Performance analysis of assembly systems

PERFORMANCE ANALYSIS OF ASSEMBLY SYSTEMS[*]

MARCEL VAN VUUREN[†] AND IVO J. B. F. ADAN[‡]

Abstract. In this paper we present an approximation for the performance analysis of assembly systems with finite buffers and generally distributed service times. The approximation is based on decomposition of the assembly system in subsystems. Each subsystem can be described by a finite-state quasi-birth-and-death process, the parameters of which are determined by an iterative algorithm. Numerical results show that the approximation accurately predicts performance characteristics such as throughput and mean sojourn time.

Key words. assembly system, approximation, decomposition, finite buffer, matrix analytic method

AMS subject classifications. 60K25, 68M20

1. Introduction. Queueing networks with finite buffers have been studied extensively in the literature; see, e.g., [3] and [9]. These models have many applications in manufacturing, communication and computer systems. Usually, it is assumed that these networks consist of single-server or multi-server nodes. But, in manufacturing systems, it often occurs that different parts arrive at a node (machine), where they are assembled into one product. The performance analysis of assembly nodes is much more complicated, and did not receive much attention in the literature. In this paper we study an assembly node in isolation, with general service times, finite buffers and blocking after service (BAS), and we propose a method for the approximative performance analysis. We are interested in the steady-state queue-length distribution of each buffer; these distributions may be used to determine performance characteristics, such as the throughput and mean sojourn time.

We consider a queueing system (denoted by L; see Fig. 1.1) assembling n parts into one product. The parts are labeled $1, \ldots, n$. The arrival processes of parts are modeled as follows. Type i parts are generated by a so-called arrival server, denoted by M_i, $i = 1, \ldots, n$. For example, in manufacturing systems, arrival server M_i may typically represent the upstream production line producing type i parts. Arrival server M_i serves one part at a time and is never starved (i.e., there is always a new part available). The generic random variable S_i denotes the service (or inter-arrival) time of server M_i; S_i is generally distributed with rate μ_i and coefficient of variation c_i. After service completion at M_i, type i parts are put in buffer B_i, where they wait for assembly. The size of buffer B_i is b_i. Server M_i operates according to the BAS blocking protocol: if upon service completion, buffer B_i is full, then server M_i becomes blocked and the finished part waits until space becomes available in buffer B_i. The parts in the buffers $B_1, \ldots, B_n$ are assembled into one product by (assembly) server M_a. The assembly can start as soon as a part of each type is available. If some are not available yet, the other ones can wait in the assembly server (i.e., they are removed from the buffer). The generic random variable S_a denotes the assembly time of server M_a; S_a is generally distributed with rate μ_a and coefficient of variation c_a.

The method, proposed in this paper, to approximate the steady-state queue-length distribution of the buffers is based on decomposition of the assembly system into subsystems. Each buffer is considered in isolation, and the interaction with other buffers is incorporated in the service time: it consists of a so-called wait-to-assembly time and the actual assembly time. The wait-to-assembly time reflects that a part may have to wait for other parts to arrive, and the parameters of the wait-to-assembly time (such as the first two moments) are determined by an iterative algorithm. In this algorithm, the inter-arrival times and service times are approximated by fitting simple phase type distribution on the first two moments; then each buffer can be described by a finite-state quasi-birth-and-death process (QBD), the steady-state distribution of which can be efficiently determined by matrix-analytic techniques.

Assembly queueing systems have been studied by several authors. Hemachandra and Eedupuganti [6] look at a fork-join queue in an open system. Rao and Suri [2] and Krishnamurti et al. [7] also treat a fork-

[*]This research is supported by the Technology Foundation STW, applied science division of NWO and the technology programme of the Dutch Ministry of Economic Affairs.

[†]Department of Mathematics and Computer Science, University of Technology Eindhoven, P.O. Box 513, 5600 MB, Eindhoven, The Netherlands (m.v.vuuren@tue.nl).

[‡]Department of Mathematics and Computer Science, University of Technology Eindhoven, P.O. Box 513, 5600 MB, Eindhoven, The Netherlands (i.j.b.f.adan@tue.nl).

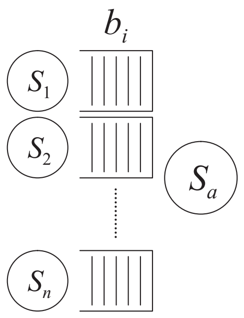

FIG. 1.1. *A schematic representation of an assembly system.*

join queue, but then as a closed system. These references develop approximations. An exact analysis of an assembly system is presented by Gold [5]. None of these references, however, consider general inter-arrival and assembly times, and some of them only look at assembly systems for two parts.

