Estimating process batch flow times in a two-stage stochastic flowshop with overlapping operations

I. Van Nieuwenhuyse • N. Vandaele

Department of Applied Economics, University of Antwerp, Prinsstraat 13, 2000 Antwerp, Belgium
inneke.vannieuwenhuyse@ua.ac.be • nico.vandaele@ua.ac.be

This paper focuses on modelling the impact of lot splitting on average process batch flow times, in a two-stage stochastic flowshop. It is shown that the traditional queueing methodology for estimating flow times cannot be directly applied to a system with lot splitting, as the arrival process of sublots at the second stage is not a renewal process. Consequently, an embedded queueing model is developed in order to approximate the average flow time of the flags through the system; from the flow time of the flags, the flow time of process batches can then be derived. The model turns out to yield very satisfactory results, and provides a tool to quantify the reduction in flow time that can be obtained by overlapping operations at different processing stages. Moreover, it allows to model the trade-off between flow time improvement and gap time occurrence by using it within the scope of a cost model.

1. Introduction

The past decades have witnessed a surge in research efforts, aimed at modelling the performance of stochastic production or service systems by means of queueing theory (e.g. Karmarkar et al. 1985, Karmarkar 1987, Lambrecht et al. 1998). While exact models are only available for a limited number of settings (like M/M/1 or M/G/1, see e.g. Kleinrock 1975), approximative models have been developed to estimate performance measures (like average flow times and WIP) under more general conditions, such as GI/G/1 or GI/G/m (see e.g. Kramer and Lagenbach-Belz 1967, Whitt 1983, Whitt 1993). These models are highly valued for their speed and ease of use in providing estimates of the performance indicators of interest, as opposed to e.g. simulation models (see Suri et al. 1993, Suri 1998).

It is currently well-known that the average flow time of products through a system is influenced by a range of managerial decisions, such as e.g. the product mix being produced in the shop, the layout of the shop, and (last but not least) the batching policies used on the shopfloor. The impact of batching policies on flow times is particularly interesting to analyze: by setting batch sizes in a deliberate way, managers can in fact obtain flow time improvements without a radical intervention in the system, and without large financial investments. Hence, setting batch sizes in

a production system is an important control (see e.g. Hopp and Spearman 2000, Lambrecht et al. 1998, Benjaafar 1996).

When studying the impact of batching policies, a distinction should be made between two types of batches: i.e., *process batches* and *transfer batches*. A process batch (also referred to as a production batch or production lot) is defined as the quantity of a product processed on a machine without interruption by other items (Kropp and Smunt 1990). In multiple-product environments, the use of process batches is often unavoidable due to capacity considerations: to switch from one product type to the next (e.g., to change fixtures or dies), a setup or changeover time is necessary, which consumes part of the capacity of the machine. After a setup has been performed, a certain quantity of the product (the process batch size) can be produced. Hence, a process batch can also be defined as the quantity of a product produced between two consecutive setups.

The relationship between the process batch size and the average flow time has been thoroughly studied, and turns out to be convex (Karmarkar 1987, Suri 1998). Current queueing models are able to take into account the impact of the process batching policy (e.g. Karmarkar 1985a, 1985b), and the insight into the convex relationship has stirred the development of optimization procedures, aimed at determining the optimal process batch size for a given objective function (e.g. the weighted average flow time, Lambrecht et al. 1998).

The term transfer batch (also called transfer lot or sublot) on the other hand refers to the size of a sublot of the process batch, moved after production on one machine to another operation or machine (Kropp and Smunt 1990). The use of transfer batches is not caused by capacity considerations, but rather by flow considerations. Indeed, it is widely accepted that the use of transfer batch sizes smaller than the process batch size can reduce product flow times by smoothing workflow and minimizing congestion levels (e.g. Santos and Magazine 1985, Benjaafar 1996, Goldratt and Cox 1984, Hopp et al. 1990 and Umble and Srikanth 1995). This is due to the mechanism of *overlapping operations*: 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 (e.g. Graves and Kostreva 1986, Jacobs and Bragg 1988, Litchfield and Narasimhan 2000).

Up to now, the impact of lot splitting on flow times in a stochastic setting has received relatively little attention in the research literature. There are a number of papers that have studied the problem by means of discrete-event simulation (e.g. Wagner and Ragatz 1994, Smunt et al. 1996, Jacobs and Bragg 1988, Ruben and Mahmoodi 1998). In the queueing literature however, the dis-

tinction between process batching and transfer batching is currently overlooked, as present models assume that products move between the machines in process batch sizes (this is also referred to as the *lot-for-lot policy*, see e.g. Van Nieuwenhuyse and Vandaele 2004a). To the best of our knowledge, only a few studies (Bozer and Kim 1996, Benjaafar 1996) have made an attempt to analyze the impact of transfer batching on flow times analytically. However, the assumptions made in these papers are rather restrictive (e.g. Poisson arrival processes and/or Poisson service processes at the different stages of the system, regardless of the batching decisions).

This apparent shortcoming of the current queueing models provided the basis for our research. In this paper, we will study a two-stage, single-product flowshop, with a general arrival and service process. The application of the queueing methodology to this type of system entails a number of difficulties, which, as will be shown, call for the development of an embedded queueing model. This model turns out to yield very good estimations of the average process batch flow time, when compared to simulation results.

Though the setting studied here is limited to two stages and one product type, we believe this work provides an important first contribution, by presenting a generic methodology which, through adequate modifications, may in the future be extended towards more general settings (such as systems with multiple product types and job shop settings).

As a starting point, section 2 briefly describes the structure and assumptions of the two-stage, single-product flowshop with overlapping operations. It also outlines the difficulties arising in modelling the impact of overlapping operations by means of the queueing methodology. These hurdles provide the argument for the development of the embedded queueing model, the framework of which is described in detail in section 3. Section 4 then tests the performance of this embedded queueing model for the two-stage stochastic system versus simulation results, and section 5 presents a cost model to analyze the trade-off between flow time improvement and the occurrence of gap times in a stochastic flowshop. Finally, section 6 summarizes the conclusions.

2. The two-stage flowshop with overlapping operations

2.1 Structure and assumptions

The system we will consider is a two-stage, stochastic system with a single product type. It is assumed to be an *open* system: production orders are launched into the system, undergo operations subsequently on the first and second stage, and leave the system after being processed. For the

sake of simplicity, we will assume that the size of a production order equals a process batch size. The interarrival times of process batches in the system, as well as the setup and processing times on the two servers, are generally distributed. Both servers are assumed to be capacity servers (implying that the processing time of a sublot is dependent upon the size of the sublot), and are preceded by a buffer with infinite capacity and FIFO queueing discipline.

Figure 1 illustrates the progress of two consecutive process batches, consisting of 4 sublots each, through the system.

