Product form solution to production systems with job type restricted machines

Citation for published version (APA):

Document status and date:
Published: 01/01/1999

Document Version:
Publisher’s PDF, also known as Version of Record (includes final page, issue and volume numbers)

Please check the document version of this publication:
• A submitted manuscript is the version of the article upon submission and before peer-review. There can be important differences between the submitted version and the official published version of record. People interested in the research are advised to contact the author for the final version of the publication, or visit the DOI to the publisher’s website.
• The final author version and the galley proof are versions of the publication after peer review.
• The final published version features the final layout of the paper including the volume, issue and page numbers.

Link to publication

General rights
Copyright and moral rights for the publications made accessible in the public portal are retained by the authors and/or other copyright owners and it is a condition of accessing publications that users recognise and abide by the legal requirements associated with these rights.

• Users may download and print one copy of any publication from the public portal for the purpose of private study or research.
• You may not further distribute the material or use it for any profit-making activity or commercial gain
• You may freely distribute the URL identifying the publication in the public portal.

If the publication is distributed under the terms of Article 25fa of the Dutch Copyright Act, indicated by the “Taverne” license above, please follow below link for the End User Agreement:
www.tue.nl/taverne

Take down policy
If you believe that this document breaches copyright please contact us at:
openaccess@tue.nl
providing details and we will investigate your claim.
Product form solution to production systems with job type restricted machines

Jeremy Visschers  Ivo Adan  Jaap Wessels
jeremy@win.tue.nl  iadan@win.tue.nl  wessels@win.tue.nl
Eindhoven University of Technology
P.O.box 513
5600MB Eindhoven
The Netherlands

Abstract

In this paper we present a class of queueing models with multiple job types, multiple machines and machine dependent processing times. Each machine is restricted in such a way that it can only handle jobs from a specific, machine dependent, set of job types. If a job arrives at the system and it can be handled by two or more idle machines, then a probability distribution determines to which machine the job is sent. These probability distributions are control parameters for this model and the value of the probabilities influences the performance of the model. Using partial balance equations we show that only for appropriately chosen values of these probabilities, this model has a product form solution.

1 Introduction

In research considerable attention has been devoted to product form stationary distributions for queueing models. Most models that are known to have a product form distribution are reversible or quasi reversible ([Muntz, 1972],[Kelly, 1979], [Walrand & Varaiya, 1982]). A recent result is given by [Berezner, Kriel & Krzesinski, 1995]. They present the class of Order Independent queues and show that these models are quasi reversible.

In this paper we consider a class of queueing models with multiple machines and multiple job types. Each machine can handle jobs from a specific subset of job types. A special model in this class was considered by [Green, 1985]. She considered a model with general-use machines, capable of handling all job types, and limited-use machines, capable of handling only specific jobs. She assumed a FCFS service order. The specific jobs, which can be handled by both types of machines, prefer the limited-use machines. It can be verified that, under these assumptions, this model is not quasi reversible, and that it does not have a product form solution. Green constructed an approximation for this model based on the Matrix Geometric theory ([Neuts, 1980]).

Greens assumption that the specific jobs prefer the limited-use machines, however, can be relaxed. One way of relaxing it, is to introduce a probability distribution to determine, to which machine an arriving job is sent, in case of a choice between two or more idle machines. This is done for a model with one general-use machine and one limited-use machine in [Visschers, Adan & Wessels, 1999]. In this model only a choice has to be made if a specific job enters the empty system. They choose to sent a job to the limited-use machine with probability $1 - \gamma$ and to the general-use machine with probability $\gamma$. The control parameter $\gamma$ is used as a degree of freedom to solve the equilibrium equations and it is shown that for one specific value of $\gamma$, this model does have a product form solution.

In this paper we extend this result to the general model with multiple machines. In this general model an arriving job can find the system in a lot of different situations, with respect to which and how many idle machines there are. For each of these situations a probability distribution is needed to
determine to which machine the job is sent. This results in a lot of different probability distributions, each dependent on the job type of the arriving job and on which and how many machines it can choose from (i.e. idle machine that can handle it). All these probability distributions are control parameters of the model.

For this model we derive partial balance equations, that are similar to the partial balance equations that imply quasi reversibility ([Kelly, 1979]). The equations we derive, however, do not imply quasi reversibility. But if these equations are satisfied, then the model has a product form solution. We will show that the partial balance equations are satisfied (and thus that the model has a product form solution) if a condition, the so-called condition A, is satisfied. Condition A is in fact a set of equations which the probability distributions, that determine to which machine an arriving job is sent, must satisfy. We will show that there is at least one feasible solution to the equations in condition A (i.e. all the probability distributions sum up to 1 and all probabilities are between 0 and 1).

In the next section we will introduce the general model and introduce the used notation. Since the notation is rather complex, we will illustrate it with an example. At the end of that section we will formulate the equilibrium equations for the model. In Section 3 we will formulate the main results and proof it. The main result depends on whether the so-called condition A is satisfied. In Section 4 we show that this condition is indeed satisfied for at least one specific combination of values for the control parameters. We also show that these values are feasible. In the final section of this paper we will give some conclusions and some suggestions for further research.