The paper is organized as follows. In Section 2 we explain the decomposition of the assembly system in subsystems. In the section thereafter we take a close look at the subsystems. Section 4 describes the iterative algorithm. Numerical results are presented in Section 5. Finally, Section 6 contains some concluding remarks.

2. Decomposition of the assembly system. We decompose the original assembly system L into n subsystems $L_1, L_2, \ldots, L_n$. Subsystem L_i describes the processing of type i parts in isolation; it consists of a finite buffer of size b_i, arrival-server M_i in front of the buffer, and a so-called departure-server D_i behind the buffer. In Figure 2.1 we show the decomposition of assembly system L in Figure 1.1.

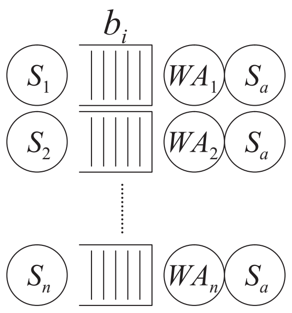

FIG. 2.1. *Decomposition of the assembly system in Figure 1.1.*

The service time of departure-server D_i consists of two components. The random variable WA_i denotes the wait-to-assembly time in subsystem L_i, $i = 1, \ldots, n$. This random variable WA_i represents the time that elapses from the departure of an assembled product until the moment that all parts $j \neq i$ are available for assembly; part i is excluded, because its presence is explicitly modeled by the subsystem. Note the clock for WA_i starts to tick immediately after a departure, irrespective of whether there is a part in buffer B_i or not; also, WA_i maybe equal to zero, namely when the buffers B_j, $j \neq i$, are nonempty just after departure. An important (approximation) assumption is that the successive wait-to-assembly times in subsystem L_i are *independent and identically distributed*. Thus, if it takes A time units for the next part i to become available, then the next assembly can start after $\max\{A, WA_i\}$ time units. The second part of the service time of departure server D_i is the assembly time S_a itself.

In the next section we elaborate further on the subsystems.

3. The subsystems. In this section we describe how the wait-to-assembly times of subsystem L_i are determined, and subsequently, how the steady-state queue length distribution of subsystem L_i can be found by employing matrix-analytic techniques. Crucial to the analysis is that the distributions of the random variables involved are represented by simple phase-type distributions matching the first two moments. Below we first explain which phase-type distributions will be used.

3.1. Two moment fit. We will model the distribution of a *positive* random variable with rate λ and coefficient of variation c as a mixed Erlang distribution with equal rates of the exponential phases if $c^2 \leq 1$, and otherwise, as a Hyperexponential distribution (see, e.g., [11]).

More specifically, if $1/k \leq c^2 \leq 1/(k-1)$ for some $k = 2, 3, \ldots$, then the rate and squared coefficient of variation of the Erlang$_{k-1,k}$ distribution with density

$$f(t) = p\mu^{k-1}\frac{t^{k-2}}{(k-2)!}e^{-\mu t} + (1-p)\mu^k\frac{t^{k-1}}{(k-1)!}e^{-\mu t}, \qquad t \geq 0,$$

matches with λ and c^2, provided the parameters p and μ are chosen as

$$p = \frac{1}{1+c^2}[kc^2 - \{k(1+c^2) - k^2c^2\}^{1/2}], \qquad \mu = (k-p)\lambda.$$

If $c^2 > 1$, then the mean and squared coefficient of variation of the Hyperexponential distribution with density

$$f(t) = p\mu_1 e^{-\mu_1 t} + (1-p)\mu_2 e^{-\mu_2 t}, \qquad t \geq 0$$

matches with λ and c^2, provided the parameters p, μ_1 and μ_2 are chosen as

$$p = \frac{1}{2}\left(1 + \sqrt{\frac{c^2-1}{c^2+1}}\right), \qquad \mu_1 = 2p\lambda, \qquad \mu_2 = 2(1-p)\lambda.$$

Also other parameter choices or other distributions (like the Coxian distribution) may be used to match the first two moments, but numerical experiments suggest that the quality of the approximation is fairly insensitive to the choice of (phase-type) distributions.

