total average costs

	Without shifting		With Shifting		
S	Optimal T	Total average cost	Optimal T	Total average cost	Cost reduction
100	1.90	€5126.13	1.90	€5053.02	1.43%
300	2.07	€5343.87	2.07	€5096.15	4.64%
500	2.24	€5296.33	2.25	€5150.03	2.76%
800	2.48	€5422.99	2.49	€5324.31	1.82%
1000	2.63	€5485.75	2.63	€5402.49	1.52%

For all the values of S, the shifting has no impact on the optimal T. However, the total average cost changes and has also changes between the different values for S. We notice that the value for $S = 300$ without shifting is higher than the value for $S = 500$. The reason is that the modified algorithm of Goyal will search for the optimal total average cost. However, we took an assumption for this method that the value of the average number of trucks is not rounded up when we take the derivative to T. The values of the method with shifting will be correct, because we do not use this assumption here. The average number of trucks will be calculated with our shifting problem.

In table 14, the results of the seven product data-set are shown. In the last two columns, we see the decrease in total average cost in percentage between the several methods. The first column shows the decrease in percentage of the total average cost between Goyal without and with shifting (Δ *Costs 1*) and the second column shows the decrease in percentage between the Goyal with shifting and the Iterative method (Δ *Costs 2*).

Table 14: Different values for the order costs S, basic cycle time and total average costs for the seven products data-set

	Without shifting		With Shifting		Iterative			
S	T	TRC	T	TRC	T	TRC	Δ Costs 1	Δ Costs 2
100	0.83	€3210.33	0.67	€3103.41	0.72	€2801.35	3.33%	9.73%
300	1.05	€3440.52	0.99	€3352.96	0.91	€2922.43	2.54%	12.84%
500	1.23	€3637.61	1.15	€3303.00	1.19	€3149.53	11.95%	4.65%
800	1.46	€3844.43	1.46	€3479.37	1.39	€3479.37	9.50%	0.00%
1000	1.59	€4064.60	1.54	€3661.24	1.56	€3636.81	9.92%	0.67%

From table 14, we can observe some interesting cases. First, we like to point out that the reduction of the total average cost by the Iterative method is significantly large. The Iterative method is an useful improvement on the algorithm of Goyal with shifting. The second observation is that for higher values of the major setup costs S, the improvement made by the Iterative method is very small. So it seems that for high values of S, the algorithm of Goyal with shifting already gives a good estimation of the total average cost. we think the reason for this is that when the Iterative method makes an improvement, it search for a solution with more orders, but with less number of trucks with help of shifting. When the major costs are very high, the decrease of the transport costs needs to be higher than these already high order costs.

In our methodology, we took the order costs and the transport costs separate. So, in table 15 we will see the results of different values for the transport costs c.

Table 15: Different values for the truck costs c, basic cycle time and total average costs

	Without shifting		With Shifting		
c	Optimal T	Total average cost	Optimal T	Total average cost	Cost reduction
500	2.24	€3434.81	2.24	€3361.88	2.12%
800	2.24	€4551.72	2.25	€4434.96	2.57%
1000	2.24	€5296.33	2.25	€5150.03	2.76%
1200	2.24	€6040.94	2.25	€5865.11	2.91%
1500	2.24	€7157.86	2.25	€6937.72	3.08%

The changes of the costs per truck will not effect the optimal basic cycle time T, because these costs are not present in equation 6. In table 15 we see the pattern that if we increase c, the reduction of the total costs when we use shifting will also be higher. The reason of this is because the average number of trucks corresponds with the costs per truck. So, if the shifting method decreases the number of average trucks, this will immediately have a bigger effect on the total average costs when we have a high value for c.

In table 16, we again look to different values for the truck costs, only now for the seven products data-set. In the table we show the optimal basic cycle time T and the total average costs TRC. The last two columns give the cost reduction in percentage between respectively the first two methods and the last two methods.