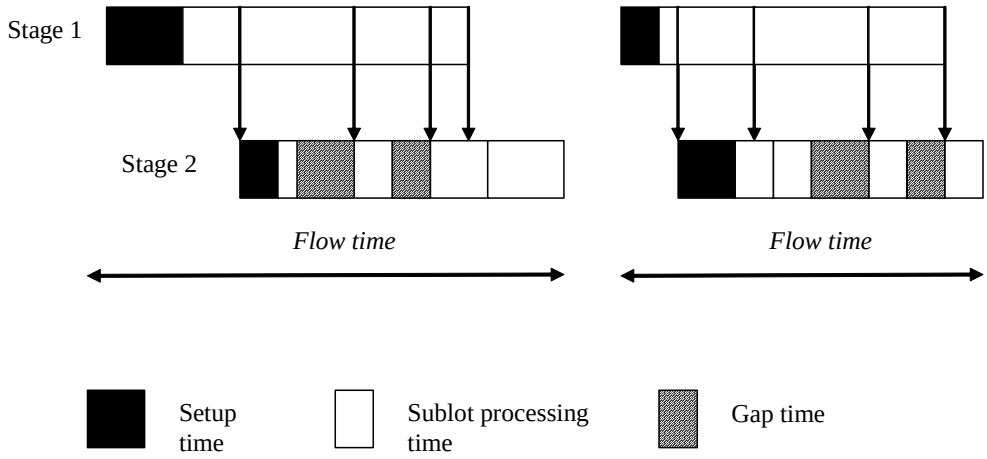

Figure 1: Flowchart of 2 process batches, consisting of 4 sublots, going through a 2-stage stochastic flowshop

Upon arrival, a process batch may have to wait in front of the first stage before being processed (e.g., when this stage is still busy processing a previous process batch). When the server becomes idle, the setup is performed and the product units in the process batch are processed one by one. Products move between the two machines in transfer batches: it is assumed that the transfer batch size is a common divisor of the process batch size, such that a process batch is split into an integer number of sublots. As soon as the transfer batch size is reached, the sublot is moved to the second stage (this is referred to in the literature as *batch availability*, e.g. Bukchin et al. 2002 and Santos and Magazine 1985). It is important to note that, in our setting, the setup time on stage two can not start before the first transfer batch of the involved process batch has arrived in its input buffer. This type of setup has been referred to in the literature as an *attached* setup (e.g. Chen and Steiner 1998, Potts and Kovalyov 2000). Hence, the first transfer batch of a process batch acts as a *flag* (Smunt et al. 1996): its arrival in front of the second stage authorizes the start of the setup, thereby causing the operations on stage two to partly overlap with the operations on stage

one.

Upon arrival at the second stage, the flag may have to wait until the server becomes idle. As soon as it does, the setup for the process batch is performed and the flag is processed. The remaining sublots are processed as soon as they are finished on stage one, provided that stage two is available at that moment; otherwise, they will wait in the input buffer of the second stage.

The structure of the system described above implies that the second server may remain idle between the processing of two consecutive sublots, belonging to the same process batch. These idle times are referred to as *gaps* (Van Nieuwenhuyse 2004, Van Nieuwenhuyse and Vandaele 2004a). As illustrated in Figure 1, they occur whenever the second stage finishes processing a sublot before the next sublot is available from the first stage. In deterministic settings, gaps can be avoided by balancing the processing rates on the different machines in the shop (this is referred to as the *no-idling assumption*, e.g. Baker and Jia 1993, Ramasesh et al. 2000); in stochastic settings however, gaps may occur even when the system is perfectly balanced, due to the inherent variability in the setup and processing times at the different stages.

The occurrence of gaps obviously leads to an increase in the average makespan of a process batch on stage two, without adding value to the product. Nevertheless, as will become clear below, the use of lot splitting will drive down average process batch flow times in our system, thanks to the overlapping of operations.

2.2 Obstacles to the application of the queueing methodology

Current open queueing models analyze the performance of a production network by means of the *decomposition approach*. In this approach, the network is decomposed into the different building blocks (the servers). Each server has its own specific arrival and processing characteristics. In a first step, the queue in front of each server is analyzed separately based on three different input parameters: i.e., parameters for the arrival process, the service process and the utilization rate of the server (see e.g. Whitt 1983, Whitt 1994, Buzacott and Shantikumar 1985, Suri et al. 1993 for more details). The different servers in the network are linked together by means of so-called linking equations, which ensure that the output stream of a particular server represents the input stream for the next server in the product's routing. Finally, the results are recomposed in order to determine the performance of the entire production system.

The decomposition approach is however based upon the critical assumption that the interarrival times and service times at each service center are both independent and identically distributed

(IID), and that there is no correlation between interarrival and service times. Unfortunately, this assumption does not hold in a system with lot splitting.

To substantiate this point, Figure 2 shows the arrival pattern for sublots at server 2, for the example previously shown in Figure 1.

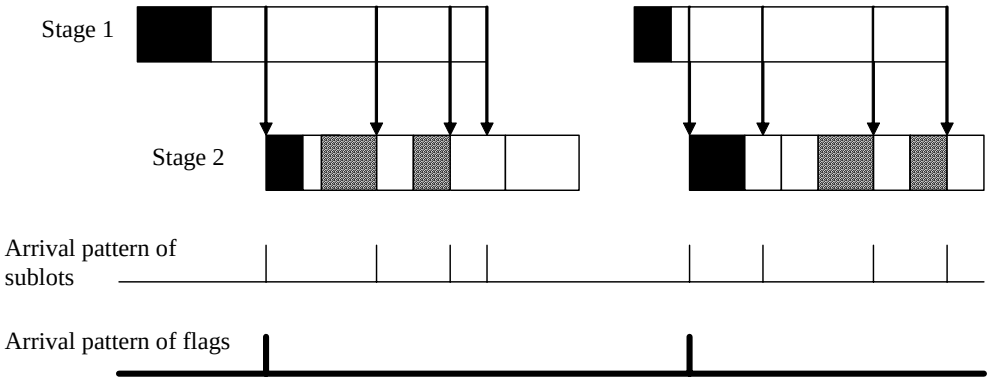

Figure 2: Flowchart with arrival pattern of sublots and flags

Typically, the arrival process will exhibit a *bursty* pattern: as soon as the flag has arrived, the remaining sublots of the same production batch will arrive within rather short time intervals. Next, there will be a longer time interval before the next flag arrives. Consequently, the interarrival times of sublots are correlated, and no longer stem from the same probability distribution. As mentioned above, this violates one of the basic assumptions underlying the decomposition approach.
However, this issue may be circumvented by only considering the flags as entities moving through the network, instead of looking at individual transfer batches. Figure 2 illustrates the argument: indeed, the bursty pattern is caused by the fact that the interarrival times of the remaining sublots do not follow the same probability distribution as the interarrival times of the flags. When we only consider the flags as entities moving through the network, and ignore the remaining transfer batches, the bursty pattern disappears. The arrival process of flags in front of a machine can then be approximated by a renewal process.
These observations lead us to the conclusion that, although the queueing methodology cannot be applied to estimate the average flow time of individual sublots, it may be useful in modeling the average flow time of the flags. From the average flow time of the flags, we can then derive the average flow time of a process batch. This is explained in detail in the next section.