3.2. The wait-to-assembly time. As said before, the wait-to-assembly time at subsystem L_i is the time that elapses from the departure of an assembled product in subsystem L_i until the moment that all parts $j \neq i$ are available at the assembly server. So, WA_i is the maximum of the residual inter-arrival times RA_j of the parts $j \neq i$; note that the residual inter-arrival time RA_j is equal to 0 when buffer B_j is nonempty just after the departure of the assembled product (i.e., the next part j is immediately available). So, we have that

$$WA_i = \max_{j \neq i} RA_j.$$

Below we determine the first two moments of WA_i and its probability mass at 0.

Denote by $p_{e,j}$ the probability that buffer B_j is empty just after the departure of an assembled product; so RA_j is positive with probability $p_{e,j}$ (and zero otherwise). In Section 3.4 we will determine, for each subsystem L_j, the probability $p_{e,j}$ and the first two moments of the *conditional* RA_j, i.e., RA_j given that it is positive. Then, by adopting the approximation assumption that the random variables RA_j are *independent*, we have the ingredients to recursively compute the first two moments of WA_i and its mass at 0. The computation is based on the following relation for the maximum of the first k residual inter-arrival times:

$$\max_{1 \leq j \leq k} RA_j = \max\{RA_k, \max_{1 \leq j \leq k-1} RA_j\}. \tag{3.1}$$

Hence, once the first two moments of the first $k-1$ residual inter-arrival times are known, we first condition on whether the two random variables RA_k and $\max_{1 \leq j \leq k-1} RA_j$ are positive or not; note that, by the independence assumption,

$$P(\max_{1 \leq j \leq k-1} RA_j = 0) = \prod_{1 \leq j \leq k-1} (1 - p_{e,j}).$$

Then we fit phase-type distributions on the first two moments of the conditional random variables (according to the recipe of Section 3.1) to compute the first two moments of their maximum. The exact computation of the maximum of two independent phase-type distributed random variables is presented in the next section.

Thus, by repeated application of (3.1), we can compute the first two moments of WA_i, and its probability mass at 0, denoted by $p_{ne,i}$, which immediately follows from the assumption that the random variables RA_j are independent, yielding

$$p_{ne,i} = \prod_{j \neq i} (1 - p_{e,j}).$$

A representation of WA_i is shown in Figure 3.1, where WAC_i is the conditional wait-to-assembly time. The distribution of WAC_i is approximated by a phase-type distribution, matching its first two moments,

$$\mathbb{E}(WAC_i) = \frac{\mathbb{E}(WA_i)}{1 - p_{ne,i}}, \tag{3.2}$$

$$\mathbb{E}(WAC_i^2) = \frac{\mathbb{E}(WA_i^2)}{1 - p_{ne,i}}. \tag{3.3}$$

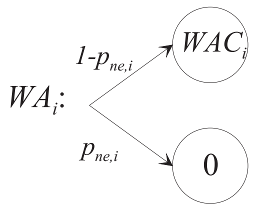

FIG. 3.1. *The wait-to-assembly time of subsystem L_i.*

3.3. The maximum of two phase-type random variables. In this section we calculate the first two moments of the maximum of two independent Erlang distributed random variables. Let E_i denote an Erlang$_{k_i}$ distributed random variable with scale parameter μ_i, $i = 1, 2$, and assume that E_1 and E_2 are independent. The maximum of E_1 and E_2 is phase-type distributed, the first (random) number of exponential phase have rate $\mu_1 + \mu_2$. These phases are followed by a (random) number of exponential phases with rate μ_1 or rate μ_2, depending on which of the random variables E_1 and E_2 finishes first. Let $q_{1,j}$ with $0 \leq j \leq k_2 - 1$ be the probability that E_2 has completed j phases when E_1 completes its final phase, and similarly, let $q_{2,i}$ with $0 \leq i \leq k_1 - 1$ be the probability that E_1 has completed i phases when E_2 completes its final phase. It is easily verified that $q_{1,j}$ and $q_{2,i}$ both follow a Negative Binomial distribution, i.e.,

$$q_{1,j} = \binom{k_1 - 1 + j}{k_1 - 1} \left(\frac{\mu_2}{\mu_1 + \mu_2}\right)^j \left(\frac{\mu_1}{\mu_1 + \mu_2}\right)^{k_1}, \quad 0 \leq j \leq k_2 - 1,$$

$$q_{2,i} = \binom{k_2 - 1 + i}{k_2 - 1} \left(\frac{\mu_1}{\mu_1 + \mu_2}\right)^i \left(\frac{\mu_2}{\mu_1 + \mu_2}\right)^{k_2}, \quad 0 \leq i \leq k_1 - 1.$$

