**Determining the optimal number of sublots in a single-product, deterministic
flow shop with overlapping operations**

Inneke Van Nieuwenhuyse^{1*}

Faculty of Applied Economics UFSIA-RUCA, Antwerp, Belgium

Nico Vandaele^{**}

Faculty of Applied Economics UFSIA-RUCA, Antwerp, Belgium

Abstract: This paper describes a model for minimizing total costs in a single-product, deterministic flow shop with overlapping operations in terms of the sublot size used. Three types of costs are considered: the inventory holding costs, the transportation costs and the so-called “gap costs” which may result from the intermittent idling of machines between consecutive sublots. We provide explicit expressions for the gaps in terms of the system parameters, and discuss the impact of the transfer batch sizing policy on the average transfer batch lead time and the total gap time in the system. The proposed cost model is believed to be particularly relevant for manufacturing cell environments.

Keywords: Lot splitting, Transfer batching, Overlapping operations, Lead time management

---

¹ Inneke Van Nieuwenhuyse, UFSIA, Prinsstraat 13, 2000 Antwerp, Belgium
Tel.: +32/3/220.40.44, Fax: +32/3/220.47.99, inneke.vannieuwenhuyse@ua.ac.be

* Inneke Van Nieuwenhuyse is a Research Assistant of the Fund for Scientific Research-Flanders (Belgium)

** This research is supported by project n° G.o3.60 of the Fund for Scientific Research-Flanders (Belgium) ('The design of a stochastic Advanced Resources Planning (ARP) system')

1. Introduction

Today, it is commonly known that batching decisions have an important impact on the performance of a production system. By setting batch sizes, one can influence performance measures such as cycle times and work-in-process levels. As these are decisive factors for the responsiveness of any production environment, setting batch sizes in a production system is an important control ([13], [7]).

In studying the impact of batching decisions, one should make a distinction between two types of batches: *process batches* and *transfer batches*. A process batch (also referred to as a *production batch*) is defined as the quantity of a product processed on a machine without interruption by other items, whereas a transfer batch is the size of a subplot moved after production on one machine to another operation or machine ([9]).

In the literature, it is widely accepted that the use of transfer batch sizes smaller than the process batch size can reduce product lead times. This is due to the mechanism of *overlapping operations* (e.g. [3], [5], [6], [8], [14]): by allowing transportation of partial batches to a downstream station, this station can already start processing these partial batches while work proceeds at the upstream station, thereby accelerating the progress of work through the production facility.

In the literature, the process of splitting a process batch into sublots and scheduling those sublots in overlapping fashion is often referred to as “*lot streaming*” ([18]) or “*lot splitting*” ([16], [12], [17]). Previous research has primarily been aimed at finding the optimal lot streaming procedure in order to minimize either makespan or mean flow time in a deterministic flow shop environment. The different variations of lot streaming models reflect the constraints imposed on the environment that one wishes to model. Firstly, it may be assumed that a machine may not remain idle between consecutive sublots. This is referred to as the “no-idling assumption” (e.g.

[18], [1], [15]). Secondly, it may be assumed that all sublots have to be of equal size (e.g. [8]). Thirdly, it may be imposed that sublots have to be consistent throughout the system, implying that the same subplot sizes have to be used at each machine (e.g. [12] and [2]). More recently, the issue of lot splitting has also received attention within the framework of supply chain management, as a practice that is consistent with the JIT policy of frequent deliveries (e.g. [4] and [10]).

This paper focuses on a deterministic flow shop environment, in which intermittent idling is allowed and in which sublots are assumed to be equal and consistent. The model proposed is flexible, in the sense that it allows to model delay servers (such as transportation resources) and setup times explicitly. The findings described in this paper extend the existing research results in that we do not focus on minimizing makespan or average flow time, but on minimizing total costs. The objective function consists of the following three cost components: the inventory holding costs, the transportation costs and the so-called “gap costs”. As will be shown, a tradeoff exists between these three components. On the one hand, it is known from Little’s law ([11]) that the average work-in-process inventory is related to the average lead time in the system. Hence, as lot splitting leads to lower average lead times, we can expect smaller transfer batch sizes to lead to a decrease in the average inventory holding costs. On the other hand however, it is obvious that smaller transfer batch sizes will lead to an increase in transportation costs. Moreover, the use of lot splitting may cause a machine to remain idle between two consecutive transfer batches belonging to the same job, due to the fact that its own setup and processing times and those of the preceding machine are not synchronized. These idle times cause the makespan of a process batch on a server to increase, and will be referred to as *gaps*. They may represent a cost for the system because during gap times the server has to be kept

“operational”, i.e. ready for processing the next transfer batch when it arrives; depending upon the type of server, this may entail labour and/or energy costs. In order to incorporate these gap costs explicitly in the objective function, we provide explicit expressions for the gap times in terms of the system parameters and the transfer batch size used.

The remainder of this paper will be organized as follows: in section 2, we give an overview of the assumptions and the notation used to describe the system. In section 3, expressions are derived for the gaps on the different machines and the lead time of transfer batches through the system. In sections 4 and 5, we further explore the impact of the transfer batch sizing policy (i.e., the number of sublots) on the average transfer batch lead time and the total gap time in the system. The results are illustrated by means of an example. Section 6 describes the resulting cost optimization model, and finally a last section summarizes the most important conclusions.

2. Assumptions and notation

It is assumed that the flow shop consists of M servers. In our setting, a “server” may refer to a machine on which operations are performed, or to a transportation resource (forklift truck, conveyor, etc.) needed to move products between the different machines. The production line produces only one product type, which is processed on the consecutive machines in *process batches* of size N .

It is assumed that products arrive in process batches of size N in front of the first machine of the production line. The interarrival times of process batches are deterministic, as well as the setup times and the processing times on the different servers. After processing on a server, products are collected to form a *transfer batch*

of size L ($L\beta N$). For simplicity as well as practical reasons, L is supposed to be a common divisor of N , such that a single process batch is split into an integer number (T) of transfer batches ($T = N/L$). As soon as a transfer batch is complete, it is moved to the next server. The transfer batch size used between the consecutive servers is presumed to be the same throughout the product's routing. At the end of the routing, the transfer batches belonging to the same process batch are regrouped for shipment.

Each server m may require a setup time $S_m > 0$ ($m=1, \dots, M$) to be performed at the start of every process batch. For example, it is possible that the production line only produces one product type but in different colors or sizes, such that at the start of a new process batch a setup time is necessary to change the paint or the fixtures. It is assumed that this setup time can not start before the first transfer batch of the involved process batch is present in the input buffer. We say that the first transfer batch acts as a “flag”²: its arrival in front of a server authorizes the start of the setup, thereby causing the downstream operation to overlap partly with the upstream operation.