2 The model

In this section we will give the description of the model. Since the model and the notation we are going to use are rather complex, we will use the example in figure 1 (a) as an illustration. First we will introduce the model, followed by the state description and a description of the transition behaviour. Before we conclude this section with the set of equilibrium equations, we will introduce some notation.

We consider a machine model with $K$ machines ($K = 1, 2, 3, \ldots$) and machine dependent processing times. The $i$-th machine is denoted by $m_i$ and $\mathcal{M}$ denotes the set of machines $\{m_1, m_2, \ldots, m_K\}$. Each machine is capable of handling a specific subset of job types. The total set of job types is denoted by $\mathcal{C}$ and machine $m_i$ can only handle jobs from the set $\mathcal{C}(m_i) \subset \mathcal{C}$. In figure 1 (a) a system is shown with two machines, where $\mathcal{C}(m_1) = \{a, b\}$ and $\mathcal{C}(m_2) = \{a\}$.

Jobs arrive at the system according to Poisson process with intensity $\lambda_i$ for jobs of type $i \in \mathcal{C}$. Jobs are processed in a first-come-first-served (FCFS) order. Processing times on machine $m_i$ are exponentially distributed with parameter $\mu_{mi}$ ($m_i \in \mathcal{M}$). An arriving job joins the end of the queue if it finds all machines, that are capable of handling it, busy. If an arriving job finds exactly one idle machine, that can process it, then it is taken into service by this machine. However, if an arriving job finds more idle machines, that can process it, then there is a choice to which machine it is sent. For example, if in figure 1 (a) a type $a$ job arrives in the empty system, it can be sent to each of the two machines. It seems a reasonable choice to send it to the fastest machine, but if machine $m_1$ is the fastest, then the capacity for type $b$ jobs becomes occupied. An arriving type $b$ job has to wait until the type $a$ job is finished. If the type $a$ job was sent to the slower machine $m_2$, then the arriving type $b$ job could be taken into service directly. So with respect to (for example) the mean waiting time the best choice is not trivial.
Figure 1: (a) An example of a two-machine model. Machine $m_1$ can handle both type $a$ and $b$ job and machine $m_2$ can only handle type $a$ jobs. Processing times are exponentially distributed with parameter $\mu_i$ on machine $m_i$. (b) the transition rate diagram of the model in figure (a). The transitions are characterised by the large jumps to the North-West. The behaviour on the horizontal boundary is not given in detail, since it would only make the picture unnecessary complicated. Probabilities $p_i$ equal $(1 - p)p^i$ ($p = \lambda_b/(\lambda_a + \lambda_b)$).

It is clear that there are numerous policies to assign jobs to machines in case of a choice. In this paper we will use a probability distribution to determine to which machine a job is sent. For each possible situation, with respect to which machines are idle or busy, a probability distribution is required. All these probability distributions are control parameters for the model and thus determine the performance of the model.

Before we continue we need to introduce the following notation:

- $\lambda_X := \sum_{c \in X} \lambda_c$, where $X \subset C$
- $\mu_X := \sum_{M \in X} \mu_M$, where $X \subset M$
- $\mathcal{M}^i := \text{set of all possible sequences } M_1, M_2, \ldots, M_i \text{ with } M_j \in \mathcal{M} (j = 1, 2, \ldots, i) \text{ and } M_j \neq M_k (j, k = 1, 2, \ldots, i \text{ and } j \neq k)$
- $U(X) := \text{set of job types unique to the machines in } X \subset \mathcal{M}, \text{thus the set of job types that cannot be handled by machines outside } X$
- $C(X) := \text{total set of job types that can be handled by the machines in } X \subset \mathcal{M}, \text{which is equal to } \bigcup_{m \in X} C(m)$

### 2.1 The state description

Because of the job type restrictions on the machines it is possible that jobs are overtaken by jobs of other job types. Type $a$ jobs can overtake waiting type $b$ jobs if machine $m_1$ is busy and $m_2$ is not
(see figure 1 (a)). This leads to a very complex transition behaviour and it may be necessary to use a detailed state description. A possible state description is a string representation of the total queue, including jobs in service, in order of arrival (from right to left). This state description is similar to the one used by [Berezner, Kriel & Krzesinski, 1995]. The only difference is that jobs in the queue, that are in service, are denoted by the machine they are being processed on. For example \((baambbbm1)\) is a possible state of the model in figure 1 (a). In this state both machines are busy. Machine \(m1\) is handling a job that arrived in the system before all the other jobs in the queue. The type \(b\) jobs between machine \(m1\) and \(m2\) are overtaken by the job on machine \(m2\). These jobs arrived between the jobs on machines \(m1\) and \(m2\) and they have to wait until \(m1\) can process them. At the end of the queue there are type \(a\) and \(b\) jobs that arrived in the system with two busy machines.