Conditioned on the event that E_1 finishes first and E_2 has then completed j phases, the maximum of E_1 and E_2 is Erlang distributed with $k_1 + k_2$ phases, the first $k_1 + j$ of which have rate $\mu_1 + \mu_2$ and the last $k_2 - j$ have rate μ_2. Let $M_{1,j}$ denote this conditional maximum, then

$$\mathbb{E}M_{1,j} = \frac{k_1 + j}{\mu_1 + \mu_2} + \frac{k_2 - j}{\mu_2},$$

$$\mathbb{E}M_{1,j}^2 = \frac{(k_1 + j)(k_1 + j + 1)}{(\mu_1 + \mu_2)^2} + \frac{(k_1 + j)(k_2 - j)}{(\mu_1 + \mu_2)\mu_2} + \frac{(k_2 - j)(k_2 - j + 1)}{\mu_2^2}.$$

Similarly, let $M_{2,i}$ denote the maximum of E_1 and E_2, conditioned on the event that E_2 finishes first and E_1 has then completed i phases. For the first two moments of $M_{2,i}$ we have

$$\begin{aligned}
\mathbb{E}M_{2,i} &= \frac{k_2+i}{\mu_1+\mu_2} + \frac{k_1-i}{\mu_1}, \\
\mathbb{E}M_{2,i}^2 &= \frac{(k_1+j)(k_1+j+1)}{(\mu_1+\mu_2)^2} + \frac{(k_1+j)(k_2-j)}{(\mu_1+\mu_2)\mu_2} + \frac{(k_2-j)(k_2-j+1)}{\mu_2^2}.
\end{aligned}$$

The first two moments of the maximum of E_1 and E_2 can now easily be computed by conditioning on the above events, yielding

$$\begin{aligned}
\mathbb{E}(\max\{E_1,E_2\}) &= \sum_{j=0}^{k_2-1} q_{1,j}\mathbb{E}M_{1,j} + \sum_{i=0}^{k_1-1} q_{2,i}\mathbb{E}M_{2,i}, \\
\mathbb{E}(\max\{E_1,E_2\}^2) &= \sum_{j=0}^{k_2-1} q_{1,j}\mathbb{E}M_{1,j}^2 + \sum_{i=0}^{k_1-1} q_{2,i}\mathbb{E}M_{2,i}^2.
\end{aligned}$$

Note that, if E_1 and E_2 are both probabilistic mixtures of Erlang random variables, then the first two moments of the maximum of E_1 and E_2 can be computed from the above equations by conditioning on the composition of the maximum.

3.4. Subsystem analysis. In this subsection we analyze substem L_j (and in the remainder of this section we drop the subscript j). The conditional wait-to-assembly time WAC in the subsystem is represented by a phase-type random variable, the first two moments of which match (3.2) and (3.3). Further, we also fit simple phase-type distributions (according to the recipe in Section 3.1) on the first two moments of the service time S of the arrival server, the assembly time S_a. In doing so, the subsystem can be described by a finite state Markov process, with states (i,j,k). The state variable i denotes the total number of parts in the subsystem (at the assembly server or waiting in the buffer). Thus, i is at least 0 and at most $b+1$. The state variable j indicates the phase of the service time S of the arrival server, and k indicates the phase of the residence time $WA+S_a$ at the assembly server.

To define the generator of the Markov process we use the Kronecker product: If A is an $n_1 \times n_2$ matrix and B is an $n_3 \times n_4$ matrix, the Kronecker product $A \otimes B$ is defined by

$$A \otimes B = \begin{pmatrix} A(1,1)B & \cdots & A(1,n_2)B \\ \vdots & & \vdots \\ A(n_1,1)B & \cdots & A(n_1,n_2)B \end{pmatrix}.$$

By ordering the states lexicographically and partitioning the state space into levels $0,1,\dots,b+1$, where level i is the set of all states with i customers in the system, it is immediately seen that the Markov process is a QBD, the generator $\mathbf{Q}$ of which has the following form:

$$\mathbf{Q} = \begin{pmatrix} B_{00} & B_{01} & & & & \\ B_{10} & A_1 & A_0 & & & \\ & A_2 & \ddots & \ddots & & \\ & & \ddots & \ddots & A_0 & \\ & & & A_2 & A_1 & C_{10} \\ & & & & C_{01} & C_{00} \end{pmatrix}$$

Below we specify the submatrices in $\mathbf{Q}$. To describe the service processes of the arrival server and assembly server we use the concept of a Markovian Arrival Process (MAP); see [1]. In general, a MAP is defined in terms of a continuous-time Markov process with finite state space $\{0, \cdots, m-1\}$ and generator $G_0 + G_1$. The element $G_{1,(i,j)}$ denotes the intensity of transitions from i to j accompanied by an arrival, whereas for $i \neq j$ element $G_{0,(i,j)}$ denotes the intensity of the remaining transitions from i to j and the diagonal elements $G_{0,(i,i)}$ are negative and chosen such that the row sums of $G_0 + G_1$ are zero.