The processing time for a transfer batch on server m is denoted by X_m ($m=1, \dots, M$). X_m may be dependent on the transfer batch size; in that case, m is referred to as a capacity server and X_m can be expressed as the transfer batch size (L) times the unit processing time on server m ($X_{u,m}$). Alternatively, X_m may be a constant independent of the transfer batch size; in that case, m is referred to as a delay server.

Transportation resources (forklift trucks, conveyors etc.) are typically delay servers.

For ease of notation, the set of capacity servers will hereafter be denoted by M_C , the

---

² The term “flag” is taken from a paper by Smunt et al. [17]. In their setting, the first transfer batch has the smallest possible nonzero subplot size, while all remaining transfer batches have equal size. In our setting, all transfer batches (including the flag) have the same size. However, the purpose of the flag is the same, i.e. to signal the impending arrival of subsequent transfer batches to the machine and to authorize a setup.

set of delay servers by M_D . The set of transportation servers will be denoted by M_{trans} ($M_{trans} \subset M_D$).

The empty travel time that may be required by a transportation resource m (e.g. forklift trucks needing to travel back and forth between the machines) is denoted by B_m , (≥ 0), and is assumed to occur immediately after the resource has delivered a transfer batch to the next downstream server.

3. Lead time model and gap calculations

In what follows, we will derive expressions for the average transfer batch lead time and the gaps which occur between consecutive transfer batches on an arbitrary server. These expressions will then be used in the next section to calculate the inventory costs and the idle time costs.

Given that the system is completely deterministic and that we only have one product type, we know that the flow pattern of process batches through the system will be cyclic: it will be the same for each process batch, and the pattern is simply repeated each time a new process batch arrives in front of the first machine in the line.

Moreover, in order for the system to be stable, the interarrival time of process batches in front of the first machine must be larger than the service time needed for the entire process batch (including the setup time) on the bottleneck server. Under this assumption, there will be no competition for server capacity between consecutive process batches: each server will have finished processing the former process batch by the time the first transfer batch of the next process batch arrives in its input buffer. Consequently, the setup time on a server always starts as soon as the first transfer batch of a particular process batch has arrived.

In order to facilitate the analysis, we use the following example. Suppose that we have a system consisting of three machines and two forklift trucks used for transportation between the machines. The process batch size of the product equals 12 product units ($N=12$), and each process batch is split up into 4 transfer batches ($T=4$). Consequently, the transfer batch size equals 3 product units ($L=3$). The setup times (S_m), total processing times per transfer batch (X_m) and empty travel times (B_m), expressed in hours, are given in Table 1. In this example, it is assumed that the processing time for a transfer batch on M1, M2 and M3 is dependent upon the transfer batch size L and that there is a setup required on each of these machines. For the two forklift trucks (F1 and F2), there is no setup time required. The processing time represents the time needed to move a transfer batch between the machines and is independent of the transfer batch size. The empty travel time is set equal to the processing time. It is assumed that the interarrival time of process batches entering the system equals 78 hours.

Figure 1 shows the flowchart for the production line³. The black time blocks on the different machines refer to the setup times, the processing times for the transfer batches are indicated in white and the empty travel times in gray. The gaps occurring on the servers are indicated by the dashed lines; the corresponding number refers to the duration of the gap. As indicated by the vertical arrows, each transfer batch is moved to the next server as soon as it has finished processing. From the figure, it can also be seen that a forklift is assumed to travel back to the upstream machine as soon as it has delivered a transfer batch at the downstream machine.

---

³ Figure 1 shows the flow of only one single process batch through the system. However, as mentioned before, it suffices to study a single production cycle of one process batch in order to develop the lead time model.

As the first transfer batch of a process batch never waits in front of a particular server, the average transfer batch lead time $E(W)$ of an arbitrary transfer batch t ($t=1, \dots, T$) through the system can be written as:

$$E(W) = \sum_{m=1}^M S_m + \sum_{m=1}^{M-1} X_m + T * X_M + (T-1) * B_M + G(M, T) \quad (1)$$

The first two terms represent the time spent before the first transfer batch starts processing on the final server. In expression (1), $G(M, T)$ is defined as the total amount of idle time (or total "gap") occurring before the last transfer batch (T) on the last server (M). In general, $G(m, T)$ is defined as the total gap occurring before transfer batch T on an arbitrary server m .

It is clear from expression (1) that, in order to calculate $E(W)$, we have to derive an expression for $G(M, T)$ and, more in general, for $G(m, T)$. As the occurrence of a gap is a binary event, we introduce the 0/1 variable $\gamma(m, T)$ for $m \geq 2$:

$$\begin{aligned} \gamma(m, T) &= 1 && \text{if } G(m-1, T) + (T-1) * (X_{m-1} + B_{m-1} - X_m - B_m) - S_m > 0 \\ &= 0 && \text{otherwise} \end{aligned} \quad (2)$$

If $\gamma(m, T) = 1$, this indicates that a gap occurs before transfer batch T on server m .

Note that, in our setting, $\gamma(m, T)$ only needs to be defined for $m \geq 2$. On the first server, $\gamma(1, T) = 0$ since we have assumed that products arrive in process batches in front of the line. Note also that expression (2) is only valid because we have assumed that the entire system is deterministic, implying that the first transfer batch of each process batch never has to wait in front of a server until it becomes idle.

For ease of notation, we introduce the variable $V(m, T)$ for $m \geq 2$:

$$V(m, T) = (T-1) * (X_{m-1} + B_{m-1} - X_m - B_m) - S_m \quad (3)$$

such that we can rewrite (1) for $m \geq 2$ as:

$$\gamma(m, T) = 1 \quad \text{if } G(m-1, T) + V(m, T) > 0 \quad (4)$$

$$= 0 \quad \text{otherwise}$$

It is clear from expressions (2) and (4) that there is a positive relationship between $G(m, T)$ and $G(m-1, T)$: the larger the total gap on a server m , the higher the probability that a gap will also occur on the subsequent server.