It is important to note that in this state description we loose job type information for the jobs that are in service, since we only denote the machine that is handling the job and not the job type of the job. In [Visschers, Adan & Wessels, 1999] it was shown that a more detailed state description, where this job type information is remembered, does not result in a product form solution for the simple two-machine model in figure 1. It is reasonable to assume that for more complex models with more machines it is also essential to forget this job type information.

Since it is cumbersome to analyse the model using this detailed state description, it might be more convenient to aggregate the state description. This can be done, without losing the Markov property, by denoting only the number of jobs between two machines in the queue instead of the whole string of jobs. The state \((baambbbm1)\) then becomes \((3, m2, 2, m1)\). With this aggregation we loose job type information about the jobs in the queue. But since we know what the job types of these jobs could be, we also know what the probability is that the job is of a certain type. For example: \((2, m3, 4, m2, 1, m1)\) is a possible state of the system in figure 2. Obviously, the job between \(m1\) and \(m2\) can only be of type \(b\). Between \(m2\) and \(m1\) there can be only jobs of type \(b\) and \(c\) (since a job of type \(a\) would be processed by machine \(m3\)). Thus we know that a job between \(m2\) and \(m3\) is of type \(b\) with probability \(\lambda_b/(\lambda_b + \lambda_c)\) and of type \(c\) with probability \(\lambda_c/(\lambda_b + \lambda_c)\).

In general we will denote a state, in which there are \(i\) machines busy (These machines are denotes by \(M1, M2, \ldots, Mi\)), by \((ni, M1, ni-1, M1-1, \ldots, n1, M1)\), where \(nj\) denote the number of jobs between the machines \(Mj+1\) and \(Mj\) (except \(n1\), which denotes the number of job behind \(M1\)). The empty state will be denoted by 0. The state space is denoted by \(\delta\) and to simplify the notation we will use \(s\) to denote an arbitrary state \((ni, M1, \ldots, n1, M1)\) \(\in \delta\). Note that the waiting jobs between machines \(Mj+1\) and \(Mj\) can only be handled by the machines \(Mj, \ldots, M1\) and not by any of the machines \(Mi, \ldots, Mj+1\) or any of the idle machines. Thus waiting jobs between machines \(Mj+1\) and \(Mj\) can only be of type \(c \in U(M1, \ldots, Mj)\).

### 2.2 The transition behaviour

Even with the aggregated state space the transition behaviour is still complex. The transition behaviour of the model in figure 1 (a) is depicted in figure 1 (b). The transition behaviour is characterised by geometric jumps to the North-West direction. We will discuss the possible transitions below.

From an arbitrary state the following transitions are possible:

(i) Arrival of jobs: if a job arrives that cannot be handled by any of the idle machines, it joins the end of the queue. From state \(s = (ni, M1, \ldots, n1, M1)\), arriving jobs that cannot by handled by any of the idle machine must be of type \(c \in U(M1, \ldots, M1)\) Thus such a job arrives with
Figure 2: An example of a three machine model. Machine $m_1$ can handle type $a$ and $b$ jobs, machine $m_2$ can handle type $a$ and type $c$ job en machine $m_3$ can only handle type $a$ jobs. Again the processing times are exponentially distributed with parameter $\mu_i$ on machine $m_i$. 

With intensity $\lambda_U(M_i,...,M_1)$ a job arrives that can be handled by one or more idle machines. For this job a probability distribution determines to which machine it is sent. In figure 1 (b) an arrival is depicted by a jump to the East or, for states on the vertical boundary, by jumps to the North and East. States on the vertical boundary are states in which only machine $m_1$ is busy. If a job of type $b$ arrives, it joins the queue, since $m_2$ cannot process it (jump to the North). If a job of type $a$ arrives it is taken into service by $m_2$ (jump to the West). If a job of type $a$ arrives in the empty system, we can sent it to each of the two machines. It is sent to machine $m_1$ with probability $\gamma$ and to machine $m_2$ with probability $1 - \gamma$.

The probability distributions determine the total transition rate from a state $s$ to 0, $M$, $s$. In the model in figure 1 (a) the transition rate from the empty state to state 0, $m_i$ equals $\gamma \lambda_a + \lambda_b$ and to state 0, $m_2$ it equals $(1 - \gamma) \lambda_a$. In case of more machines the notation can become very complex. It can, however, be simplified by introducing the following notation:

$$\lambda_M(M_i, \ldots, M_1) := \text{transition intensity from state } s = (n_i, M_i, \ldots, n_1, M_1) \text{ to state } 0, M, s, \text{ for all possible states } s \in S \text{ and all machines } M \text{ that are not busy in state } s$$

These arrival intensities are determined by the probability distributions. Given the probability distributions, the arrival rates can be calculated. The other way around is more complicated and will be discussed in section 4. Note that the following holds

$$\sum_{M \in M \setminus \{M_i, \ldots, M_1\}} \lambda_M(M_i, \ldots, M_1) = \lambda_C - \lambda_U(M_i, \ldots, M_1)$$

(1)
(ii) Departure of jobs: if a job is finished on a machine, then that machine scans the queue from right to left until it finds the first job it can handle. There are two possibilities:

1. The machine does not find a job it can handle. In figure 1 (b) these transitions are depicted by the jumps from the interior of the random walk (states where both machines are busy) to the vertical boundary (states where only $m_1$ is busy). Machine $m_2$ checks all the $n_2$ jobs behind it and with probability $p^M$ none of these jobs is of type $a$ (where $p = \lambda_b/(\lambda_a + \lambda_b)$). In general such a transition is possible to state $s = (n_i, M_i, \ldots, n_1, M_1)$ from state $(n_i, M_i, \ldots, l, M, n_j - l, M_j, \ldots, n_1, M_1)$, where machine $M$ is situated in the queue between $M_j + 1$ and $M_j$. We will denote this state by $\text{insert}^M_{jl}(s)$. In this state there are $n_j - l$ jobs between $M$ and $M_j$ and $l$ jobs between $M_{j+1}$ and $M$. The jobs between machines $M$ and $M_{j+1}$ can only be of type $c \in U((M, M_j, \ldots, M_1))$. With probability $\lambda_U((M_j, \ldots, M_1))/\lambda_U((M, M_j, \ldots, M_1))$ such a job cannot be handled by machine $M$. A job between machine $M_{j+2}$ and $M_{j+1}$ cannot be handled by machine $M$ with probability $\lambda_U((M_{j+1}, \ldots, M_1))/\lambda_U((M, M_{j+1}, \ldots, M_1))$. Thus if machine $M$ finishes its job, it finds no job in the queue, it can handle with probability

$$p^M_{jl}(s) := \delta_i(M)^{n_i} \cdots \delta_{j+1}(M)^{n_{j+1}} \delta_j(M)^{l}$$

where

$$\delta_k(M) := \frac{\lambda_U((M_k, \ldots, M_1))}{\lambda_U((M, M_k, \ldots, M_1))} \quad k = 1, 2, \ldots, K$$
Thus with probability $p^M_{jl}(s)$ a jump is made from state $\text{insert}_{jl}^M(s)$ to state $s$, given that machine $M$ finishes its job. This transition is illustrated in figure 3.

\[
\text{swap}^M_{kl}(n_i, M_i, \ldots, n_1, M_1)
\]

\[
\begin{array}{cccc}
n_i & n_j + 1 + n_{j-1} & l & n_1 \\
\circ \ldots \circ M_i & \circ \ldots \circ M_{j-1} & \circ \ldots \circ M_j & \circ \ldots \circ M_1
\end{array}
\]

$M_j$ finishes its job and finds a job in the queue

with probability

\[
q^M_{kl}(n_i, M_i, \ldots, n_1, M_1)
\]

\[
\begin{array}{cccc}
n_i & n_j & n_1 \\
\circ \ldots \circ M_i & \circ \ldots \circ M_j & \circ \ldots \circ M_1
\end{array}
\]

Figure 4: The transition from state $\text{swap}^M_{kl}(s)$ to state $s$ illustrated. The circles represent jobs that are waiting in the queue. The squares represent jobs that are in service at a machine. If machine $M_j$ finishes its job in state $\text{swap}^M_{kl}(s)$, then the first job, it can handle, is with probability $q^M_{kl}(s)$ the $n_{j-1} + 1$-th job between $M_{j-1}$ and $M_j+1$. Machine $M_j$ takes this job into service, leaving $n_{j-1}$ waiting jobs between $M_j$ and machine $M_{j-1}$ and $n_{j+1}$ waiting jobs between $M_j$ and $M_{j+1}$. Now a transition is made to state $s$.

(2) The machine finds a job it can handle. In this case one of the busy machines finishes its job and finds somewhere in the queue another job it can process. For the model in figure 1 (a) there are two possibilities. Either machine $m_1$ finishes its job and takes the next job in the queue (jump to the South) or machine $m_2$ finishes its job and takes the next type $a$ job in the queue. This job overtakes a geometrically distributed number of type $b$ jobs (large jump to the North-West, figure 1 (a)). In general the state $s$ can be reached by such a transition from state

\[
(n_i, M_i, \ldots, n_1, M_1)
\]

This state will be denoted by $\text{swap}^M_{kl}(s)$. In this state machine $M_j$ is located between machine $M_{k+1}$ and $M_k$. Between machines $M_{j-1}$ and $M_{j+1}$ there are $n_{j-1} + n_j + 1$ jobs and no machines. If machine $M_j$ finishes its job a transition is made to state $s$ if the first job $M_j$ can handle is the $n_{j-1} + 1$-th job (from the right) between $M_{j+1}$ and $M_{j-1}$. The probability that the $n_{j-1} + 1$-th job between $M_{j-1}$ and $M_{j+1}$ is the first job that $M_j$ can handle equals:

\[
q^M_{kl}(s) := \delta_{j-1}(M_j)n_{j-1} \ldots \delta_{k+1}(M_j)n_{k+1}\delta_k(M_j)(1 - \delta_{j-1}(M_j))
\]
The system makes a jump from state $swap_{kl}^M(s)$ to state $s$ with probability $q_{kl}^M(s)$, given that $M_f$ finishes its job. This transition is illustrated in figure 4.