Table 16: Different values for the truck costs c, basic cycle time and total average costs for the seven products data-set

	Without shifting		With Shifting		Iterative			
c	T	TRC	T	TRC	T	TRC	Δ Costs 1	Δ Costs 2
500	1.23	€2419.66	1.15	€2191.22	1.19	€2165.30	9.44%	1.18%
800	1.23	€3150.43	1.15	€2798.29	1.19	€2755.84	11.18%	1.52%
1000	1.23	€3637.61	1.15	€3303.00	1.19	€3149.53	11.95%	4.65%
1200	1.23	€4124.79	1.15	€3607.72	1.19	€3543.22	12.54%	1.79%
1500	1.23	€4855.56	1.15	€4214.78	1.19	€4133.76	13.20%	1.92%

We see, as in table 15, that the reduction of the costs is higher when the truck costs are higher. We can also see that, like in table 15, the truck costs has no impact on the optimal basic cycle time. If we evaluate the Iterative method, we like to make a nice remark that for example the total average costs of the Iterative method for $c = 1000$ is lower than the total average costs of the basic method for $c = 800$. So the truck costs are higher, however with shifting time-lines the total costs are lower. Another remark, the reduction in percentage of the total average cost is higher in table 16 than in table 15. In table 16, we run the seven products data-set, because we have a little number of products the truck costs are, in comparison with the other costs, very large. So, this will give a bigger reduction in percentage when we could lose some trucks with help of shifting.

Finally, we want to consider different values for the truck capacity. In the standard situation, we got $W = 24$. We will look to some different truck-space values and we will evaluate the effect on the fill rate of the trucks. The first results are shown in table 17.

Table 17: Different values for the truck-space W, basic cycle time and total average costs

	Without shifting		With Shifting		
W	Optimal T	Total average cost	Optimal T	Total average cost	Cost reduction
10	2.25	€10219.82	2.25	€10162.36	0.56%
18	2.55	€6550.60	2.25	€6401.16	2.28%
24	2.24	€5296.33	2.25	€5150.03	2.76%
30	2.55	€4787.69	2.25	€4568.80	4.57%
100	2.24	€2640.05	2.25	€2465.22	6.62%

The different value for W has an effect on the optimal value of T, the effect is very small because the impact of W on T is not very large. How larger the value for W, how lower the total average costs. The number of average trucks will decrease and this will result in a large reduction on the costs. We also observe the pattern that for a high value of W the cost reduction with the shifting method is bigger than for a low value of W. This means that when the truck capacity is higher, the shifting method works better. We are interested if the fill rate of the trucks is more optimal for the higher values of W. In table 18, we see the first ten time-periods of the solutions with W = 10, 24 and 100.

Table 18: Number of trucks per time-period after shifting, with different values for the truck capacity

	Time-period									
W	1	2	3	4	5	6	7	8	9	10
10	19.00	19.31	18.82	18.89	18.76	19.87	18.71	18.94	18.76	19.36
24	7.99	7.87	7.87	7.91	7.96	7.91	7.96	7.94	7.90	7.95
100	2.00	1.88	1.89	1.92	1.88	1.90	1.95	1.85	1.89	1.95

It is logical that the number of trucks is higher when the truck capacity is low. However, we also see that there are low filled trucks at the time-period 2 and 10 for $W = 10$. When we look to the values of $W = 24$, we see that this got already very high fill rates, only a few under the 90%. When we look at the numbers of $W = 100$, we see that all the fill rates are a little bit worse, this because it is harder to fill a truck when you have a lot of space that you do not need.

At last, we will show the results for different truck-space for the seven products data-set. The last two columns gives the decrease of the total average cost in percentage between respectively the first two methods and the last two methods.

Table 19: Different values for the truck-space W, basic cycle time and total average costs for the seven products data-set

	Without shifting		With Shifting		Iterative			
W	T	TRC	T	TRC	T	TRC	Δ Costs 1	Δ Costs 2
10	1.23	€6073.51	1.23	€6073.51	1.17	€5893.27	0.00%	2.97%
18	1.17	€4161.56	1.17	€4161.56	1.14	€3821.09	0.00%	8.18%
24	1.23	€3637.61	1.15	€3303.00	1.19	€3149.53	11.95%	4.65%
30	1.19	€3149.53	1.23	€2825.64	1.23	€2825.64	10.28%	0.00%
100	1.17	€2029.62	1.23	€2013.68	1.23	€2013.68	0.79%	0.00%

From the table, we can see that when the truck-space is very low, the algorithm of Goyal with shifting is not effective. However, the Iterative method finds a better solution. We also see that when the truck-space is high, the improvement by the Iterative method is low. Also, in the case of a truck-space of 100 pallets, we see that the improvement of any method is very low.