3. Embedded queueing model

In this section, we will describe the structure of the embedded queueing model. It is very similar to the structure of the traditional open queueing models with a lot-for-lot policy ; the main difference is that the level of analysis is not the process batches but the flags. The following notation will be used:

m = machine number index ($m = 1, ..., M$)

N = process batch size

L = sublot size

T = number of sublots in a process batch ($T = \frac{N}{L}$)

$Y_{f,m}$ = interarrival time of flags at stage m

$Y_{PB,m}$ = interarrival time of process batches at stage m

$\lambda_{f,m}$ = average arrival rate of flags at stage m

$\lambda_{PB,m}$ = average arrival rate of process batches at stage m

SU_m = setup time on stage m

x_m = unit processing time on stage m

G_2 = total gap time on stage 2

$W_{f,m}$ = waiting time in queue of flags at stage m

P_m = process batch makespan at stage m

ρ_m = utilization of machine m

$\tau_{f,m}$ = interdeparture time of flags at machine m

F_{PB} = flow time of a process batch through the flow shop

F_f = flow time of a flag through the flow shop

For any random variable Z, $E(Z)$ will refer to the mean of the variable, s_Z^2 to its variance, and c_Z^2 to its squared coefficient of variation (SCV).

3.1 Model structure and assumptions

As sublots are regrouped into process batches at the end of the routing, the average flow time of a process batch going through the system can be approximated by estimating the average flow time of the flag, and adding the average extra time elapsing on the second server before the completed process batch leaves the system. This average extra time consists of the average total gap time occurring on the second server, and the average processing time of the remaining $T - 1$ transfer

batches. This yields:

$$E(F_{PB}) = E(F_f) + (T-1) * L * E(x_2) + E(G_2) \tag{1}$$

As mentioned above, the objective of the embedded queueing model is to yield a realistic estimate of the $E(F_f)$. This flow time consists of three components: the waiting time spent in queue at both servers, the setup time experienced at both servers, and the processing time of the flag itself. Hence:

$$E(F_f) = \sum_{m=1}^{2} E(W_{f,m}) + E(SU_m) + L * E(x_m) \tag{2}$$

Regarding each stage m as a $G/G/1$ server, we expect that the average waiting time of a flag in front of each stage can be approximated by standard $G/G/1$ queueing expressions (such as the Kraemer-Lagenbach-Belz expression, see Kraemer and Lagenbach-Belz 1967), which is in general given by:

$$E(W_{f,m}) = \begin{cases} = \frac{\rho_m^2 * (c_{Y_{f,m}}^2 + c_{P_m}^2)}{2\lambda_{f,m} * (1-\rho_m)} \exp\{\frac{-2*(1-\rho_m)*(1-c_{Y_{f,m}}^2)^2}{3*\rho_m*(c_{Y_{f,m}}^2 + c_{P_m}^2)}\} & c_{Y_{f,m}}^2 \leq 1 \\ = \frac{\rho_m^2 * (c_{Y_{f,m}}^2 + c_{P_m}^2)}{2\lambda_{f,m} * (1-\rho_m)} & c_{Y_{f,m}}^2 > 1 \end{cases} \tag{3}$$

For the first stage, this yields no particular problems. As setup and processing times on a server are assumed to be independent, we may write:

$$c_{P_1}^2 = \frac{s_{SU_1}^2 + N * s_{x_1}^2}{[E(SU_1) + N * E(x_1)]^2} \tag{4}$$

Moreover, as products arrive in process batches in front of stage 1, $c_{Y_{f,m}}^2$ will be equal to the SCV of the interarrival times of process batches at stage 1, and hence is given:

$$c_{Y_{f,1}}^2 = c_{Y_{PB,1}}^2$$

The utilization rate of stage 1 is given by:

$$\rho_1 = \frac{E(SU_1) + N * E(x_1)}{E(Y_{PB,1})}$$

Hence, $E(W_{f,1})$ can be readily calculated.

For the second stage however, the expressions for $c_{Y_{f,2}}^2$ and $c_{P_2}^2$ are more complicated: the interarrival time of flags at the second stage ($Y_{f,2}$) is determined by the interdeparture time of flags at the first stage (which we will denote by $\tau_{f,1}$), and the makespan of a process batch at the second stage (P_2) needs to include the total gap time. Consequently, estimating $E(W_{f,2})$ first requires us to develop appropriate approximations for these input parameters.

Approximations for $\tau_{f,1}$ and P_2 have been studied in detail in earlier research (Van Nieuwenhuyse 2004, Van Nieuwenhuyse and Vandaele 2004b, Van Nieuwenhuyse and Vandaele 2005). For ease of reference, we highlight the most important results from these studies in the following subsection.

3.2 Approximating $E(W_{f,2})$

3.2.1 Approximation for $c^2_{Y_{f,2}}$

From the structure of the system, it is clear that $c^2_{Y_{f,2}}$ equals $c^2_{\tau_{f,1}}$:

$$c^2_{Y_{f,2}} = c^2_{\tau_{f,1}} = \frac{s^2_{\tau_{f,1}}}{E[\tau_{f,1}]^2} \tag{5}$$

In a stable system, the average interdeparture time of flags $E[\tau_{f,1}]$ will equal the average interarrival time of flags $E[Y_{f,1}]$, and hence equals the average interarrival time of process batches in front of stage 1 ($E[Y_{PB,1}]$):

$$E[\tau_{f,1}] = E[Y_{f,1}] = E[Y_{PB,1}] \tag{6}$$

As shown in Van Nieuwenhuyse and Vandaele (2004b), $s^2_{\tau_{f,1}}$ can be approximated by:

$$s^2_{\tau_{f,1}} = s^2_{\tau_{PB,1}} + 2(1-\rho_1^2)Cov[\sum_{i=2}^{T} X_{1,i}, R_1]_{app} \tag{7}$$

In this expression, $s^2_{\tau_{PB,1}}$ refers to the variance of the interdeparture times of process batches under a lot-for-lot policy. It is given by (see Marshall 1968):

$$\begin{aligned} s^2_{\tau_{PB,1}} \quad = \quad & 2s^2_{SU_1} + N * s^2_{x_1} + s^2_{Y_{PB,1}} \\ & -2E[W_{PB,1}] * [E[Y_{PB,1}] - E[SU_1] - N * E[x_1]] \end{aligned} \tag{8}$$