2.3 The equilibrium equations

We will now formulate the set of equilibrium equations. The equilibrium probability of being in the state $s = (n_i, M_i, \ldots, n_1, M_1)$ is denoted by $\pi(s)$. We assume that the system is ergodic and that the equilibrium probabilities exist. This assumption will be discussed in remark 3.2.

In the previous subsection we showed that the state $s$ can be reached by (i) an arrival of a job, (ii) a departure of a job on a machine that finds no new job in the queue and (iii) the departure of a job on a machine that does find a new job in the queue. The equilibrium equations display these three possibilities. The left hand side of the equations equals the total probability flux out of state $s$. The right hand side of the equations equals the probability flux into state $s$ and consist of three parts, corresponding to resp. (i), (ii) and (iii). In part (ii) we need to sum over all possible states with one more busy machine (machine $M_1$) and over all possible positions of this machine in the queue. In part (iii) we need to sum over all machine $M_1, \ldots, M_i$ and the positions of the machine. The equilibrium equations are given by (for all possible states $s = (n_i, M_i, \ldots, n_1, M_1) \in S\setminus\{0\}$)

In case $n_i > 0$:

$$
\left(\lambda_e + \mu_{(M_1, \ldots, M_i)}\right) \pi(s) = \lambda U((M_1, \ldots, M_i)) \pi(n_i - 1, M_1, \ldots, n_1, M_1) + \sum_{M \in M\setminus\{M_1, \ldots, M_i\}} \mu_M P_M(s) + \sum_{j=1}^{i} \mu_{M_j} Q_{M_j}(s)
$$

(2)

In case $n_i = 0$:

$$
\left(\lambda_e + \mu_{(M_1, \ldots, M_i)}\right) \pi(s) = \lambda M_i(M_1, \ldots, M_i-1) \pi(n_i-1, M_i-1, \ldots, n_1, M_1) + \sum_{M \in M\setminus\{M_1, \ldots, M_i\}} \mu_M P_M(s) + \sum_{j=1}^{i} \mu_{M_j} Q_{M_j}(s)
$$

(3)

where

$$
P_M(s) = \sum_{k=1}^{i} \sum_{l=0}^{n_k} p_{kl}^M(s) \pi(insert_{kl}^M(s)) + p_{in_1}^M(s) \pi(s, 0, M)
$$

(4)

$$
Q_{M_j}(s) = \sum_{k=1}^{j-1} q_{kl}^M(s) \pi(swapp_{kl}^M(s)) + q_{in_1}^M \pi(swapp_{00}^M)
$$

(5)

We may omit the equation for state 0, since the set of equilibrium equations is dependent.


\section{The main result}

In the previous section we have described the model and formulated the equilibrium equations for the model. In this section we will formulate partial balance equations for the model. Although these partial balance equations do not imply quasi reversibility, we will show that if these equations are satisfied, then the model has a product form solution. We will show that the partial balance equations are satisfied and thus has a product form distribution, if a so-called condition \( A \) is satisfied.

\subsection{Partial balance equations}

In [Kelly, 1979] it is stated that a model is quasi reversible if the following partial balance equations are satisfied: for every job type \( c \) and for every state the probability flux out of that state, due to an arrival of a type \( c \) job, equals the probability flux into the state, due to the departure of a type \( c \) job. For our model it is hard to formulate these local balance equations, since the job type of a departing job is unknown. A departing job from machine \( M_j \) can be of type \( c \in \mathcal{C}(M_j) \). The probabilities that a departing job from machine \( M_j \) is of type \( c \in \mathcal{C}(M_j) \), however, are not known. These probabilities depend on the state of the system, because otherwise a product form solution should have been found for a more detailed state description that includes the job type of the jobs on the machines. This is not the case, not even for a simple two-machine model [Visschers, Adan & Wessels, 1999].

We formulate local balance equations for each ‘machine-type’ instead of job-type. For state \( s \) this results in two equations. One equation for each machine \( M \in \mathcal{M}\{M_i, \ldots, M_1\} \): the total probability flux out of state \( s \) due to an arrival of a job that is taken into service by machine \( M \) (arrival intensity \( \lambda_M(M_i, \ldots, M_1) \)) equals the total probability flux into state \( s \) due to the departure of a job on machine \( M \) after which machine becomes idle. This equation is given below:

\begin{equation}
\mu_M \mathcal{P}_M(s) = \lambda_M(M_i, \ldots, M_1) \pi(s) \quad M \in \mathcal{M}\{M_i, \ldots, M_1\} \tag{6}
\end{equation}

In the second equation the total probability flux out of state \( s \), due to an arrival of a job that can be processed on machine \( M_j \in \{M_i, \ldots, M_1\} \) and not on any of the machines in \( \{M_i, \ldots, M_{j+1}\} \) (arrival intensity \( \lambda_U((M_{j+1}, \ldots, M_1)) - \lambda_U((M_{j-1}, \ldots, M_1)) \)), equals the total probability flux into state \( s \), due to the departure of a job on one of the machines \( M_i, \ldots, M_1 \). This results in the following equation.

\begin{equation}
\mu_M \mathcal{Q}_{M_j}(s) = (\lambda_U((M_{j+1}, \ldots, M_1)) - \lambda_U((M_{j-1}, \ldots, M_1))) \pi(s) \tag{7}
\end{equation}

Subtracting the equations (6) for all machines \( M \in \mathcal{M}\{M_i, \ldots, M_1\} \) and (7) for all machines \( M_j \in \{M_i, \ldots, M_1\} \) from equations (2) resp. (3) yields the following two equations.