6.5 Results order heuristic

For the results of the order heuristic, we consider the main data-set. So, we have S = 500, c = 1000 and W = 24. For this heuristic, we expect more orders than with the other methods. We compare these outcomes and look to cases with lower major setup costs. Secondly, we compare the outcomes of the cases with different initial stock levels. In the first situation, the stock levels will all be zero. In our examples, we have a t^{max} equal to 365. t is in days and we want to know the ordering for a whole year. For the order size rate λ, we iterated over the interval zero till five, because the basic cycle time of the algorithm of Goyal with shifting is equal to 2,23. An order size rate of three gives the lowest total average costs, so $\lambda = 2.5$. 20 shows the total average costs of all the different methods for the main data-set, including the costs of the order heuristic. In the other columns, we can see the total average costs more detailed.

Table 20: Compare methods, total average costs of the main data-set

	TRC	Holding	Minor Setup	Major Setup	Truck
Goyal without shifting	€5296.33	€790.91	€562.16	€223.29	€3719.97
Goyal with shifting	€5150.03	€774.81	€565.93	€223.29	€3586.00
Order heuristic	€6763.32	€847.64	€744.45	€1628.77	€3542.46

We can see from the table that the total average costs are higher for the order heuristic than for the methods of the modified algorithm of Goyal. One of the big reasons is the many more orders and this increases the major setup costs as you can see in the fourth column. In a whole year, the solution of the order heuristic includes 1182 orders. This is much higher than from the other methods, as is shown in the following table.

Table 21: Compare methods, number of orders and trucks of the main data-set

	Number of orders	Average number of trucks
Goyal without shifting	163	8.33
Goyal with shifting	163	8.03
Order heuristic	1182	1.09

The other methods have only 163 orders per year. But, the number of trucks per order is much higher. For the order heuristic, most of the orders consists of one truck, it is very rare when there is more than one truck for an order. We can also see that in total, the order heuristic needs less trucks. When we multiply the number of orders with the average number of trucks per order, we have the total number of trucks for the whole time interval. This is respectively 1358, 1309 and 1288. So, there are less trucks needed with the order heuristic. Because of the major setup costs the total costs are very high. If we reduce these costs, we get the following results

Table 22: Different values for the major setup costs, with the order heuristic

S	Optimal TRC	Number of orders	Average number of trucks
500	€6763.32	1182	1.09
300	€6111.82	1182	1.09
100	€5460.31	1182	1.09
0	€5134.56	1182	1.09

So from table 22, we can see that the other variables stay constant. While the total average costs decreases more than three times the reduction of the major setup costs. We also like to compare the cases with different initial stock levels. For the past problems, we have a stock level of zero. Now, we consider higher values for the initial stock. In table 23, we see the results of these cases. The initial stock level is for all the products the same value.

Table 23: Different initial stock levels, with the order heuristic

Stock level	Total average costs	Number of orders	Average number of trucks
0	€6763.32	1182	1.09
5	€6756.76	1185	1.09
10	€6740.04	1182	1.09
50	€6885.35	1166	1.08
100	€7596.26	1112	1.10

From the table, we can see that the total average costs is going down, when we have a stock level of 5 or 10. However, when we have a initial stock level that is to high, we see that the total average costs is rising again. The number of orders is going down, this because you need less orders in the beginning because of the already many items in stock. The unnecessary high holding costs will be the reason for the increase in the total average costs. The reason why we have more orders with an initial stock of five than with an initial stock of zero could be that, the heuristic determines somewhere in the algorithm that another product will be first out of stock and this will lead to a different order time-line, which contains more orders. The heuristic is sensible for this kind of things, but big differences will not occur.

7 Conclusion

In the introduction of this paper, we define the question: Which benefits are there if we take in to account shifting time-lines and what is the effect on the optimal solution? After introducing several methods in chapter 5, we compare the solution of these methods in chapter 6 and we can conclude that there is a significant effect of the shifting time-lines on the optimal solution. First, we have shown that if we consider shifting time-lines in the modified algorithm of Goyal, the optimal solution will have many products that are not included in the first order. So, the optimal solution uses a lot of the possibility of shifting the time-lines of the products. We run the method of Goyal, with and without shifting time-lines. We compared the total average costs and showed that there is a decrease in costs and number of used trucks.