By means of expressions (3) and (4), $G(m, T)$ (for any server $m \geq 2$) can be expressed as:

$$\begin{aligned} G(m, T) &= \max [0, V(m, T) + G(m-1, T)] \\ &= \gamma(m, T) * [V(m, T) + G(m-1, T)] \end{aligned} \quad (5)$$

Solving this expression recursively finally leads to the following expression:

$$G(m, T) = \sum_{i=2}^m \left[\prod_{j=i}^m \gamma(j, T) \right] * V(i, T) \quad (6)$$

such that the last term in expression (1) equals:

$$G(M, T) = \sum_{i=2}^M \left[\prod_{j=i}^M \gamma(j, T) \right] * V(i, T) \quad (7)$$

Note that $G(M, T)$ is only strictly larger than 0 if there exists a server $i \geq 2$ such that, for this server i and all downstream servers j up to server M , the value of $\gamma(j, T)$ is equal to 1. Stated more generally (in accordance with expression (6)), this implies that a non-zero gap time on an upstream server u will only influence the gap time on an arbitrary downstream server d if, for all servers m between u and d , the gap time is non-zero.

By means of expressions (1) and (7), we can now write the average transfer batch lead time $E(W)$ as follows:

$$E(W) = \sum_{m=1}^M S_m + \sum_{m=1}^{M-1} X_m + T * X_M + (T-1) * B_M + \sum_{i=2}^M \left[\prod_{j=i}^M \gamma(j, T) \right] * V(i, T) \quad (8)$$

For the example given in Figure 1, $E(W)$ equals 51 hours. As $E(W)$ is a function of the number of transfer batches T , we know from Little's law that the average work-in-process in the system (and, hence, the inventory holding costs) will also be dependent on T . Consequently, it is important to gain insight in the relationship between $E(W)$ and T . In section 4, this relationship will be further explored.

Based upon expression (5), we can write the total amount of gap time G_{tot} in the system as:

$$G_{tot} = \sum_{m=2}^M G(m, T) = \sum_{m=2}^M \max[0, G(m-1, T) + V(m, T)] \quad (9)$$

where T , as mentioned before, represents the last transfer batch of the process batch.

For the example given in Figure 1, the total amount of gap time equals 50 hours.

From expression (9) we can conclude that G_{tot} will also vary with the transfer batch sizing policy, and hence will have an impact on the gap costs for the system. The relationship between G_{tot} and T is examined in section 5.

4. Relationship between the lot splitting policy and the average transfer batch lead time

To explore this relationship, it is useful to rewrite expression (8) as follows:

$$E(W) = \sum_{m \in M_C} \frac{N}{T} * X_{u,m} + \sum_{m \in M_D} X_m + \max \left[\max_{i \in M_D} \left[\sum_{m=1}^i S_m + (T-1) * (X_i + B_i) \right], \max_{j \in M_C} \left[\sum_{m=1}^j S_m + (T-1) * \frac{N}{T} * X_{u,j} \right] \right] \quad (10)$$

The proof of this expression is given in Appendix A. Expression (10) offers several useful insights. First of all, we can derive that in systems that only consist of delay servers, the average transfer batch lead time will be equal to:

$$E(W) = \sum_{m \in M_D} X_m + \max_{i \in M_D} \left[\sum_{m=1}^i S_m + (T-1) * (X_i + B_i) \right] \quad (11)$$

which is linearly increasing in the number of transfer batches T . Hence, in systems that only consist of delay servers, the average transfer batch lead time is minimized at $T=I$, i.e. when no lot splitting is used.

In systems that only consist of capacity servers, the average transfer batch lead time is given by:

$$E(W) = \sum_{m \in M_C} \frac{N}{T} * X_{u,m} + \max_{j \in M_C} \left[\sum_{m=1}^j S_m + (T-1) * \frac{N}{T} * X_{u,j} \right] \quad (12)$$

which is clearly convex and continuously decreasing in T . Hence, in systems that only consist of capacity servers, $E(W)$ is minimized when $T=N$, i.e. when the process batch is split in the largest possible number of transfer batches. The continuous decrease in $E(W)$ in terms of T is caused by the overlapping operations mechanism, which leads to a smoother workflow when a larger number of sublots (or, equivalently, a smaller subplot size) is used for a given process batch size. Hence, in the remainder of this paper, it will be referred to as the *overlapping operations effect*.

In general, for systems with both capacity and delay servers, we can rewrite expression (10) as follows:

$$E(W) = \max[f_1, f_2] \quad (13)$$

with

$$f_1 = \sum_{m \in M_C} \frac{N}{T} * X_{u,m} + \sum_{m \in M_D} X_m + \max_{i \in M_D} \left[\sum_{m=1}^i S_m + (T-1) * (X_i + B_i) \right]$$

$$f_2 = \sum_{m \in M_C} \frac{N}{T} * X_{u,m} + \sum_{m \in M_D} X_m + \max_{j \in M_C} \left[\sum_{m=1}^j S_m + (T-1) * \frac{N}{T} * X_{u,j} \right]$$

As both f_1 and f_2 are convex in T , and the pointwise maximum of two convex functions is also a convex function, we can conclude from expression (13) that $E(W)$ is convex in T . However, it may no longer be continuously decreasing in T . To explain this, we will treat $E(W)$ as a continuous function of T . Suppose that the transfer batching policy is such that $\max[f_1, f_2] = f_2$. This means that, given that policy, there is a server $r^*(M, T) \in M_C$ for which:

$$\begin{aligned} \sum_{m=1}^{r^*(M, T)} S_m + (T-1) * \frac{N}{T} * X_{u, r^*(M, T)} &= \max_{\substack{j \in M_C \\ j \leq M}} \left[\sum_{m=1}^j S_m + (T-1) * \frac{N}{T} * X_{u, j} \right] \\ &\geq \max_{\substack{i \in M_D \\ i \leq M}} \left[\sum_{m=1}^i S_m + (T-1) * (X_i + B_i) \right] \end{aligned} \quad (14)$$

We say that $r^*(M, T)$ is the *dominating server* for M at policy T . The average transfer batch lead time is then given by:

$$\begin{aligned} E(W) &= \sum_{m \in M_C} \frac{N}{T} * X_{u, m} + \sum_{m \in M_D} X_m + \sum_{m=1}^{r^*(M, T)} S_m + (T-1) * \frac{N}{T} * X_{u, r^*(M, T)} \\ &= \sum_{\substack{m \in M_C \\ m \neq r^*(M, T)}} \frac{N}{T} * X_{u, m} + \sum_{m \in M_D} X_m + \sum_{m=1}^{r^*(M, T)} S_m + N * X_{u, r^*(M, T)} \end{aligned}$$

and the first derivative with respect to T is given by:

$$\frac{\partial E(W)}{\partial T} = \frac{-N}{T^2} \sum_{\substack{m \in M_C \\ m \neq r^*(M, T)}} X_{u, m} < 0$$

Hence, we can conclude that, as long as the transfer batching policy is such that server M is dominated by a capacity server, $E(W)$ will be decreasing in T . The decrease is again caused by the mechanism of overlapping operations, so we refer to this decrease as the *overlapping operations effect*.