The notation $Cov[\sum_{i=2}^{T} X_{1,i}, R_1]_{app}$ in expression (7) refers to the covariance between the processing times of the remaining transfer batches on stage 1 ($\sum_{i=2}^{T} X_{1,i}$), and the idle time that may occur afterwards, before the processing of the next production batch starts (this idle time is denoted by R_1). This covariance is analytically intractable, but it has been shown (Van Nieuwenhuyse and Vandaele 2004b) that it may be adequately approximated by rewriting:

$$\sum_{i=2}^{T} X_{1,i} = V$$

$$R_1 = \max[0, M]$$

$$M = Y_{f,1} - SU_1 - T * X_1$$

and assuming normal probability distributions for both V and M. Note that this assumption implies a zero-inflated normal probability distribution for R_1 (see e.g. Blumenfeld 2001). This

allows us to write:

$$\begin{aligned} & Cov[\textstyle\sum_{i=2}^{T} X_{1,i}, R_1]_{app} \\ = \; & E[V * \max(0, M)] - E[V]E[\max(0, M)] \\ = \; & \textstyle\int_{-\infty}^{+\infty} \int_{0}^{+\infty} v * m * f_{V,M}(v, m) * dmdv \\ & -(T-1) * E[X_1] * (\frac{Exp\{\frac{E[M]^2}{2s_M^2}\} * s_M^2}{\sqrt{2\Pi} s_M} + \frac{E[M]}{2} * (1 + Erf[\frac{E[M]}{\sqrt{2} s_M}])) \end{aligned} \tag{9}$$

with

$$\begin{aligned} f_{V,M}(v,m) &= \frac{1}{2\pi s_V s_M \sqrt{1-\rho_{V,M}^2}} * h(v,m) \\ h(v,m) &= e^{\frac{-1}{2(1-\rho_{V,M}^2)}[\frac{(v-E[V])^2}{s_V^2} - 2\rho_{V,M}\frac{(v-E[V])}{s_V}\frac{(m-E[M])}{s_M} + \frac{(m-E[M])^2}{s_M^2}]} \\ \rho_{V,M} &= \frac{-s_V^2}{s_V * s_M} \end{aligned} \tag{10}$$

and

$$\begin{aligned} E[V] &= (T-1) * E[X_1] \\ E[M] &= E[Y_{f,1}] - (E[SU_1] + T * E[X_1]) \\ s_V^2 &= (T-1) * s_{X_1}^2 \\ s_M^2 &= s_{Y_{f,1}}^2 + s_{SU_1}^2 + T * s_{X_1}^2 \end{aligned} \tag{11}$$

Note that the parameters used in calculating $E[M]$, s_M^2, $E[V]$ and s_V^2 are all known exactly. The notation $Erf[x]$ in expression (9) refers to the error function in x:

$$Erf[x] = \frac{2}{\sqrt{\pi}} \int_0^x \exp\{-t^2\} dt \tag{12}$$

Combining expressions (5), (6) and (7), we get the following approximation for $c_{\tau_{f,1}}^2$:

$$c_{\tau_{f,1}}^2 = c_{\tau_{PB,1}}^2 + \frac{2(1-\rho_1^2) Cov[\sum_{i=2}^{T} X_{1,i}, R_1]_{app}}{E[Y_{f,1}]^2} \tag{13}$$

This expression clearly reveals the relationship between $c_{\tau_{f,1}}^2$ in case of a lot splitting policy, and $c_{\tau_{PB,1}}^2$ with a lot-for-lot policy.

3.2.2 Approximation for $c_{P_2}^2$

From the definition of covariance, we know that:

$$c_{P_2}^2 = \frac{s_{P_2}^2}{E[P_2]^2} \tag{14}$$

Morever, the random variable P_2 is given by:

$$P_2 = SU_2 + \sum_{i=1}^{T} X_{2,i} + G_2$$

such that

$$E[P_2] = E[SU_2] + N * E[x_2] + E[G_2] \tag{15}$$

and

$$\begin{aligned} s^2_{P_2} &= s^2_{SU_2} + N * s^2_{x_2} + s^2_{G_2} \\ &\quad +2 * Cov(SU_2 + \textstyle\sum_{i=1}^{T} X_{2,i}, G_2) \end{aligned} \tag{16}$$

Hence, approximating $c^2_{P_2}$ requires us to find expressions for $E[G_2]$, $s^2_{G_2}$ and $Cov(SU_2 + \sum_{i=1}^{T} X_{2,i}, G_2)$.

Earlier research (Van Nieuwenhuyse and Vandaele 2005, Van Nieuwenhuyse 2004) has shown that the total gap time G_2 can be adequately approximated as follows:

$$G_2 = \max[Z, 0] \tag{17}$$

with Z a normally distributed random variable given by:

$$Z = \sum_{k=2}^{T} X_{1,k} - (SU_2 + \sum_{j=1}^{T-1} X_{2,j}) \tag{18}$$

This assumption implies a zero-inflated normal distribution for G_2, and yields the following approximation for $E[G_2]$:

$$E[G_2]_{app} = \frac{E[Z]}{2} + \frac{Exp\{\frac{-E[Z]^2}{2s^2_Z}\} * s^2_Z}{\sqrt{2\pi} s_Z} + \frac{E[Z]}{2} * Erf[\frac{E[Z]}{\sqrt{2} s_Z}] \tag{19}$$

in which $Erf[x]$ refers to the error function in x (see expression (12)), and $E[Z]$ and s^2_Z are given by:

$$\begin{aligned} E[Z] &= (T-1) * [E[X_1] - E[X_2]] - E(SU_2) \\ s^2_Z &= (T-1) * [s^2_{X_1} + s^2_{X_2}] + s^2_{SU_2} \end{aligned} \tag{20}$$

Using expression (15), this yields the following approximation for $E[P_2]$:

$$E[P_2]_{app} = E[SU_2] + N * E[x_2] + E[G_2]_{app} \tag{21}$$

The assumption of a zero-inflated normal distribution for G_2 also allows us to write the following approximation for $s^2_{G_2}$ (Blumenfeld, 2001):

$$\begin{aligned} s^2_{G_2,app} \approx\ & Exp\{\tfrac{-E[Z]^2}{2s^2_Z}\} * \tfrac{E[Z] * s_Z}{\sqrt{2\pi}} + \tfrac{E[Z]^2 + s^2_Z}{2} * (1 + Erf[\tfrac{E[Z]}{\sqrt{2} s_Z}]) \\ & -[\tfrac{Exp\{\frac{-E[Z]^2}{2s^2_Z}\} * s^2_Z}{\sqrt{2\pi} s_Z} + \tfrac{E[Z]}{2} * (1 + Erf[\tfrac{E[Z]}{\sqrt{2} s_Z}])]^2 \end{aligned} \tag{22}$$

Let's now turn to $Cov(SU_2 + \sum_{i=1}^{T} X_{2,i}, G_2)$. It is clear that $X_{2,T}$ is independent of G_2, such that we can write:

$$Cov(SU_2 + \sum_{i=1}^{T} X_{2,i}, G_2) = Cov(SU_2 + \sum_{i=1}^{T-1} X_{2,i}, G_2)$$