We also proposed a second method as an extension on the first one. We concluded that the solution of the algorithm of Goyal with shifting is not optimal in many cases. However, the solution is not far from a better one. We proposed an Iteration method, which searched for a solution nearby with lower total average costs. To compare the results of the three methods, we define an extra data-set. In this example we showed that the costs reduction is beneficial and that it influences the decision variables of the optimal solution.

We investigated what the influence of the different variables is on the optimal solution. We considered different values for the major setup costs, the costs per truck and the truck capacity. We saw that the improvement of the Iterative method is not very high with high values for the major setup costs. We showed that the cost per truck is very important for the total average costs and that a high truck capacity is not working for the proposed methods. We also looked to the average number of trucks and the fill rate of these trucks. We saw that the iterative method not only optimizes the fill rate of the last truck in every order, also there is an equality of the number of trucks over all the orders. Which is useful when you need to own the used trucks.

At the end, we also give an order heuristic. In this heuristic we do not have a constant basic cycle time. The computation time of the heuristic is much faster. The computation time of the order heuristic for the main data-set is less than two seconds, while the computation time of the other methods are close to two minutes. If we want to iterate over more cases in the Iterative method, we could deal with computation times of multiple days. The solution of the order heuristic includes many orders. This because the orders are most of the time one truck. The major setup costs per order results in high total average costs, however when we lower these costs, the total average costs decrease fast and the optimal solution is close to the solutions of the other methods.

At the end of our research, we like to conclude that the idea of shifting time-lines is beneficial and an interesting area which is not discussed many times in the past literature. For following research, we like to propose more investigation in a method with dynamic basic cycle times or a faster iteration method to iterate over more possible cases.

References

Arkin, E., Joneja, D., and Roundy, R. (1989). Computational complexity of uncapacitated multi-echelon production planning problems. *Operations Research Letters*, 8(2):61–66.

Goyal, S. K. (1974). Determination of optimum packaging frequency of items jointly replenished. *Management Science*, 21(4):436–443.

Goyal, S. K. and Satir, A. T. (1989). Joint replenishment inventory control: deterministic and stochastic models. *European journal of operational research*, 38(1):2–13.

Guijarro, A. M. Optimizing transportation cost with joint ordering: an application on a mexican distribution company.

Hoque, M. (2006). An optimal solution technique for the joint replenishment problem with storage and transport capacities and budget constraints. *European journal of operational research*, 175(2):1033–1042.

Jackson, P., Maxwell, W., and Muckstadt, J. (1985). The joint replenishment problem with a powers-of-two restriction. *IIE transactions*, 17(1):25–32.

Khouja, M. and Goyal, S. (2008). A review of the joint replenishment problem literature: 1989–2005. *European Journal of Operational Research*, 186(1):1–16.

Lee, F.-C. and Yao, M.-J. (2003). A global optimum search algorithm for the joint replenishment problem under power-of-two policy. *Computers & Operations Research*, 30(9):1319–1333.

Lee, L. H. and Chew, E. P. (2005). A dynamic joint replenishment policy with auto-correlated demand. *European journal of operational research*, 165(3):729–747.

Minner, S. and Silver, E. A. (2005). Multi-product batch replenishment strategies under stochastic demand and a joint capacity constraint. *IIE Transactions*, 37(5):469–479.

Ongkunaruk, P., Wahab, M., and Chen, Y. (2016). A genetic algorithm for a joint replenishment problem with resource and shipment constraints and defective items. *International Journal of Production Economics*, 175:142–152.

Oudenes, L. Joint replenishment problem of lubricant ordering with truck limitations.

Paul, S., Wahab, M., and Ongkunaruk, P. (2014). Joint replenishment with imperfect items and price discount. *Computers & Industrial Engineering*, 74:179–185.

Porras, E. and Dekker, R. (2008). A solution method for the joint replenishment problem with correction factor. *International Journal of Production Economics*, 113(2):834–851.

Salameh, M. K., Yassine, A. A., Maddah, B., and Ghaddar, L. (2014). Joint replenishment model with substitution. *Applied Mathematical Modelling*, 38(14):3662–3671.