The service process of the arrival server can be straightforwardly represented by a MAP, the states of which correspond the phases of the service time S. Its generator can be expressed as $AR_0 + AR_1$, where the transition rates in AR_1 are the ones that do correspond to a service completion, i.e., an arrival in the buffer. Hence, in $\mathbf{Q}$, the transitions in AR_1 lead to a transition from level i to $i + 1$, whereas the ones in AR_0 correspond to transitions within level i.

The MAP for the service process of the assembly server will be described in more detail. Let us assume that the distribution of the conditional wait-to-assembly time WAC can be represented by a phase-type distribution with n_{wac} phases, numbered $1, \ldots, n_{wac}$; the rate of phase i is ν_i and p_i is the probability to proceed to phase $i+1$, and $1-p_i$ is the probability that the wait-to-assembly time is finished. Similarly, the distribution of the assembly time S_a can be represented by a phase-type distribution with n_{s_a} phases, with rates μ_i and transition probabilities q_i, $i = 1, \ldots, n_{s_a}$. Now the states of the MAP are numbered $1, \ldots, n_{wac}+n_{s_a}$, and its generator can be expressed as DE_0+DE_1, where the transition rates in DE_1 are the ones corresponding to a service completion, i.e., a departure from the system. So, in $\mathbf{Q}$, the transitions in DE_1 lead to a transition from level i to $i-1$, whereas the ones in DE_0 correspond to transitions within level i. The non-zero elements of DE_0 and DE_1 are specified below.

$$\begin{aligned}
DE_0(i,i) &= -\nu_i, && i = 1, \ldots, n_{wac}, \\
DE_0(i,i+1) &= p_i\nu_i, && i = 1, \ldots, n_{wac} - 1, \\
DE_0(i, n_{wac}+1) &= (1-p_i)\nu_i, && i = 1, \ldots, n_{wac}, \\
DE_0(i,i) &= -\mu_i, && i = n_{wac}+1, \ldots, n_{wac}+n_{s_a}, \\
DE_0(i,i+1) &= q_i\mu_i, && i = n_{wac}+1, \ldots, n_{wac}+n_{s_a}-1, \\
DE_1(i,1) &= (1-p_{ne})(1-q_i)\mu_i, && i = n_{wac}+1, \ldots, n_{wac}+n_{s_a}, \\
DE_1(i, n_{wac}+1) &= p_{ne}(1-q_i)\mu_i, && i = n_{wac}+1, \ldots, n_{wac}+n_{s_a}.
\end{aligned}$$

Now we can describe the submatrices in $\mathbf{Q}$. The transition rates from levels $1 \leq i \leq b$ are given by

$$\begin{aligned}
A_0 &= AR_1 \otimes I_{n_{wac}+n_{s_a}}, \\
A_1 &= AR_0 \otimes I_{n_{wac}+n_{s_a}} + I_{n_a} \otimes DE_0, \\
A_2 &= I_{n_a} \otimes DE_1,
\end{aligned}$$

where I_n is the identity matrix of size n.

If the subsystem is empty and the wait-to-assembly time elapsed, the assembly can not start yet (but has to wait for a part to arrive). This implies that the transition rates from level 0 are slightly different from the ones at higher levels. Therefore, we introduce the square matrix $\tilde{DE}_0$ of size $n_{wac} + n_{s_a}$. The transitions from states $1, \ldots, n_{wac}$ remain the same, but now n_{wac+1} is an absorbing state, indicating that the wait-to-assembly time has been finished. The non-zero elements of $\tilde{DE}_0$ are

$$\begin{aligned}
\widetilde{DE}_0(i,i) &= -\nu_i, && i = 1, \ldots, n_{wac}, \\
\widetilde{DE}_0(i,i+1) &= p_i\nu_i, && i = 1, \ldots, n_{wac} - 1, \\
\widetilde{DE}_0(i, n_{wac}+1) &= (1-p_i)\nu_i, && i = 1, \ldots, n_{wac}.
\end{aligned}$$