Suppose now that the transfer batching policy is such that $\max[f_1, f_2] = f_1$. This means that server M is now dominated by a delay server $s^*(M, T)$ for which:

$$\begin{aligned}
\sum_{m=1}^{s^*(M,T)} S_m + (T-1) * (X_{s^*(M,T)} + B_{s^*(M,T)}) &= \max_{\substack{i \in M_D \\ i \leq M}} \left[\sum_{m=1}^i S_m + (T-1) * (X_i + B_i) \right] \\
&\geq \max_{\substack{j \in M_C \\ j \leq M}} \left[\sum_{m=1}^j S_m + (T-1) * \frac{N}{T} * (X_{u,j}) \right]
\end{aligned} \tag{15}$$

Note that, in practice, this implies that $s^*(M,T)$ is a bottleneck for all servers between $s^*(M,T)$ and M . Indeed, it follows from the last expression that

$$\begin{aligned}
\frac{S_{s^*(M,T)}}{T-1} + X_{s^*(M,T)} + B_{s^*(M,T)} &> \frac{N}{T} * X_{u,j} + \frac{S_j}{T-1} \quad \forall j \in M_C, j > s^*(M,T) \\
\frac{S_{s^*(M,T)}}{T-1} + X_{s^*(M,T)} + B_{s^*(M,T)} &> X_i + B_i + \frac{S_i}{T-1} \quad \forall i \in M_D, i > s^*(M,T)
\end{aligned}$$

The average transfer batch lead time is then given by:

$$E(W) = \sum_{m \in M_C} \frac{N}{T} * X_{u,m} + \sum_{m \in M_D} X_m + \sum_{m=1}^{s^*(M,T)} S_m + (T-1) * (X_{s^*(M,T)} + B_{s^*(M,T)})$$

and the first derivative is:

$$\frac{\partial E(W)}{\partial T} = \frac{-N}{T^2} \sum_{m \in M_C} X_{u,m} + X_{s^*(M,T)} + B_{s^*(M,T)}$$

Hence, $E(W)$ will be increasing in T when

$$X_{s^*(M,T)} + B_{s^*(M,T)} > \frac{N}{T^2} \sum_{m \in M_C} X_{u,m} \tag{16}$$

This is an intuitively appealing result. The left-hand side of expression (16) represents the increase in lead time on the bottleneck server, caused by an increase in T . The right-hand side of the inequality shows the total reduction in lead time on all capacity servers, caused by that increase in T . Hence, when the lead time increase on the bottleneck server is larger than the lead time decrease on all capacity servers, the average transfer batch lead time will increase. From (16), we can derive that this effect is most likely to occur as a result of excessive lot splitting, i.e. when T is high. As the increase in $E(W)$ can only occur when a delay server becomes the bottleneck,

we refer to this phenomenon as the *delay effect*. Note however that (16) is a more stringent condition than (15): hence, the fact that a delay server becomes the bottleneck will not necessarily cause the delay effect to occur.

We will now illustrate these findings by means of the example given in section 3. We use the same input data as in Table 1, but let the number of transfer batches vary over all possible values. Figure 2 (a) shows the resulting average lead time $E(W)$ in terms of the number of transfer batches T . As N is equal to 12 units and T has to be a common divisor of N , it can only take on the values 1,2,3,4,6 or 12. Figure 2 (b) shows $E(W)$ in terms of the transfer batch size L .

From Figure 2 (a), we see that $E(W)$ is indeed convex in T . As shown, the convex nature of the curve is the result of the *overlapping effect* and the *delay effect*. At $T=1$, server 5 is the dominating server, and at $T=2,3$ or 4 server 1 is the dominating server. These are capacity servers, so the overlapping operations effect plays until $T=4$. However, at $T=6$ and $T=12$, forklift 2 becomes the dominating server. Moreover, it can be shown that at $T=6$ and $T=12$, condition (16) is satisfied: the increase in lead time on forklift 2 more than compensates the lead time decreases obtained on all capacity servers combined. Hence, the delay effect comes into play at $T=6$ and $T=12$, and the average transfer batch lead time increases.

From Figure 2 (b), it can be seen that there is an interesting parallel between the overlapping and delay effect for transfer batch sizes, and the saturation and batching effect as it is already known in the literature for process batch sizes ([9], [13]). The delay effect can be seen as similar to the saturation effect, as it is caused by an increase in utilization of the delay server. This obstructs the flow of work through the system, thereby leading to longer delays over this segment of the product's routing.

The overlapping effect can be compared to the inverse of the batching effect: splitting a process batch in a larger number of transfer batches leads to a decrease in wait-for-batch times behind the machines, thereby leading to a smoother workflow.

Note that, in systems with delay servers, the range of feasible transfer batch sizes is bounded below by a *critical transfer batch size* L_c that may be strictly larger than 1 product unit. Indeed, in order for the system to be stable, the following condition has to be satisfied:

$$\lambda_{PB} < \frac{1}{\underbrace{\max_{m \in M_D} [S_m + T^*(X_m + B_m)]}_{S_{\max} + T^*(X_{\max} + B_{\max})}}$$

in which λ_{PB} equals the arrival rate of process batches entering the system (in our example, λ_{PB} equals 1/78). In other words, the lot splitting policy has to be such that the utilization of the slowest delay server stays below 100%. The critical transfer batch size can then be calculated as follows:

$$L_c = \left\lceil \frac{N^*(X_{\max} + B_{\max})}{\frac{1}{\lambda_{PB}} - S_{\max}} \right\rceil$$

where $\lceil x \rceil$ refers to the smallest integer larger than or equal to x . In our example, L_c is equal to 1 product unit.

It is important to note that the delay effect can only occur when, due to excessive lot splitting, a delay station becomes the dominating server in the system. If, for example, forklift 2 would have the same processing and empty travel time as forklift 1, the effect would not occur: machine 1 would be the dominating server at all values of T . The resulting curve for $E(W)$ is shown in Figure 3: the average transfer batch lead time is now convex and continuously decreasing in the number of transfer batches.

5. Relationship between the lot splitting policy and the total gap time in the system

As mentioned before, the use of lot splitting and overlapping operations may cause a server to remain idle between consecutive transfer batches. These “gaps” may represent a cost for the system, as during these times resources (such as the machine, the operator,...) need to be available without being productive. Expression (9) already showed that G_{tot} is dependent on the transfer batch sizing policy. In this section, the relationship between T and G_{tot} is further explored.