Rewriting

$$SU_2 + \sum_{i=1}^{T-1} X_{2,i} = U$$

and assuming a normal distribution for U (see Van Nieuwenhuyse 2004) allows us to write the following approximation for $Cov(SU_2 + \sum_{i=1}^{T} X_{2,i}, G_2)$:

$$\begin{aligned} & Cov(SU_2 + \textstyle\sum_{i=1}^{T} X_{2,i}, G_2)_{app} \\ \approx\ & E[U * \max[Z, 0]] - E[U]E[\max[Z, 0]] \\ \approx\ & \textstyle\int_{-\infty}^{+\infty} \int_{0}^{+\infty} u * z * f_{U,Z}(u, z) * dzdu \\ & -(E[SU_2] + (T-1)E[X_2]) * (\frac{Exp\{\frac{-E[Z]^2}{2s_Z^2}\} * s_Z^2}{\sqrt{2\pi} s_Z} + \frac{E[Z]}{2} * (1 + Erf[\frac{E[Z]}{\sqrt{2} s_Z}])) \end{aligned} \tag{23}$$

with $f_{U,Z}(u, z)$ denoting the bivariate normal density function for the two correlated variables U and Z:

$$\begin{aligned} f_{U,Z}(u, z) &= \frac{1}{2\pi s_U s_Z \sqrt{1-\rho_{U,Z}^2}} * h(u, z) \\ h(u, z) &= e^{\frac{-1}{2(1-\rho_{U,Z}^2)}[\frac{(u-E[U])^2}{s_U^2} - 2\rho_{U,Z}\frac{(u-E[U])}{s_U}\frac{(z-E[Z])}{s_Z} + \frac{(z-E[Z])^2}{s_Z^2}]} \end{aligned} \tag{24}$$

and with

$$\begin{aligned} E[U] &= E[SU_2] + (T-1)E[X_2] \\ s_U^2 &= s_{SU_2}^2 + (T-1) * s_{X_2}^2 \\ \rho_{U,Z} &= \frac{-s_{SU_2}^2 - (T-1)s_{X_2}^2}{s_U * s_Z} \end{aligned} \tag{25}$$

and $E[Z]$ and s_Z^2 as in expression (20).

Combining expressions (16), (22) and (23) then yields the following approximation for $s_{P_2}^2$:

$$\begin{aligned} s_{P_{2,app}}^2 &= s_{SU_2}^2 + N * s_{x_2}^2 + s_{G_{2,app}}^2 \\ &\quad +2 * Cov(SU_2 + \textstyle\sum_{i=1}^{T} X_{2,i}, G_2)_{app} \end{aligned} \tag{26}$$

Using expressions (21) and (26), the approximation for $c_{P_2}^2$ can then be written as:

$$c_{P_{2,app}}^2 = \frac{s_{P_{2,app}}^2}{[E[P_{2,app}]^2]} \tag{27}$$

3.2.3 Resulting approximation for $E[W_{f,2}]$

Expressions (13) and (27) can now be plugged into expression (3) for evaluation of $E[W_{f,2}]$. The utilization rate of stage 2 can be calculated in a straightforward manner from[1]:

$$\rho_2 = \frac{E(SU_2) + N * E(x_2)}{E(Y_{PB,2})}$$

4. Performance of the embedded queueing model

In this section, we will test the performance of the embedded queueing model in estimating $E[F_{PB}]$ versus results from discrete-event simulation. To this end, we study two settings: in the first setting (Case 1), the second stage has a considerably higher processing rate than the first stage, whereas in the second setting (Case 2), the two stages are perfectly balanced. Consequently, the gap times occurring in Case 2 are solely due to the variability present in the system.

4.1 Simulation experiment

Table 1 gives an overview of the input parameters for the two cases. Both cases where evaluated at three different utilization rates for stage 1: $\rho_{M1} = 30\%$, 60% and 90%. The random numbers for interarrival times, setup and processing times in the simulation experiment were all drawn from gamma distributions. This distribution was chosen because of its positive skewness, which is a desirable characteristic for the individual time components. Other suitable candidates would have been a lognormal distribution, or a beta($\alpha 1$, $\alpha 2$) distribution with $\alpha 1 < \alpha 2$. The runlength of the simulation was equal to 100 000 process batches, of which the first 20 000 were considered to be part of the warm-up period.

4.2 Results

Table 2 compares the resulting estimates for $E[F_{PB}]$ for both Case 1 and Case 2, showing the percentage deviation of the queueing model ($E[F_{PB}]_Q$) with respect to the simulation results ($E[F_{PB}]_S$).

From the results, it appears that the embedded queueing model performs quite well: the percentage deviations with respect to simulation are similar for the lot-for-lot policy ($T = 1$) and the different

[1]Note that the average total gap time on the second server ($E[G_2]$) need not be included in the utilization rate, as it does not constitute a part of the time busy on server 2.

	Case 1	*Case* 2
N	30	30
T	1,2,3,5,6,10,15,30	1,2,3,5,6,10,15,30
$E[SU_1]$	2	2
$E[SU_2]$	2	2
$E[x_1]$	1	1
$\frac{E[x_2]}{E[x_1]}$	0.2	1
ρ_1	30%, 60% or 90%	30%, 60% or 90%
$c^2_{SU_1}$	0.75	0.75
$c^2_{SU_2}$	0.75	0.75
$c^2_{x_1}$	0.5	0.5
$c^2_{x_1}$	0.5	0.5
$c^2_{Y_{PB,1}}$	0.75	0.75