In case \( n_i > 0 \):

\begin{equation}
\mu_{\{M_i, \ldots, M_1\}} \pi(s) = \lambda_U((M_i, \ldots, M_1)) \pi(n_i - 1, M_i, \ldots, n_1, M_1) \tag{8}
\end{equation}

In case \( n_i = 0 \):

\begin{equation}
\mu_{\{M_i, \ldots, M_1\}} \pi(s) = \lambda_{M_i}(M_1, \ldots, M_{i-1}) \pi(n_{i-1}, M_{i-1}, \ldots, n_1, M_1) \tag{9}
\end{equation}

From these equations the product form solution can be derived.
\[ \pi(s) = \alpha_i^{n_i} \frac{\lambda_{M_i}(M_i \ldots, M_1)}{\mu(M_i \ldots, M_1)} \cdots \alpha_1^{n_1} \frac{\lambda_{M_1}(\emptyset)}{\mu(M_1)} \pi(0) \]  

(10)

where

\[ \alpha_k := \frac{\lambda_j(M_i \ldots, M_1)}{\mu(M_i \ldots, M_1)} \quad k = 1, 2, \ldots, K \]

If this product form solution satisfies the equations (6) for all machines \( M \in \mathcal{M} \backslash \{M_i, \ldots, M_1\} \) and (7) then also the equations (2) and (3) are satisfied. Since \( \lambda_{M_i}(M_i \ldots, M_1) \) still depends on the control parameters, it is clear that the partial balance equation (6) and (7) are not satisfied for every value of the control parameters. We will show that these equations are satisfied if the control parameters are chosen such that the following condition is satisfied

**Condition A**

For all sequences \((M_i, \ldots, M_1) \in \mathcal{M}^j \) for all \( i = 1, \ldots, K \) the following holds

\[ \prod_{j=1}^i \lambda_{M_j}(M_{j-1} \ldots, M_1) = \prod_{j=1}^i \lambda_{M_j}(M_{j-1} \ldots, M_1) \]

for every permutation \((M_i, \ldots, M_1)\) of \((M_i, \ldots, M_1)\).

This condition implies that the product \( \prod_{j=1}^i \lambda_{M_j}(M_{j-1} \ldots, M_1) \) should be independent of the order of the machines \( M_1, \ldots, M_1 \) in the queue. To simplify the notation we will denote the product \( \prod_{j=1}^i \lambda_{M_j}(M_{j-1} \ldots, M_1) \) by \( \Pi_j(M_i, \ldots, M_1) \). The product \( \prod_{j=1}^i \mu_j(M_i \ldots, M_1) \) will be denoted by \( \Pi_j(M_i, \ldots, M_1) \). Note that \( \Pi_j(M_i, \ldots, M_1) \) is independent of the order of the machines \( M_i, \ldots, M_1 \) if condition A is satisfied and that \( \Pi_j(M_i, \ldots, M_1) \) is not. Now that we have formulated the condition A, we can state our main result.

**Theorem 3.1** The model described in section 2 has a product form solution if the control parameters of the model (the probability distributions) are chosen in such a way that condition A is satisfied. The solution is then given by

\[ \pi(s) = \alpha_i^{n_i} \cdots \alpha_1^{n_1} \frac{\Pi_j(M_i, \ldots, M_1)}{\Pi_j(M_i, \ldots, M_1)} \pi(0) \]  

(10')

**Proof**

We only have to verify whether the partial balance equations are satisfied if condition A holds. Therefore we have to verify if the solution (10) satisfies the equations (6) and (7). We will first look at one term of the sum \( \mathcal{P}_M(s) \). This term is divided by \( \pi(s) \). After inserting the product form solution (10) this term becomes

\[ \frac{\mu_M \mathcal{P}_M^M(s)}{\pi(s)} = \frac{\left( \delta_j(M_i \ldots, M_1) \right)^{n_i} \cdots \left( \delta_j(M_i \ldots, M_1) \right)^{n_1} \Pi_j(M_i, \ldots, M_1)}{\Pi_j(M_i, \ldots, M_1)} \]

(11)
Note that if condition A is satisfied then the following holds:

\[ \prod_{k} (M_i, \ldots, M_j, \ldots, M_1) = \lambda_M(M_i, \ldots, M_1) \Pi_{k} (M_i, \ldots, M_1) \] (12)