Silver, E. A. (1976). A simple method of determining order quantities in joint replenishments under deterministic demand. *Management Science*, 22(12):1351–1361.

Van Eijs, M. (1993). A note on the joint replenishment problem under constant demand. *Journal of the Operational Research Society*, pages 185–191.

Viswanathan, S. (1996). A new optimal algorithm for the joint replenishment problem. *Journal of the Operational Research Society*, pages 936–944.

Wildeman, R., Frenk, J., and Dekker, R. (1997). An efficient optimal solution method for the joint replenishment problem. *European Journal of Operational Research*, 99(2):433–444.

8 Appendix

Table 24: Main data-set

Product j	D	h	s	Product per pallet
1	0.58	0.52	19.6	4
2	0.38	1.08	46.7	4
3	0.33	1.65	3.3	4
4	0.63	1.30	45.1	4
5	6.58	0.84	6.4	42
6	0.67	0.80	9.4	4
7	0.92	1.07	24.4	4
8	0.17	0.99	30.0	4
9	4.54	0.63	21.2	1
10	0.17	1.22	17.3	4
11	0.17	1.11	34.5	4
12	0.71	0.73	11.8	4
13	3.08	0.97	38.6	4
14	0.17	0.46	21.1	4
15	0.25	0.29	13.4	4
16	2.96	0.97	50.9	48
17	2.71	0.60	3.2	48
18	2.00	1.11	17.7	42
19	17.38	1.93	21.7	4
20	11.75	1.54	3.0	42
21	0.71	0.34	8.0	4
22	2.21	0.85	32.9	4
23	17.13	1.75	4.7	42
24	1.13	0.51	3.7	4
25	12.67	0.59	23.4	4
26	0.17	1.47	6.1	4
27	0.33	1.55	21.4	4
28	1.33	1.75	4.5	4
29	14.00	1.08	28.3	48
30	7.71	0.92	41.4	42
31	0.50	1.23	16.8	4
32	0.25	0.42	17.4	4
33	1.25	1.59	34.1	4
34	0.25	0.44	8.1	4
35	9.33	0.23	16.9	100
36	67.38	0.91	28.7	42
37	0.17	0.68	11.8	1
38	11.38	1.98	38.5	4
39	3.83	1.15	11.9	42
40	3.50	1.95	1.2	42
41	1.04	1.37	5.9	1
42	19.17	1.76	46.9	42

Table 25: Main data-set second part

Product j	D	h	s	Product per pallet
43	3.08	1.95	41.4	4
44	10.21	1.01	37.9	4
45	3.25	1.15	37.8	4
46	5.13	0.33	14.9	42
47	1.04	1.36	11.7	4
48	0.75	1.02	27.4	4
49	3.29	0.88	25.6	42
50	5.54	0.60	5.7	4
51	1.33	1.29	3.5	1
52	6.29	1.34	21.9	42
53	77.29	0.58	23.9	4
54	0.17	1.20	47.8	1
55	19.42	0.64	19.0	42
56	14.13	0.97	43.0	4
57	21.88	1.85	20.4	42
58	16.29	1.38	39.4	4
59	23.88	0.70	10.5	4
60	0.29	0.88	32.2	4
61	6.58	1.38	44.9	1
62	2.54	1.51	50.8	4
63	4.92	1.75	42.2	42
64	0.63	0.47	21.5	4
65	1.83	1.03	47.6	42
66	18.71	1.65	44.1	42
67	1.83	1.12	7.4	4
68	12.42	1.23	16.0	42
69	1.46	0.78	47.1	4
70	6.63	1.43	18.4	42
71	3.21	1.41	44.3	4
72	6.29	1.36	39.1	42
73	3.46	1.67	35.3	4
74	4.42	1.03	33.1	42
75	0.79	0.64	4.2	4
76	0.42	1.90	28.4	4
77	0.50	0.44	1.6	4
78	5.83	0.75	41.1	4
79	1.67	0.67	12.2	4
80	20.67	1.76	2.0	4
81	1.75	0.84	35.7	42
82	8.08	1.46	14.8	42
83	0.67	1.76	44.2	9