Table 1: Input parameters for Case 1 and Case 2

		Case 1			Case 2		
T		$\rho_1 = 30\%$	$\rho_1 = 60\%$	$\rho_1 = 90\%$	$\rho_1 = 30\%$	$\rho_1 = 60\%$	$\rho_1 = 90\%$
1	$E[F_{PB}]_Q$	44.7230	57.8358	149.8667	72.5929	91.6654	192.9906
	$E[F_{PB}]_S$	**44.2365**	**56.7900**	**142.4430**	**68.9152**	**83.8548**	**185.3986**
	% *dev*	(1.0998 %)	(1.8416 %)	(5.2117 %)	(5.3367 %)	(9.3145 %)	(4.0949 %)
2	$E[F_{PB}]_Q$	39.7231	52.8342	144.8678	58.4557	77.4342	178.5937
	$E[F_{PB}]_S$	**39.2478**	**51.8012**	**137.4543**	**54.7106**	**69.5224**	**170.8449**
	% *dev*	(1.2112 %)	(1.9941 %)	(5.3934 %)	(6.8452 %)	(11.3803 %)	(4.5355 %)
3	$E[F_{PB}]_Q$	38.7144	51.8220	143.8667	53.6230	72.5008	173.1278
	$E[F_{PB}]_S$	**38.2441**	**50.7976**	**136.4506**	**50.0651**	**64.8216**	**166.0559**
	% *dev*	(1.2296 %)	(2.0167 %)	(5.4350 %)	(7.1064 %)	(11.8467 %)	(4.2588 %)
5	$E[F_{PB}]_Q$	37.9134	51.0204	143.0666	49.7602	68.5973	169.0724
	$E[F_{PB}]_S$	**37.4427**	**49.9962**	**135.6492**	**46.4214**	**61.1331**	**162.3155**
	% *dev*	(1.2571 %)	(2.0486 %)	(5.4681 %)	(7.1924 %)	(12.2098 %)	(4.1628 %)
6	$E[F_{PB}]_Q$	37.7129	50.8196	142.8666	48.7898	67.6086	167.9903
	$E[F_{PB}]_S$	**37.2431**	**49.7965**	**135.4496**	**45.5275**	**60.2357**	**161.3738**
	% *dev*	(1.2616 %)	(2.0546 %)	(5.4759 %)	(7.1655 %)	(12.2401 %)	(4.1001 %)
10	$E[F_{PB}]_Q$	37.3127	50.4192	142.4666	46.8558	65.6592	165.9988
	$E[F_{PB}]_S$	**36.8451**	**49.3985**	**135.0516**	**43.7802**	**58.4658**	**159.5666**
	% *dev*	(1.2691 %)	(2.0662 %)	(5.4905 %)	(7.0251 %)	(12.3036 %)	(4.0310 %)
15	$E[F_{PB}]_Q$	37.1124	50.2188	142.2666	45.8864	64.6777	164.9665
	$E[F_{PB}]_S$	**36.6472**	**49.2006**	**134.8537**	**42.9326**	**57.6051**	**158.6767**
	% *dev*	(1.2696 %)	(2.0694 %)	(5.4970 %)	(6.8799 %)	(12.2776 %)	(3.9639 %)
30	$E[F_{PB}]_Q$	36.9123	50.0185	142.0666	44.9176	63.6998	163.9580
	$E[F_{PB}]_S$	**36.4568**	**49.0102**	**134.6633**	**42.1144**	**56.7741**	**157.8308**
	% *dev*	(1.2495 %)	(2.0573 %)	(5.4977 %)	(6.6563 %)	(12.1986 %)	(3.8821 %)

Table 2: Results for $E(F_{PB})_Q$ and $E(F_{PB})_S$, for Case 1 and Case 2

lot splitting policies ($T > 1$). The performance is in general slightly worse for Case 2. This is not surprising: as the gap times in balanced lines are only due to the presence of variability, the approximation for $c^2_{P_2}$ tends to yield larger errors when compared to simulation results (see Van Nieuwenhuyse 2004, Van Nieuwenhuyse and Vandaele 2005). The overall performance of the model is nevertheless still satisfactory.

As shown in Figure 3, the curves for $E[F_{PB}]_Q$ and $E[F_{PB}]_S$ are convex and continuously decreasing in the number of sublots, for both cases. This convex behavior gives a graphical illustration of the impact of overlapping operations, and confirms the general intuition that lot splitting has a positive impact on average flow times, at least in a setting that consists of capacity servers. In settings that also contain delay servers, it should be noted that excessive use of lot splitting may lead to capacity problems on these servers, as their utilization increases with the number of sublots used. Hence, in these settings, the benefits of the overlapping operations effect may be countered by increasing queueing times in front of the delay servers.

5. Impact of lot splitting on total costs in a system with capacity servers

In a two-stage stochastic flowshop with capacity servers, the use of lot splitting may or may not be desirable depending on the relative importance of the inventory holding costs versus the gap costs. Indeed, gaps may represent a cost for the production 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 this section, we study the trade-off between these two cost types in more detail, and illustrate it by means of an example.

5.1 Cost model and insights

Focusing on inventory holding costs and gap costs, the average total costs in a two-stage system can be written as:

$$E[Cost_{tot}] = E[Cost_{Inv}] + E[Cost_{gap}]$$

From Little's law (e.g. Kleinrock 1975), we know that the average work-in-process inventory in a system is related to the average flow time through the system. Hence, in the stochastic system, the average number of process batches ($E[WIP_{PB}]$) will be equal to the throughput rate of process batches ($\lambda_{PB,1}$) times the average process batch flow time ($E[F_{PB}]$). This will be the case both