In order to simplify equation (11) we define

\[ \beta_k := \frac{\delta_k(M)}{\mu_k(M)} \cdot \frac{1}{\alpha_k} \]

\[ = \frac{\mu(M)}{\mu(M, M_1)} \] \quad \text{for } k = i, \ldots, j

If we substitute \( \beta_k \) into equation (11) and use equation (12) to simplify it then this equation can be reduced to

\[ \mu_M \frac{p_{M}(s)}{\pi(s)} = \lambda_M(M_1 \ldots, M_1) (\beta_1)^{n_i+1} \ldots (\beta_{k+1})^{n_{k+1}+1} (1 - \beta_k) (1 - \beta_k)' \] (13)

where

\[ 1 - \beta_k = \frac{\mu_M}{\mu(M, M_1)} \]

This equation can be used to simplify the equation (6) with the solution (10) inserted. This leads to

\[ \frac{\mu_M \mathcal{P}_M(s)}{\pi(s)} = \lambda_M(M_1 \ldots, M_1) \sum_{k=1}^{i} \sum_{l=0}^{n_k} (\beta_1)^{n_i+1} \ldots (\beta_{k+1})^{n_{k+1}+1} (1 - \beta_k) \]

\[ + \lambda_M(M_1 \ldots, M_1) (\beta_1)^{n_i+1} \ldots (\beta_2)^{n_2+1} (\beta_1)^{n_1+1} \] (14)

With the use of the following argument this equation can easily be simplified. Let us assume that a machine is checking a queue of jobs separated into \( i \) subsequent parts. First it checks part \( i \), that consists of \( n_i + 1 \) jobs. The probability that an arbitrary job in this part cannot be handled by the machine equals \( \beta_i \). Then the probability that no job in this part can be handled by the machine equals \( \beta_i^{n_i+1} \). If it finds no job in part \( i \) it continues to check part \( i - 1 \) that consists of \( n_{i-1} + 1 \) jobs. The probability that an arbitrary job of this part cannot be handled by the machine is \( \beta_{i-1} \). It continues checking the jobs until it finds the first job it can handle. The probability that the first job in part \( k \) equals \( \beta_i^{n_i+1} \ldots \beta_k^{n_k+1} (1 - \beta_k) \). The sum over all jobs in the queue of the probability that it is the first job found by the machine plus the probability that the machine finds no job (\( \beta_i^{n_i+1} \ldots \beta_k^{n_k+1} \)) should equal 1. Using this argument it is clear that equation (14) reduces to

\[ \frac{\mu_M \mathcal{P}_M(s)}{\pi(s)} = \lambda_M(M_1, \ldots, M_1) \] (15)

In a similar way it can be proven that (7) is also satisfied if condition A holds. Thus the partial balance are satisfied and thus the model has a product form solution, if condition A is satisfied.

**Remark 3.2** We assumed that the model was ergodic and that the equilibrium probabilities exist. Now that we have found the equilibrium probabilities it is easy to derive sufficient conditions for ergodicity. The models that satisfy condition A are ergodic if

\[ \lambda_{U(M_i, \ldots, M_1)} < \mu_{M_1} + \cdots + \mu_{M_i} \]

for every set \( \{M_i, \ldots, M_1\} \in \mathcal{M}^i \), for every \( i = 1, 2, \ldots, K \).
4 The condition A

In the previous section we showed that the model has a product form solution if condition A is satisfied. We also stated that the condition A will not be satisfied for all values of the control parameters. In this section we will show that the control parameters can be chosen such that condition A is satisfied. We will prove that these values are feasible, i.e. all probabilities are between zero and one and all probability distributions sum up to one. This will be proven using a backwards recursive relation for the arrival intensities $\lambda_M(M_1, \ldots, M_1)$, which we will derive.

4.1 Recursive relation

In the previous section we defined the transition rate from state $s$ to state 0, $M$, $s$ as $\lambda_M(M_1, \ldots, M_1)$. If we apply condition A on $\{M_1, \ldots, M_1\}$ and the permutation $\{M_1, M_1, M_1, \ldots, M_1\}$ the following equation is obtained

$$\lambda_{M_1}(M_1, \ldots, M_1)\lambda_{M_1}(M_1, \ldots, M_1) = \lambda_{M_1}(M_1, M_1, \ldots, M_1)\lambda_{M_1}(M_1, \ldots, M_1)$$

(16)

If equation (16) holds for all sequences of machines $\{M_1, \ldots, M_1\} \in \mathcal{M}$ then also condition A is satisfied. Equation (16) implies that for two arbitrary machines $M, m \in \mathcal{M}/\{M_1, \ldots, M_1\}$ the following holds

$$\lambda_m(M_1, \ldots, M_1) = \lambda_m(M, \ldots, M)\lambda_m(M_1, \ldots, M_1)$$

(17)

We choose an arbitrary machine $M \in \mathcal{M}/\{M_1, \ldots, M_1\}$ and insert for every machine $m \in \mathcal{M}/\{M_1, \ldots, M_1\}$ the equation (17) into equation (1). Now the following backwards recursive relation for $\lambda_M(M_1, \ldots, M_1)$ is obtained.

$$\lambda_M(M_1, \ldots, M_1) = \frac{\lambda_C - \lambda_U(M_1, \ldots, M_1)}{\sum_{m \in \mathcal{M}/\{M_1, \ldots, M_1\}} \lambda_m(m, \ldots, M_1)\lambda_m(M, \ldots, M_1)}$$

(18)

4.2 A feasible solution

With this recursive relation it is possible to calculate the arrival rates. This is described in the proof of the following lemma.