Hence, for the transition rates at level 0 we have

$$B_{01} = AR_1 \otimes I_{n_{wac}+n_{s_a}},$$
$$B_{00} = AR_0 \otimes I_{n_{wac}+n_{s_a}} + I_{n_a} \otimes \widetilde{DE}_0,$$
$$B_{10} = I_{n_a} \otimes DE_1,$$

Finally, we have

$$C_{01} = AR_1 \otimes I_{n_{wac}+n_{s_a}},$$
$$C_{00} = I_{n_a} \otimes DE_0,$$
$$C_{10} = I_{n_a} \otimes DE_1,$$

This completes the description of the QBD. The steady-state distribution can be determined by the matrix geometric method. More specifically, we use the efficient techniques developed by Latouche and Ramaswami [8], and Naoumov et al. [10]. If we denote the equilibrium probability vector of level i by p_i, then p_i has a matrix-geometric form

$$\pi_i = x_1 R^{i-1} + x_b \hat{R}^{b-i}, \qquad i = 1, \ldots, b. \tag{3.4}$$

Here, R is the minimal nonnegative solution of matrix-quadratic equation

$$A_0 + RA_1 + R^2 A_2 = 0,$$

and $\hat{R}$ is the minimal nonnegative solution of equation

$$A_2 + \hat{R}A_1 + \hat{R}^2 A_0 = 0.$$

The matrices R and $\hat{R}$ are determined by using an iterative algorithm developed by Naoumov et al. [10]. The algorithm for R is listed in Figure 3.2.

```
N := $A_1$
L := $A_0$
M := $A_2$
W := $A_1$
dif := 1

while dif > $\epsilon$
{
   X := $-N^{-1}L$
   Y := $-N^{-1}M$
   Z := LY
   dif := $\|Z\|$
   W := W + Z
   N := N + Z + MX
   Z := LX
   L := MY
   M := Z
}
R := $-A_0 W^{-1}$
```

FIG. 3.2. *Algorithm of Naoumov et al. [10] for finding the rate matrix R, where $\|.\|$ denotes a matrix-norm and ϵ some positive number.*

The final step is to determine x_1 and x_b. The balance equations at the boundary levels $0, 1, b$ and $b+1$ are given by

$$\begin{aligned}
0 &= \pi_0 B_{00} + \pi_1 B_{10}, \\
0 &= \pi_0 B_{01} + \pi_1 A_1 + \pi_2 A_2, \\
0 &= \pi_{b-1} A_0 + \pi_b A_1 + \pi_{b+1} C_{01}, \\
0 &= \pi_b C_{10} + \pi_{b+1} C_{00}.
\end{aligned}$$

Eliminating π_0 and π_{b+1} from the equations above, and then substituting the form (3.4) for π_1 and π_b yields

$$\begin{aligned}
0 &= x_1(A_1 + RA_2 - B_{10}B_{00}^{-1}B_{01}) + x_b(\hat{R}^{b-1}A_1 + \hat{R}^{b-2}A_2 - \hat{R}^{b-1}B_{10}B_{00}^{-1}B_{01}, \\
0 &= x_1(R^{b-2}A_0 + R^{b-1}A_1 - R^{b-1}C_{10}C_{00}^{-1}C_{01}) + x_b(\hat{R}A_0 + A_1 - C_{10}C_{00}^{-1}C_{01}.
\end{aligned}$$

These equations have, together with the normalization equation, a unique solution x_1 and x_b.

From the queue-length distribution we can readily derive performance measures, such as throughput, mean buffer content and mean sojourn time (where the sojourn time is the time that elapses from arrival in the buffer until service completion at the assembly server). Also, the probability p_e that the buffer is empty just after a departure and the distribution of the conditional residual inter-arrival time $RA|RA > 0$ can be obtained; namely

$$p_e = \frac{\pi_1 B_{10} e}{T},$$

where e is a vector of ones and T is the throughput. The probability vector $\frac{\pi_1 B_{10}}{\pi_1 B_{10} e}$ yields the distribution of the phase of the inter-arrival time S just after a departure leaving behind an empty buffer, and thus it can be used to determine the distribution, and the first two moments in particular, of the conditional residual inter-arrival time.

4. The iterative algorithm.

We now describe the iterative algorithm for approximating the characteristics of the assembly system L. The algorithm is based on the decomposition of L in n subsystems $L_1, L_2, \ldots, L_n$. Before going into detail in Section 4.2, we present the outline of the algorithm in Section 4.1.