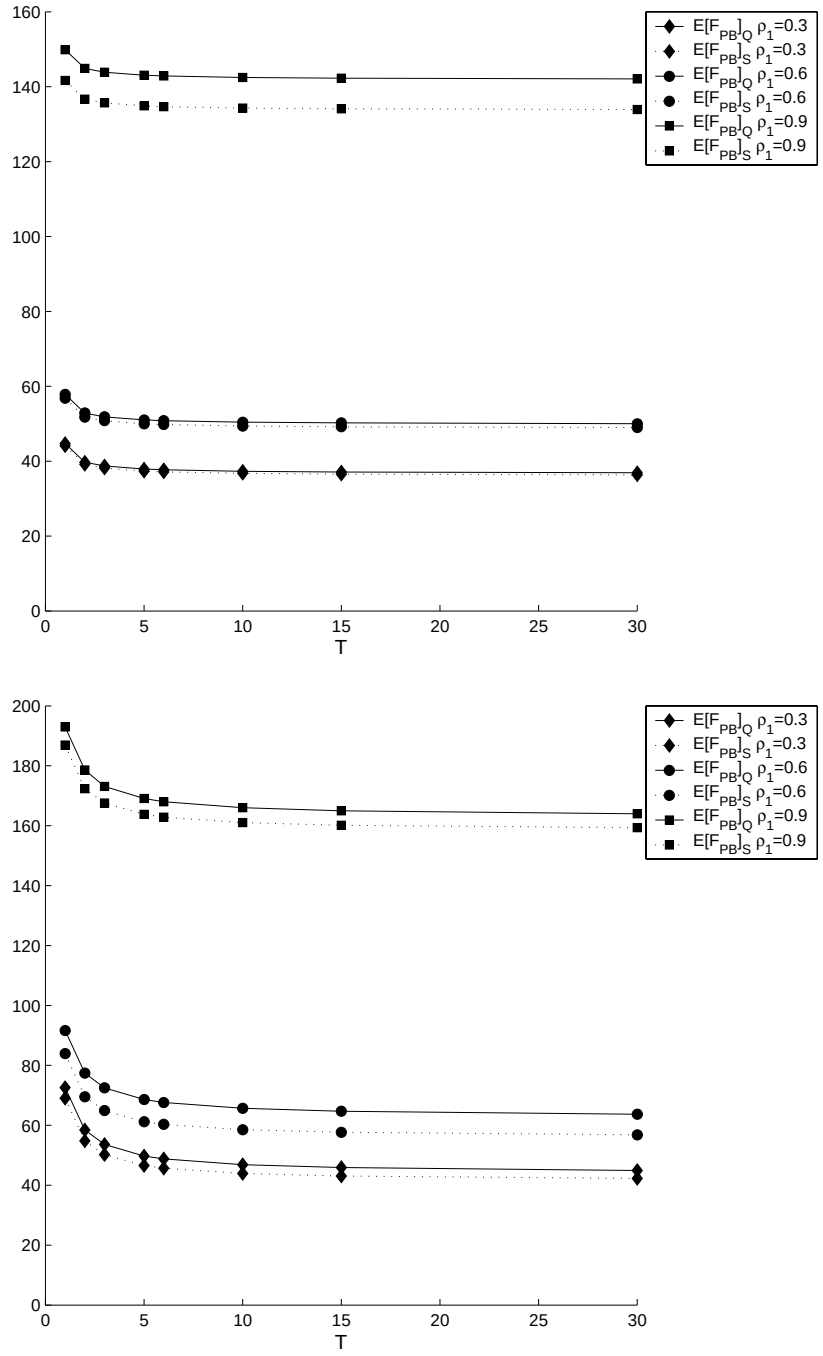

Figure 3: $E[F_{PB}]_Q$ and $E[F_{PB}]_S$ in terms of T, for Case 1 (top pane), and Case 2 (bottom pane)

for a lot-for-lot policy and for a lot splitting policy:

$$\begin{aligned} E[WIP_{PB}]_{LFL} &= \lambda_{PB,1} * E[F_{PB}]_{LFL} \\ E[WIP_{PB}]_{LS} &= \lambda_{PB,1} * E[F_{PB}]_{LS} \end{aligned}$$

The average inventory holding costs ($E[Cost_{Inv}]$) per production cycle can then be calculated by multiplying this average work-in-process inventory by a specified holding cost per process batch per cycle, c_{Inv} (> 0):

$$\begin{aligned} E[Cost_{Inv}]_{LFL} &= c_{Inv} * \lambda_{PB,1} * E[F_{PB}]_{LFL} \\ E[Cost_{Inv}]_{LS} &= c_{Inv} * \lambda_{PB,1} * E[F_{PB}]_{LS} \end{aligned}$$

As, in a two-stage system with capacity servers, $E[F_{PB}]$ is convex and continuously decreasing in T, we can conclude that the following is valid:

$$\begin{aligned} E[WIP_{PB}]_{LFL} &> E[WIP_{PB}]_{LS} \\ E[Cost_{Inv}]_{LFL} &> E[Cost_{Inv}]_{LS} \end{aligned}$$

The average inventory holding cost will be convex and continuously decreasing in T.

The average gap costs per production cycle can be determined by multiplying the average gap time on the second stage ($E[G_2]$) by a fixed cost c_{gap} (> 0):

$$E[Cost_{gap}] = c_{gap} * E[G_2]$$

For the lot-for-lot policy ($T = 1$), the average gap costs are obviously equal to zero. It can be proven that $E[G_2]$ is concave and continuously increasing in T when $E(x_1) > E(x_2)$ (see e.g. Van Nieuwenhuyse and Vandaele 2005). When $E(x_1) < E(x_2)$, the reverse may occur: $E(G_2)$ may be decreasing in T.

Based upon the discussion above, we can conclude that the lot splitting policy will be preferable to a lot-for-lot policy from a cost viewpoint when:

$$E[Cost_{Inv}]_{LFL} > E[Cost_{Inv}]_{LS} + E[Cost_{gap}]_{LS}$$

or, in other words, when the difference in inventory costs is not compensated by the occurrence of the gap costs:

$$c_{Inv} * \lambda_{PB,1} * (E[F_{PB}]_{LFL} - E[F_{PB}]_{LS}) > c_{gap} * E[G_2]$$

This will depend on the relative importance of c_{Inv} and c_{gap}.

5.2 Example

As an illustration, we apply the model presented above to an unbalanced flowshop, for which the parameters are given in Table 3. The arrival rate of process batches $\lambda_{PB,1}$ was chosen such that $\rho_1 = 90\%$. The inventory holding cost c_{Inv} is equal to 10; the fixed gap cost c_{gap} can either be low (0.1), intermediate (0.15) or high (3).

	$m = 1$	$m = 2$
$E[x_m]$	0.5	0.1
$c^2_{x_m}$	1	1
$E[SU_m]$	2	2
$c^2_{SU_m}$	0.75	0.75
N	30	30

Table 3: Input parameters for the example

Table 4 gives an overview of the resulting total costs for the three cases, in terms of the lot splitting policy T. The optimal total costs are given in bold.

	$c_{gap} = 0.1$	$c_{gap} = 0.15$	$c_{gap} = 3$
$T = 1$	43.3090	43.3090	**43.3090**
$T = 2$	41.9019	42.1055	53.7138
$T = 3$	41.8010	**42.1019**	59.2554
$T = 5$	41.7412	42.1215	63.7989
$T = 6$	41.7273	42.1275	64.9406
$T = 10$	41.7002	42.1403	67.2276
$T = 15$	41.6868	42.1469	68.3724
$T = 30$	**41.6735**	42.1536	69.5177

Table 4: Average total costs for the example, when $c_{gap} = 0.1$,$c_{gap} = 0.15$ and $c_{gap} = 3$

As Table 4 shows, the optimal lot splitting policy from a cost perspective depends heavily on the relative importance of the two components, i.e. the gap costs versus the inventory holding costs. When the gap costs are very low ($c_{gap} = 0.1$), the shape of the total cost curve is entirely dominated by the shape of the average inventory holding costs; i.e., it is convex and continuously decreasing. From a cost viewpoint, this implies that it is optimal to split the process batch in the maximum number of transfer batches ($T_{opt} = N$). Conversely, when the gap costs are very high ($c_{gap} = 3$), the shape of the total cost curve is entirely dominated by the shape of the average gap costs; i.e., concave and continuously increasing as in our case $E[x_1] > E[x_2]$. From a cost viewpoint, it is now optimal to use a lot-for-lot policy ($T_{opt} = 1$).

Situations may also arise in which neither of the two components is clearly dominant at all values

of T. In that case, the optimal lot splitting policy will be located somewhere in between the two extremes. For example, when $c_{gap} = 0.15$, the optimum is located at $T_{opt} = 3$.

6. Conclusions

In this paper, we have studied how the queueing methodology can be applied to estimate average process batch flow times in a stochastic environment with lot splitting. The discussion has revealed that we have to resort to an embedded queueing model, as the use of lot splitting entails characteristics which prevent the direct application of queueing models to our type of system. The objective of the embedded queueing model is to yield a good approximation of the average flow time of the flags; from the average flow time of the flag, we can then derive the average flow time of a process batch.