Table 26: Main data-set begin shifting time-lines

S = 500, c = 1000		Time-period							
Product j	k_j	1	2	3	4	5	6	7	8
1	4	0	0	1	0	0	0	1	0
2	5	1	0	0	0	0	1	0	0
3	1	1	1	1	1	1	1	1	1
4	4	0	1	0	0	0	1	0	0
5	1	1	1	1	1	1	1	1	1
6	2	0	1	0	1	0	1	0	1
7	3	1	0	0	1	0	0	1	0
8	7	1	0	0	0	0	0	0	1
9	1	1	1	1	1	1	1	1	1
10	5	0	0	0	1	0	0	0	0
11	7	0	0	0	1	0	0	0	0
12	2	1	0	1	0	1	0	1	0
13	2	0	1	0	1	0	1	0	1
14	8	0	0	0	1	0	0	0	0
15	7	0	0	0	0	0	0	1	0
16	2	0	1	0	1	0	1	0	1
17	1	1	1	1	1	1	1	1	1
18	2	1	0	1	0	1	0	1	0
19	1	1	1	1	1	1	1	1	1
20	1	1	1	1	1	1	1	1	1
21	3	0	1	0	0	1	0	0	1
22	2	1	0	1	0	1	0	1	0
23	1	1	1	1	1	1	1	1	1
24	1	1	1	1	1	1	1	1	1
25	1	1	1	1	1	1	1	1	1
26	3	0	1	0	0	1	0	0	1
27	3	1	0	0	1	0	0	1	0
28	1	1	1	1	1	1	1	1	1
29	1	1	1	1	1	1	1	1	1
30	1	1	1	1	1	1	1	1	1
31	3	0	0	1	0	0	1	0	0
32	8	0	0	0	1	0	0	0	0
33	2	0	1	0	1	0	1	0	1
34	4	0	1	0	0	0	1	0	0
35	1	1	1	1	1	1	1	1	1
36	1	1	1	1	1	1	1	1	1
37	5	0	0	0	0	1	0	0	0
38	1	1	1	1	1	1	1	1	1
39	1	1	1	1	1	1	1	1	1
40	1	1	1	1	1	1	1	1	1
41	1	1	1	1	1	1	1	1	1
42	1	1	1	1	1	1	1	1	1

Table 27: Main data-set begin shifting time-lines, second part

S = 500, c = 1000		Time-period							
Product j	k_j	1	2	3	4	5	6	7	8
43	1	1	1	1	1	1	1	1	1
44	1	1	1	1	1	1	1	1	1
45	2	1	0	1	0	1	0	1	0
46	2	0	1	0	1	0	1	0	1
47	2	0	1	0	1	0	1	0	1
48	3	0	1	0	0	1	0	0	1
49	2	1	0	1	0	1	0	1	0
50	1	1	1	1	1	1	1	1	1
51	1	1	1	1	1	1	1	1	1
52	1	1	1	1	1	1	1	1	1
53	1	1	1	1	1	1	1	1	1
54	8	0	0	0	0	0	0	0	1
55	1	1	1	1	1	1	1	1	1
56	1	1	1	1	1	1	1	1	1
57	1	1	1	1	1	1	1	1	1
58	1	1	1	1	1	1	1	1	1
59	1	1	1	1	1	1	1	1	1
60	6	1	0	0	0	0	0	1	0
61	1	1	1	1	1	1	1	1	1
62	2	1	0	1	0	1	0	1	0
63	1	1	1	1	1	1	1	1	1
64	4	1	0	0	0	1	0	0	0
65	3	0	1	0	0	1	0	0	1
66	1	1	1	1	1	1	1	1	1
67	1	1	1	1	1	1	1	1	1
68	1	1	1	1	1	1	1	1	1
69	3	0	0	1	0	0	1	0	0
70	1	1	1	1	1	1	1	1	1
71	2	1	0	1	0	1	0	1	0
72	1	1	1	1	1	1	1	1	1
73	1	1	1	1	1	1	1	1	1
74	1	1	1	1	1	1	1	1	1
75	2	0	1	0	1	0	1	0	1
76	3	1	0	0	1	0	0	1	0
77	1	1	1	1	1	1	1	1	1
78	2	0	1	0	1	0	1	0	1
79	2	1	0	1	0	1	0	1	0
80	1	1	1	1	1	1	1	1	1
81	3	0	1	0	0	1	0	0	1
82	1	1	1	1	1	1	1	1	1
83	3	1	0	0	1	0	0	1	0