4.1. Outline of the algorithm.

- Step 0: Choose initial characteristics of the wait-to-assembly time for each subsystem $L_1, \ldots, L_n$.
- Step 1: For each subsystem $L_i, i = 1, \ldots, n$: Determine $p_{ne,i}$ and the first two moments of WAC_i.
- Step 2: For each subsystem $L_i, i = 1, \ldots, n$: Determine the queue-length distribution.
- Repeat Step 1 and 2 until the characteristics of the wait-to-assembly times have converged.

4.2. Details of the algorithm.

Step 0: Initialization

The first step of the algorithm is to initially assume that the wait-to-assembly times are zero. This means that the probabilities $p_{e,j}$ are set to 0. More sophisticated initializations, allowing faster convergence, are probably possible, but the present initialization already works well.

Step 1: The wait-to-assembly times

By using the probabilities $p_{e,j}$ that buffer j is empty just after a departure and the first two moments of the conditional residual inter-arrival times (obtained from the initialization or the previous iteration), we determine for subsystem L_i the (new) first two moments of the wait-to-assembly time and its mass at 0 (as described in Section 3.2).

Step 1 is performed for each subsystem L_i, $i = 1, \ldots, n$.

Step 2: Analysis of subsystem L_i

Based on the (new) estimates for the first two moments and the mass at 0 of the wait-to-assembly time, we determine the steady-state queue length distribution of subsystem L_i, as described in Section 3.4.

Then, by using the steady-state queue length distribution, we calculate the probability $p_{e,i}$ that buffer B_i is empty just after a departure and the conditional residual inter-arrival time, as well as the performance characteristics such as throughput and mean sojourn time.

Step 2 is performed for each subsystem L_i, $i = 1, \ldots, n$.

After completion of Step 1 and 2 we check whether the iterative algorithm has converged or not. This can be done by comparing the new estimates for the probabilities $p_{e,i}$ with the ones from the previous iteration. If the sum of the absolute values of the differences between these estimates is less than ε, the algorithm stops; otherwise Step 1 and 2 are repeated.

Of course, other stop-criteria may be used as well; for example, we may consider the throughput instead of the probabilities $p_{e,i}$. Bottom line is that we go on until 'nothing' changes anymore.

Remark: In all experiments we observed that the throughput of each of the subsystem converged, and that all throughputs converged to exactly the same value. However, we have not been able to rigorously prove that all throughputs converge to the same value.

5. Numerical Results. To investigate the quality of the proposed approximation we compare, for a large number of cases, the estimates for the mean sojourn time of each part and the throughput with the ones produced by discrete-event simulation. We are especially interested in investigating under which circumstances the approximation method gives satisfying results. Each simulation run is sufficiently long such that the widths of the 95% confidence intervals of the mean sojourn time and the throughput are smaller than 1%.

We use a broad set of parameters for the tests. The average service times of the arrival servers are all 1. The number of parts in the assembly system is varied between 2, 4 and 8. All buffers have the same size, which is varied between 0, 2, 4, and 8. The average assembly time of the assembly server is varied between 0.75 and 1, and the squared coefficient of variation (SCV) of the assembly time is varied between 0.5 and 1. We consider balanced and imbalanced systems. In the balanced cases we set the service rates of the arrival servers all to 1. Also the SCV of the service times of each arrival server is the same and is varied between 0.2, 0.5, 1 and 2. We further investigate two kinds of imbalance. We test imbalance in the average service times of the arrival servers by making the first arrival server $1/3$ faster then the last one, and by letting the service rates of the arrival servers in between change linearly (such that the overall service rate is maintained at 1). For example, in case of 4 arrival servers we get service rates $(0.857, 0.952, 1.048, 1.143)$. Imbalance in the SCV of the service times of the arrival servers is tested in the same way, but now the SCV of the service time of the last server is three times the SCV of the first server and where the SCVs of the service times of the arrival servers in between change linearly (such that the average SCV over the arrival servers is equal to one of SCVs mentioned above for the balanced cases). This leads to a total of $4^2 2^4 3 = 768$ test cases. The results for each category are summarized in Table 1. Each (sub)table lists the average error in the throughput and the mean sojourn times compared with simulation results. Table 2 summarizes all cases with the average errors and for 3 error-ranges the percentage of the cases which fall in that range.