In Appendix B, it is proven that the expression for $G(m, T)$ can be written as follows:

$$G(m, T) = \max \left[\max_{\substack{i \in M_C \\ i \leq m}} \left[\sum_{k=1}^i S_k + (N - \frac{N}{T}) * X_{u,i} \right], \max_{\substack{j \in M_D \\ j \leq m}} \left[\sum_{k=1}^j S_k + (T - 1) * (X_j + B_j) \right] \right] - \sum_{j=1}^m S_j \quad (17)$$

$$\begin{cases} -(T - 1) * (X_m + B_m) & \text{if } m \in M_D \\ -(N - \frac{N}{T}) * X_{u,m} & \text{if } m \in M_C \end{cases}$$

Hence, the total gap time in the system can be written as:

$$G_{tot} = \sum_{m \in M_D} G(m, T) + \sum_{m \in M_C} G(m, T)$$

$$= \underbrace{\sum_{m \in M_D} \left[\max \left[\max_{\substack{i \in M_C \\ i \leq m}} \left[\sum_{k=1}^i S_k + (N - \frac{N}{T}) * X_{u,i} \right], \max_{\substack{j \in M_D \\ j \leq m}} \left[\sum_{k=1}^j S_k + (T - 1) * (X_j + B_j) \right] \right] - \sum_{j=1}^m S_j - (T - 1) * (X_m + B_m) \right]}_{\text{total gap on the delay servers}}$$

$$+ \underbrace{\sum_{m \in M_C} \left[\max \left[\max_{\substack{i \in M_C \\ i \leq m}} \left[\sum_{k=1}^i S_k + (N - \frac{N}{T}) * X_{u,i} \right], \max_{\substack{j \in M_D \\ j \leq m}} \left[\sum_{k=1}^j S_k + (T - 1) * (X_j + B_j) \right] \right] - \sum_{j=1}^m S_j - (N - \frac{N}{T}) * (X_{u,m}) \right]}_{\text{total gap on the capacity servers}} \quad (18)$$

We will now separately discuss the total gap on the delay servers and the total gap on the capacity servers, in terms of the transfer batching policy T . For reasons of clarity, we will again assume that both gap functions are continuous in T .

Given a particular transfer batch policy T' , the first and second derivatives of the total gap on the *capacity servers* can be calculated as:

$$\begin{aligned} & \left. \frac{\partial}{\partial T} \sum_{m \in M_C} G(m, T) \right|_{T=T'} \\ &= \sum_{m \in M_C} -\frac{N}{T'^2} * (X_{u,m}) + \sum_{\substack{m \in M_C: \\ r^*(m, T') \in M_C}} \frac{N}{T'^2} * X_{u, r^*(m, T')} \\ &+ \sum_{\substack{m \in M_C: \\ r^*(m, T') \in M_D}} (X_{r^*(m, T')} + B_{r^*(m, T')}) \end{aligned} \quad (19)$$

$$\begin{aligned} & \left. \frac{\partial^2}{\partial T^2} \sum_{m \in M_C} G(m, T) \right|_{T=T'} \\ &= 2 \frac{N}{T'^3} * \left[\sum_{m \in M_C} X_{u,m} - \sum_{\substack{m \in M_C: \\ r^*(m, T') \in M_C}} X_{u, r^*(m, T')} \right] \end{aligned} \quad (20)$$

In these expressions, $r^*(m, T')$ denotes the server that is dominating server m at policy T' . Given the definition of $r^*(m, T')$ (cfr. expressions (14) and (15)), it is clear that the first derivative in expression (19) will always be positive. Hence, the total gap on the capacity servers will continuously increase in T . The second derivative can be positive or negative:

$$\begin{aligned} & \left. \frac{\partial^2}{\partial T^2} \sum_{m \in M_C} G(m, T) \right|_{T=T'} > 0 \Leftrightarrow \sum_{m \in M_C} X_{u,m} > \sum_{\substack{m \in M_C: \\ r^*(m, T') \in M_C}} X_{u, r^*(m, T')} \\ & \left. \frac{\partial^2}{\partial T^2} \sum_{m \in M_C} G(m, T) \right|_{T=T'} < 0 \Leftrightarrow \sum_{m \in M_C} X_{u,m} < \sum_{\substack{m \in M_C: \\ r^*(m, T') \in M_C}} X_{u, r^*(m, T')} \end{aligned} \quad (21)$$

In systems that only consist of capacity servers, the second condition will always be satisfied. Hence, we can conclude that in such systems, the total gap time in the system will be an increasing and concave function of T .

In systems that contain a mix of capacity and delay servers, it may happen that the first condition is satisfied at high values of T . For example, it is intuitively clear that if T is such that all capacity servers are dominated by delay servers, the right-hand side of the first condition reduces to 0 and the second derivative becomes positive. We can conclude that, if this condition occurs, the total gap on the capacity servers will be increasing and convex in T .

The first derivative of the total gap on the *delay servers*, given a particular transfer batching policy T' , is given by:

$$\begin{aligned}
& \left. \frac{\partial}{\partial T} \sum_{m \in M_D} G(m, T) \right|_{T=T'} \\
&= \underbrace{\sum_{\substack{m \in M_D: \\ s^*(m, T') \in M_D}} \left[-(X_m + B_m) + X_{s^*(m, T')} + B_{s^*(m, T')} \right]}_{\geq 0} \\
&+ \sum_{\substack{m \in M_D: \\ s^*(m, T') \in M_D}} \left[-(X_m + B_m) + \frac{N}{T'^2} (X_{u, r^*(m, T')}) \right]
\end{aligned} \tag{22}$$

The first term in this derivative represents the change in the total gap time, caused by an increase in T , on the delay servers that are dominated by a delay server. Given the definition of $s^*(m, T')$, this part of the derivative is always larger than or equal to zero. The second term represents the change in the total gap time, caused by an increase in T , on the delay servers that are dominated by a capacity server. This term is decreasing in T , and can be positive (which is most probable at low values of T) or negative (at higher values of T). A negative value indicates that the total gap time on the delay servers, dominated by a capacity server, is *decreasing* in terms of T . From expression (18) and the previous discussion, we can conclude that this is the only situation in which a component of G_{tot} can decrease with increasing values of T : it is caused by the fact that, as the number of sublots grows, the delay servers dominated

by a capacity server will increasingly be in processing or empty travel condition (traveling back and forth between the machines), while the makespan of a process batch on these servers does not significantly increase. Hence, the gap times on these servers will diminish until, finally, they reach zero. This effect will hereafter be referred to as the *gap compensation effect*.