The use of the embedded queueing model was illustrated by means of two cases. The performance of the model turns out to be very good when the two stages are not perfectly balanced. For balanced systems, the performance is slightly worse but still satisfactory. The outcome of the model confirms the general intuition that lot splitting has a positive impact on average flow time in systems with capacity servers.

It is known that the use of lot splitting in a stochastic system automatically introduces the risk of gap times occurring. The presented model can be used as a basis for cost/benefit analysis: hence, besides yielding good flow time approximations, it can provide added value in studying the trade-off between flow time improvement and gap time occurrence.

Though the setting studied here is limited to two stages and one product type, we believe this work provides an important first contribution, by presenting a methodology for studying the impact of lot splitting in stochastic systems. The extension of the model towards more general settings, such as systems with multiple product types and job shop settings, may require additional modifications. Given the frequent use of lot splitting in real-life settings, these extensions undoubtedly provide an interesting and challenging topic for further research.

Acknowledgments

This research was supported by the Research Foundation Flanders (Belgium). Dr. Van Nieuwenhuyse is currently a Postdoctoral Fellow of the Research Foundation Flanders.

References

Baker, K.R., D. Jia. 1993. A comparative study of lot streaming techniques. *Omega* **21** 561–566.

Benjaafar, S. 1996. On production batches, transfer batches, and lead times. *IIE Transactions* **28** 357–362.

Blumenfeld, D. 2001. *Operations research calculations handbook*. CRC Press, Boca Raton.

Bozer, Y.A., J. Kim. 1996. Determining transfer batch sizes in trip-based material handling systems. *The International Journal of Flexible Manufacturing Systems* **8** 313–356.

Bukchin, J., M. Tzur, M. Jaffe. 2002. Lot splitting to minimize average flow time in a two-machine flow shop. *IIE Transactions* **34** 953–970.

Buzacott, J.A., J.G. Shantikumar. 1985. On approximate queueing models of dynamic job shops. *Management Science* **31** 870–887.

Chen, J., G. Steiner. 1998. Lot streaming with attached setups in three-machine flowshops. *IIE Transactions* **30** 1075–1084.

Goldratt, E.M., J. Cox. 1984. *The Goal: a process of ongoing improvement*. North River Press, New York.

Graves, S.C., M.M. Kostreva. 1986. Overlapping operations in materials requirements planning. *Journal of Operations Management* **6** 283–294.

Hopp, W.J., M.L. Spearman, D.L. Woodruff. 1990. Practical strategies for lead time reduction. *Manufacturing Review* **3** 78–84.

Hopp, W. J., M.L. Spearman. 2000. *Factory Physics: Foundations of Manufacturing Management*. Irwin/McGraw-Hill, New York.

Jacobs, F.R., D.J. Bragg. 1988. Repetitive lots: flow-time reductions through sequencing and dynamic batch sizing. *Decision Sciences* **19** 281–294.

Karmarkar, U., S. Kekre, S. Freeman. 1985a. Lot sizing and lead time performance in a manufacturing cell. *Interfaces* **15** 1–9.

Karmarkar, U., S. Kekre, S. Kekre. 1985b. Lot sizing in multi-item multi-machine job shops. *IIE Transactions* **17** 290–298.

Karmarkar, U. 1987. Lot sizes, lead times and in-process inventories. *Management Science* **33** 409–417.

Kleinrock, L. 1975. *Queueing systems*. John Wiley & Sons, New York.

Kramer, W., M. Lagenbach-Belz. 1976. Approximate formulae for the delay in the queueing system GI/GI/1. *Congressbook of the Eight International Teletraffic Congress, Melbourne*, Melbourne. 2351–2358.

Kropp, D.H., T.L. Smunt. 1990. Optimal and heuristic models for lot splitting in a flow shop. *Decision Sciences* **21** 691–709.

Lambrecht, M.R., P.L. Ivens, N.J. Vandaele. 1998. ACLIPS: A capacity and lead time integrated procedure for scheduling. *Management Science* **44** 1548–1561.

Litchfield, J.L., R. Narasimhan. 2000. Improving job shop performance through process queue management under transfer batching. *Production and Operations Management* **9** 336–348.

Marshall, K.T. 1968. Some inequalities in queueing. *Operations Research* **16** 651–665.

Potts, C.N., M.Y. Kovalyov. 2000. Scheduling with batching: a review. *European Journal of Operational Research* **120** 228–249.

Ramasesh, R.V., H. Fu, D. K. H. Fong, J. C. Hayya. 2000. Lot streaming in multistage production systems. *International Journal of Production Economics* **66** 199–211.

Ruben, R.A., F. Mahmoodi. 1998. Lot splitting in unbalanced production systems. *Decision Sciences* **29** 921-949.

Santos, C., M. Magazine. 1985. Batching in single-operations manufacturing systems. *Operations Research Letters* **4** 99–103.

Smunt, T.L., A.H. Buss, D.H. Kropp. 1996. Lot Splitting in stochastic flow shop and job shop environments. *Decision Sciences* **27** 215–238.

Suri, R., J.L. Sanders, M. Kamath. 1993. Performance evaluation of production networks. S. C. Graves et al., eds. *Logistics of Production and Inventory, Handbooks in Operations Research and Management Science*. North-Holland. 199–286.

Suri, R. 1998. *Quick Response Manufacturing: a companywide approach to reducing lead times*. Productivity Press, Portland, OR.

Umble, M., M.L. Srikanth. 1995. *Synchronous Manufacturing: principles for world-class excellence*. The Spectrum Publishing Company.

Van Nieuwenhuyse, I. 2004. Lot splitting in single-product flowshops: issues of delivery reliability, production disruptions and flow times. PhD thesis, Department of Applied Economics,

University of Antwerp, Antwerp, Belgium.

Van Nieuwenhuyse, I., N. Vandaele. 2004a. Determining the optimal number of sublots in a single-product, deterministic flow shop with overlapping operations. *International Journal of Production Economics* **92** 221–239.

Van Nieuwenhuyse, I., N. Vandaele. 2004b. The impact of delivery frequency on delivery reliability in a two-stage supply chain. *Proceedings of the 13th International Workshop on Production Economics, Igls (Austria)* **3** 367–392.

Van Nieuwenhuyse, I., N. Vandaele. 2005. Analysis of gap times in a two-stage stochastic flow-shop with overlapping operations. Department of Applied Economics, University of Antwerp, Research paper 2005-004.

Wagner, B.J., G.L. Ragatz. 1994. The impact of lot splitting on due date performance. *Journal of Operations Management* **12** 13–25.

Whitt, W. 1983. The queueing network analyzer. *The Bell System Technical Journal* **62** 2779–2815.

Whitt, W. 1993. Approximations for the GI/G/m queue. *Production and Operations Management* **2** 114–161.

Whitt, W. 1994. Towards better multi-class parametric-decomposition approximations for open queueing networks. *Annals of Operations Research* **48** 221–248.