	Error in T	Error in S
Imb. SCV		
no	1.46 %	2.68 %
yes	1.49 %	2.85 %
Imb. mean		
no	1.74 %	3.01 %
yes	1.21 %	2.52 %
No. parts		
2	0.61 %	1.53 %
4	1.37 %	2.72 %
8	2.44 %	4.04 %
Buffers		
0	2.35 %	3.29 %
2	1.68 %	2.58 %
4	1.05 %	2.09 %
8	0.82 %	3.10 %
Occ. rate		
0.75	1.34 %	2.97 %
1	1.61 %	2.56 %
SCV as. ser.		
0.5	1.24 %	2.47 %
1	1.71 %	3.06 %
SCV ar. ser.		
0.2	1.99 %	4.33 %
0.5	1.52 %	2.86 %
1	1.30 %	2.22 %
2	1.09 %	1.65 %

TABLE 5.1
Results for the assembly system with one parameter fixed.

Perf. char.	Avg.	0-5 %	5-10 %	$>$ 10 %
Throughput	1.5 %	97.4 %	2.6 %	0.0 %
Mean sojourntime	2.8 %	84.9 %	13.4 %	1.7 %

TABLE 5.2
Overall results for the assembly system.

Overall we can conclude from the above results that the approximation method works very well. The average error in the throughput is around 1.5 % and the average error in the mean sojourn time is around 2.8 %.

Now let us take a look at the results in more detail. If we look at Table 1, we see that the quality of the results for the throughput and mean sojourn times are nearly insensitive to both types of imbalance, the occupation rate and the SCV of the assembly system. We see that, as expected, the errors get higher when the number of parts increases. Also, the approximation predicts the throughput best when the buffers are large, the errors in the mean sojourn times are almost insensitive for changes in the buffersizes. Finally, the approximation is slightly better when the SCV of service times of the arrival servers are higher. Note that all these results are still highly acceptable.

In Table 2 we see that the error in the throughput is almost always within 5%, which is very reliable and robust. For the mean sojourn the times the approximation gives slightly higher errors, but almost all cases are within 10%.

6. Concluding remarks. In this paper we developed an algorithm for approximating an assembly queueing system with general arrival and service times. We used a decomposition approach and developed an iterative algorithm to approximate the performance characteristics of the assembly queue. Therefore, we accurately determine the characteristics of the wait-to-assembly time at a queue. The queue-length distributions of the subsystems are determined by using a matrix geometric method.

We tested the algorithm by comparing it with a discrete-event simulation and the results are very promising. After testing many cases, we concluded that the average errors in the throughput are around 1.5% and the errors in the mean sojourn time are around 3%. The next step is to incorporate this algorithm in a network setting.

Acknowledgements. The authors would like to thank Marco Vijfvinkel for his support with the numerical experiments.

REFERENCES

[1] S. Asmussen and G. Koole (1993) Marked point processes as limits of Markovian arrival streams. *Journal of Applied Probability* 30, 365-372.

[2] P.C. Rao and R. Suri (1994) Approximate Queueing Network Models for Closed Fabrication/Assembly Systems Part 1: Single Level Systems. *Production and Operations Management* 3(4), 244-275.

[3] Y. Dallery and B. Gershwin (1992) Manufacturing flow line systems: a review of models and analytical results. *Queueing Systems 12*, 3-94.

[4] S.B. Gershwin and M.H. Burman (2000) A Decomposition Method for Analyzing Inhomogeneous Assembly/Disassembly Systems. *Annals of Operation Research* 93, 91-115.

[5] Hermann Gold (1998) A Markovian Single Server with Upstream Job and Downstream Demand Arrival Stream. *Queueing Systems* 30, 435-455.

[6] N. Hemachandra and S.K. Eedupuganti (2003) Peformance Analysis and Buffer Allocations in some Open Assembly Systems. *Computers and Operations Research* 30, 695-704.

[7] A. Krishnamurthy, R. Suri and M. Vernon (2004) Analysis of a Fork/Join Synchronization Station with Inputs from Coxian Servers in a Closed Queueing Network *Annals of Operations Research* 125, 69-94.

[8] G. Latouche and V. Ramaswami (1999) *Introduction to Matrix Analytic Methods in Stochastic Modeling.* ASA-SIAM Series on Statistics and Applied Probability 5.

[9] H.G. Perros (1989) A Bibliography of Papers on Queueing Networks with Finite Capacity Queues. *Perf. Eval. 10*, 255-260.

[10] V. Naoumov, U.R. Krieger, D. Wagner (1997) Analysis of a Multiserver Delay-Loss System with a General Markovian Arrival Process. *Matrix-Analytic Methods in Stochastic Models (A.S.Alfa and S.R.Chakravarthy eds), Lecture Notes in Pure and Applied Mathematics*, 183, Marcel Dekker, New York, 1996.

[11] H.C. Tijms (1994) *Stochastic models: an algorithmic approach.* John Wiley & Sons, Chichester.