Note that, at high values of T , the gap compensation effect is likely to disappear: when the process batch is split up into a large number of sublots, a delay server will most often be dominated by another delay server or by itself. In that case, the second term in expression (22) becomes less negative while the first term either stays equal or increases; this may cause the first derivative to become positive again.

The second derivative of the total gap on the delay servers, given a particular transfer batching policy T' , is given by:

$$\begin{aligned} & \left. \frac{\partial^2}{\partial T^2} \sum_{m \in M_D} G(m, T) \right|_{T=T'} \\ &= \sum_{\substack{m \in M_D: \\ r^*(m, T') \in M_C}} \left[-2 * \frac{N}{T'^3} (X_{u, r^*(m, T')}) \right] \end{aligned}$$

As long as the transfer batch sizing policy is such that one or more delay servers are dominated by a capacity server, the total gap on the delay servers will be concave in T . However, if T is such that all delay servers are dominated by a delay server, which is most likely to happen at high values of T , the total gap on the delay servers will be a linear function of T .

We will now illustrate these findings by means of the example. Figure 4 shows the relationship between T and G_{tot} for the example given in section 3. At $T=1$, G_{tot} obviously equals zero since this is the no-lot splitting case: in this scenario, products are moved between the machines in process batch sizes, such that a gap can never

occur. At $T=2$ and $T=3$, all servers in the system are dominated by server 1, which is a capacity server. Hence, it follows from the discussion above that G_{tot} is concave in T . Moreover, for these values of T , both expression (19) and (22) are positive, such that G_{tot} is increasing in T . At $T=4$, server 1 is still the dominating server for all machines in the system, but now the sum of expression (19) and expression (22) is negative: an increase in T leads to a decrease in total gap time on the delay servers, that overcompensates the increase in total gap time on the capacity servers. This is the gap compensation effect.

At $T=6$, servers 1 to 3 are dominated by server 1, and servers 4 and 5 by server 4. The total gap time on forklift number 2 reaches zero: it is constantly traveling back and forth between the consecutive machines. The sum of expression (19) and expression (22) is again positive: the gap compensation effect has disappeared.

Note that, as described above, the disappearance of the gap compensation effect is caused by the fact that at $T=6$, delay server 4 is no longer dominated by capacity server 1. If we again presumed that forklift 2 had the same processing and empty travel times as forklift 1, the curve of G_{tot} would look as in Figure 5. Server 1 would be the dominating server for all resources, at all levels of T . As can be seen, the gap tradeoff effect now plays until $T=N=12$.

5. Minimizing the total costs in terms of the transfer batch size used

The purpose of this section is to derive an expression for the total costs per production cycle, i.e. the sum of the inventory holding costs, the transportation costs and the gap costs. We first discuss each of these costs components in detail. Next, we explore the shape of the total cost function in terms of the transfer batch sizing policy.

a) Inventory holding costs

In order to determine the average inventory holding costs per production cycle, we make use of Little's Law ([7], [11]). Applied to our setting, this law states that the average number of transfer batches in the system (WIP_t) will be equal to the throughput rate of transfer batches (λ_t) times the average transfer batch lead time ($E(W)$):

$$WIP_t = \lambda_t * E(W) \quad (23)$$

The average inventory holding costs ($Cost_{Inv}$) per production cycle can then be calculated by multiplying this average work-in-process inventory by a specified holding cost per product unit per cycle, $c_{Inv} (\geq 0)$, times the transfer batch size L :

$$Cost_{Inv} = (c_{Inv} * L) * \lambda_t * E(W) \quad (24)$$

where $E(W)$ is given by expression (8) or, equivalently, expression (10). As $E(W)$ is convex in T and $L=N/T$, it follows that the average inventory holding cost will also be convex in T .

b) Total gap costs

In order to determine the gap costs per production cycle, it is assumed that each time unit that a server m is available without being productive involves a fixed cost $c_{gap,m} (\geq 0)$. In reality, the actual value of $c_{gap,m}$ will depend upon the variable costs associated with keeping the server available for production. For example, $c_{gap,m}$ will be higher on servers that require high energy or labor costs (like heating operations, where an oven has to be kept at constant temperature and operators may be required to load and unload the transfer batches) compared to operations that can be shut down during a gap and/or require no labor to operate (like a conveyor belt).

In our model, we make the simplifying assumption that $c_{gap,m}$ is the same for all servers m : $c_{gap,m} = c_{gap}$. The total gap costs ($Cost_{gap}$) per production cycle can then be expressed as:

$$Cost_{gap} = c_{gap} * G_{tot} \quad (25)$$

where G_{tot} is given by expression (9) or, equivalently, expression (18). As c_{gap} is assumed to be independent of the transfer batch sizing policy, the gap costs will vary with the transfer batch sizing policy in the same way as G_{tot} varies in terms of T .

c) Transportation costs

It is assumed that each unit of time that a transportation server is traveling during the production cycle (either in loaded or in empty condition) entails a fixed transportation cost $c_{trans} (\geq 0)$. The total transportation costs ($Cost_{trans}$) are then given by:

$$Cost_{trans} = \sum_{m \in M_{trans}} c_{trans} * T * (X_m + B_m) \quad (26)$$

From expression (26), it is clear that $Cost_{trans}$ will increase linearly in terms of T .

d) The total cost function

The optimization problem can now be stated as follows:

$$\begin{aligned} \underset{T}{Min} \quad & Cost_{Inv} + Cost_{gap} + Cost_{trans} \\ s.t. \quad & 1 \leq T \leq N \\ & \rho_m < 1 \quad (\forall m) \end{aligned} \quad (27)$$

The notation ρ_m in the second constraint refers to the utilization of server m . As mentioned before in the discussion about the delay effect, this constraint may be important in systems with delay servers: using a subplot size strictly smaller than L_c will cause the system to become unstable.

From the preceding discussion, we can derive that the shape of the total cost function will be highly determined by the relative proportion of the underlying cost components. As $Cost_{trans}$ is linearly increasing in T and $Cost_{Inv}$ is convex in T , we know that the sum of these two components will always be convex⁴. Consequently, when c_{gap} is equal to zero or when c_{gap} is low relative to c_{trans} and c_{inv} , the total cost function will be convex. However, when c_{gap} is high relative to c_{trans} and c_{inv} , the shape of the objective function will be dominated by the shape of $Cost_{gap}$ (i.e., the shape of G_{tot}). As described in section 5, this relationship is no longer convex. For example, assume that $c_{gap} = 0$, $c_{inv} = 2$, $c_{trans} = 0.5$ and that the interarrival time of process batches in front of the first machine is 78 hours. Figure 6 shows the resulting total cost function, along with the curves for the three cost components. The total cost function is indeed convex; the minimum total cost occurs at $T=2$, and is equal to 26.46.

Let's now assume that $c_{gap} = 0.5$. Figure 7 shows that the total cost is now no longer convex in T . Due to the impact of the total gap cost on the objective function, the total cost is now minimized at $T=1$, which implies that it is optimal not to use any lot splitting at all. This yields a total cost of 28.92.

The fact that the objective function is not always convex prohibits the development of an elegant optimization algorithm that is valid in all cases. However, in some cases, we can use our intuitive insight into the behavior of the system. It then suffices to do some pre-processing, comparing the total costs for a limited number of transfer batch sizing policies, in order to determine the optimal policy.

---

⁴ It is assumed here that $Cost_{trans} < Cost_{Inv}$ in the no-lot splitting case ($T=1$). If this is true, the sum of $Cost_{trans}$ and $Cost_{Inv}$ will indeed be convex in T . If it is not true, the sum of $Cost_{trans}$ and $Cost_{Inv}$ will be increasing in T .

First, we have to check whether the objective function is convex. To check for convexity, it suffices to determine the total costs at $T=1$ and at $T=S$, with S the smallest feasible number of transfer batches strictly larger than 1 . If the total cost at $T=1$ is higher than the total cost at $T=S$, we can conclude that the total cost function is convex and a discrete convex optimization algorithm could be used.

If the total cost at $T=1$ is lower than the total cost at $T=S$, we know that the total cost function is not convex. We then have three possible and mutually exclusive outcomes, which are summarized in Table 2. If the system does not contain any delay server, we know that the optimal transfer batch sizing policy will be $T=1$ (as the gap costs continuously increase in terms of T). If the system contains one or more delay servers, but none of these ever becomes dominating, we know that the gap tradeoff effect will play until $T=N$. It then suffices to determine the total costs at $T=1$ and $T=N$; the policy that yields the minimal total cost, will be the optimal lot splitting sizing policy. If, however, the system contains delay servers that become dominating at certain values of T , the optimum can not be intuitively derived. A simple enumeration of all possible transfer batch sizing policies with the related total costs then provides the fastest and easiest way to determine the optimal policy.

6. Conclusions

In the preceding analysis, we have been able to derive formal expressions for the calculation of gaps and transfer batch lead times in a flow shop with deterministic setup and processing times. Based upon these expressions, we were able to gain insight into the different effects governing the relationship between the transfer batch sizing policy on the one hand, and the average transfer batch lead time and the total gap time on the other hand. Next, the expressions allowed us to formulate a cost

model that takes into account the average inventory holding costs, the transportation costs and the gap costs per production cycle. It has been shown that, when gap costs are negligible or non-existent, the total cost function will be convex, implying that the optimal number of sublots will be strictly larger than 1. However, the incorporation of gap costs into the model may cause the objective function to exhibit a totally different shape: depending upon the shape of the underlying total gap function and the importance of the gap costs relative to the inventory holding costs and the transportation costs, it may be optimal to use no lot splitting at all.

We believe that the proposed model provides useful insights for production environments in which intermittent idling of machines is not technologically prohibited (e.g., due to process or product characteristics), but does entail a penalty (i.e., a cost). The contribution of this paper lies in the explicit calculation of this penalty, such that it can be taken into account as a separate cost component when designing the optimal lot splitting policy for a given system. The insights developed in this paper may also provide useful guidelines for process design. It has been shown that both total gap time and average lead time may be adversely affected when delay servers become dominant. Hence, it will be advisable to keep transportation times small, e.g. by locating production resources closely together (as in manufacturing cells). Moreover, if the process designer has control over the setup and processing times at the different servers, it is clear from expression (17) that he can obtain lead time reduction and avoid gaps altogether by making sure that, at the chosen policy, all servers are equally dominant.

Future research will focus on designing a model for transfer batch optimization in stochastic flow shops.

Appendix A

From expression (8), we have:

$$E(W) = \sum_{m=1}^M S_m + \sum_{m=1}^{M-1} X_m + T * X_M + (T-1) * B_M + G(M, T)$$

This can be rewritten as:

$$E(W) = \sum_{m=1}^M S_m + \sum_{m=1}^M X_m + (T-1) * (X_M + B_M) + G(M, T)$$

with (cf expressions (3) and (5)):

$$\begin{aligned} & (T-1) * (X_M + B_M) + G(M, T) \\ &= (T-1) * (X_M + B_M) + \max[0, (T-1) * (X_{M-1} + B_{M-1} - X_M - B_M) - S_M + G(M-1, T)] \\ &= -S_M + \max[(T-1) * (X_M + B_M) + S_M, (T-1) * (X_{M-1} + B_{M-1}) + G(M-1, T)] \end{aligned}$$

Applying the same technique recursively for all $G(M-k, T)$ ($k=1, \dots, M-1$) leads to the following expression:

$$\begin{aligned} & (T-1) * (X_M + B_M) + G(M, T) \\ &= -\sum_{m=1}^M S_m + \max \left[\begin{aligned} & \sum_{m=1}^M S_m + (T-1) * (X_M + B_M), \\ & \max \left[\begin{aligned} & \sum_{m=1}^{M-1} S_m + (T-1) * (X_{M-1} + B_{M-1}), \\ & \max \left[\begin{aligned} & \sum_{m=1}^{M-2} S_m + (T-1) * (X_{M-2} + B_{M-2}), \\ & \dots \end{aligned} \end{aligned} \right] \end{aligned} \right] \end{aligned} \right] \end{aligned}$$

This can be rewritten as:

$$\begin{aligned} & (T-1) * (X_M + B_M) + G(M, T) \\ &= -\sum_{m=1}^M S_m + \max \left[\max_{i \in M_D} \left[\sum_{m=1}^i S_m + (T-1) * (X_i + B_i) \right], \max_{j \in M_C} \left[\sum_{m=1}^j S_m + (T-1) * \frac{N}{T} * X_{u,j} \right] \right] \end{aligned}$$

such that $E(W)$ reduces to:

$$\begin{aligned} E(W) &= \sum_{m \in M_C} \frac{N}{T} * X_{u,m} + \sum_{m \in M_D} X_m \\ &+ \max \left[\max_{i \in M_D} \left[\sum_{m=1}^i S_m + (T-1) * (X_i + B_i) \right], \max_{j \in M_C} \left[\sum_{m=1}^j S_m + (T-1) * \frac{N}{T} * X_{u,j} \right] \right] \end{aligned}$$

which is what we had to prove.

Appendix B

From Appendix A, we know that:

$$(T-1) * (X_M + B_M) + G(M, T) \\ = -\sum_{m=1}^M S_m + \max \left[\max_{i \in M_D} \left[\sum_{m=1}^i S_m + (T-1) * (X_i + B_i) \right], \max_{j \in M_C} \left[\sum_{m=1}^j S_m + (T-1) * \frac{N}{T} * X_{u,j} \right] \right]$$

such that

$$G(M, T) \\ = -\sum_{m=1}^M S_m - (T-1) * (X_M + B_M) \\ + \max \left[\max_{i \in M_D} \left[\sum_{m=1}^i S_m + (T-1) * (X_i + B_i) \right], \max_{j \in M_C} \left[\sum_{m=1}^j S_m + (T-1) * \frac{N}{T} * X_{u,j} \right] \right]$$

or, for an arbitrary server m ,

$$G(m, T) = \\ -\sum_{j=1}^m S_j + \max \left[\max_{\substack{i \in M_C \\ i \leq m}} \left[\sum_{k=1}^i S_k + (N - \frac{N}{T}) * X_{u,i} \right], \max_{\substack{j \in M_D \\ j \leq m}} \left[\sum_{k=1}^j S_k + (T-1) * (X_j + B_j) \right] \right] \\ \begin{cases} -(T-1) * (X_m + B_m) & \text{if } m \in M_D \\ -(N - \frac{N}{T}) * X_{u,m} & \text{if } m \in M_C \end{cases}$$

which is what we had to prove.

References

- [1] Baker, K.R. and D. Jia, 1993, A comparative study of lot streaming techniques, Omega 21, 561-566
- [2] Bukchin, J., M. Tzur and M. Jaffe, 2002, Lot splitting to minimize average flow time in a two-machine flow-shop, IIE Transactions 34, 953-970

- [3] Çetinkaya F.C., 1994, Lot streaming in a two-stage flow shop with setup, processing and removal times separated, *Journal of the Operational Research Society* 45, 1445-1455.
- [4] Chiang C., 2001, Order splitting under periodic review inventory systems, *International Journal of Production Economics* 70, 67-76.
- [5] Goldratt, E.M. and R.E. Fox, 1986, *The Race* (North River Press, New York).
- [6] Graves, S.C. and M.M. Kostreva, 1986, Overlapping operations in materials requirements planning, *Journal of Operations Management* 6, 283-294.
- [7] Hopp, W. J. and M. L. Spearman, 2000, *Factory Physics: Foundations of Manufacturing Management* (McGraw-Hill, Chicago).
- [8] Jacobs, F.R. and D.J. Bragg, 1988, Repetitive lots: flow-time reductions through sequencing and dynamic batch sizing, *Decision Sciences* 19, 281-294.
- [9] Karmarkar, U., 1987, Lot sizes, lead times and in-process inventories, *Management Science* 33, 409 – 423.
- [10] Kim, S.-L. and D. Ha, 2003, A JIT lot-splitting model for supply chain management: Enhancing buyer-supplier linkage, *International Journal of Production Economics* 86, 1-10.

- [11] Kleinrock, L., 1975, Queueing Systems (John Wiley, New York).
- [12] Kropp, D.H. and T.L. Smunt, 1990, Optimal and heuristic models for lot splitting in a flow shop, *Decision Sciences* 21, 691-709.
- [13] Lambrecht, M.R. , P.L. Ivens and N.J. Vandaele, 1998, ACLIPS: A Capacity and Lead Time Integrated Procedure for Scheduling, *Management Science* 44, 1548-1561
- [14] Litchfield, J.L. and R. Narasimhan, 2000, Improving job shop performance through process queue management under transfer batching, *Production and Operations Management* 9, 336-348
- [15] Ramasesh, R.V., H. Fu, D.K.H. Fong and J.C. Hayya, 2000, Lot streaming in multistage production systems, *International Journal of Production Economics* 66, 199-211
- [16] Ruben, R.A. and F. Mahmoodi, 1998, Lot splitting in unbalanced production systems, *Decision Sciences* 29, 921-949
- [17] Smunt, T.L., A.H. Buss and D.H. Kropp, 1996, Lot splitting in stochastic flow shop and job shop environments, *Decision Sciences* 27, 215-238
- [18] Trietsch, D. and K.R. Baker, 1993, Basic techniques for lot streaming, *Operations Research* 41, 1065-1076

Figure 1: Flowchart for the example

Figure 2: Average transfer batch lead time $E(W)$ (in hours) in terms of (a) the number of transfer batches T , and (b) the transfer batch size L , for the example given in Table 1

Figure 3: Average transfer batch lead time $E(W)$ (in hours) in terms of the number of transfer batches T , if $X_4=B_4=1$

Figure 4: Total gap time in the system (G_{tot}) (in hours) in terms of the number of transfer batches T , for the example given in Table 1

Figure 5: Total gap time in the system (G_{tot}) (in hours) in terms of the number of transfer batches T , when $X_4=B_4=1$

Figure 6: Total cost in terms of T , for the example given in Table 1 ($c_{gap} = 0$, $c_{inv} = 2$,

$$c_{trans} = 0.5, \lambda_{PB} = 1/78 \text{ per hour})$$

Figure 7: Total cost in terms of T , for the example given in Table 1 ($c_{gap} = 0.5$,

$$c_{inv} = 2, c_{trans} = 0.5, \lambda_{PB} = 1/78 \text{ per hour})$$

	M1	F1	M2	F2	M3
Setup time S (hours)	2	0	2	0	1
Processing time X per transfer batch (hours)	$3*L$	1	$1*L$	3	$2*L$
Breakdown time B (hours)	0	1	0	3	0

Table 1: Setup and processing time data (in hours) for the different machines

System characteristics	Optimal lot splitting policy
No delay servers in the system	$T_{opt} = 1$
Delay servers present, but never dominating	$T_{opt} = 1$ or $T_{opt} = N$
Delay servers present and dominating at certain values of T	<i>No intuitive answer</i> <i>Enumeration</i>

Table 2: Overview of optimal lot splitting policies, in terms of the system characteristics, when the objective function is not convex