**Lemma 4.1** The transition rates $\lambda_M(M_1, \ldots, M_1)$ (for all $\{M_1, \ldots, M_1\} \in \mathcal{M}$, $i = 1, 2, \ldots, K$ and $M \in \mathcal{M}/\{M_1, \ldots, M_1\}$) can be chosen in such a way that condition A is satisfied.

**Proof**

The transition rates from the states $(n_{K-1}, M_{K-1}, \ldots, n_1, M_1)$ (only one idle machine) to the states $(0, M_K, \ldots, n_1, M_1)$ (no idle machines) are fixed, since there is only one machine to choose from. Since the only jobs machine $M$ can handle are of type $C(M_K)$, it is clear that the rates equal

$$\lambda_{M_K}(M_K, \ldots, M_1) = \lambda_C(M_K)$$
Note that these rates satisfied equation (1). Using the recursive relation (18), the remaining arrival rates can be calculated, such that equation (1) is satisfied. We only have to show that also equation (16) is satisfied. It can easily be verified that equation (16) holds if \( i = \mathcal{K} \), thus for all sequences \((M_\mathcal{K}, \ldots, M_1) \in \mathcal{M}^\mathcal{K}\). Let us assume that equation (16) holds for all sequences \((M_i, \ldots, M_1) \in \mathcal{M}^i\) for some \(i \in \{1, \ldots, \mathcal{K}\}\). Then the following holds for all sequences \((M_{i-\mathcal{E}}, \ldots, M_1) \in \mathcal{M}^{i-\mathcal{E}}\)

\[
\lambda_{M_{i-2}}(M_{i-2}, \ldots, M_1) \lambda_{M_{i-3}}(M_{i-3}, \ldots, M_1) = \lambda_{M_{i-1}}(M_{i-2}, \ldots, M_1) \frac{\lambda_e - \lambda_{U(M_{i-3}, \ldots, M_1)}}{\sum_{m \in \mathcal{M}\setminus(M_{i-3}, \ldots, M_1)} \lambda_m(M_{i-2}, \ldots, M_1) \lambda_{M_{i-2}}(m, M_{i-3}, \ldots, M_1)} \leq \frac{\lambda_e - \lambda_{U(M_{i-3}, \ldots, M_1)}}{\sum_{m \in \mathcal{M}\setminus(M_{i-3}, \ldots, M_1)} \lambda_m(M_{i-1}, M_{i-2}, \ldots, M_1) \lambda_{M_{i-2}}(m, M_{i-3}, \ldots, M_1)} \leq \lambda_{M_{i-2}}(M_{i-1}, M_{i-3}, \ldots, M_1) \lambda_{M_{i-1}}(M_{i-3}, \ldots, M_1)
\]

The equation with a * hold, because we assumed that equation (16) is satisfied for all sequences \((M_i, \ldots, M_1) \in \mathcal{M}^i\). Now we have shown that equation (16) also holds for \(i - 1\). Thus induction implies that equation (16) is satisfied for all \(i \in \{1, \ldots, \mathcal{K}\}\). □

We have shown that the condition A can be satisfied by choosing the values of the arrival intensities appropriately. The question remains whether these values correspond to feasible control parameters: the probability distributions. Each probability distribution should sum up to one and all probabilities should be between zero and one. It appears that there is at least one feasible combination of values for the probability distributions, corresponding to the values of the arrival intensities. This is proven in the following theorem.

**Theorem 4.2** For the model described in section 2 the control parameters of the model (i.e. the probability distributions) can be chosen in such a way (within their allowed range) that the model has a product form solution.

**Proof**

We have to show that the probabilities, within each probability distribution, can be chosen between zero and one and that each probability distribution sums up to 1. This can be done by reformulating the problem into the following transportation problem.

Given the sources \(c \in \mathcal{C}\setminus U(\{M_i, \ldots, M_1\})\). Source \(c\) has capacity \(\lambda_e\). The sources have to supply customers \(M \in \mathcal{M}\setminus\{M_i, \ldots, M_1\}\). Customer \(M\) has a demand of size \(\lambda_M(M_i, \ldots, M_1)\) and can only be supplied by sources \(c \in \mathcal{C}(M)\). The total demand \((\lambda_e - \lambda_{U(M_i, \ldots, M_1)})\) is equal to the total supply \((\sum_{c \in \mathcal{C}(M)} \lambda_c)\) and the question is: can the supply be transported to the customers in such a way that all the demands are met?

It is clear that a feasible allocation for the transport problem corresponds to a feasible combination of values for the probability distribution of the queueing problem. According to [Ahuja, Magnanti & Orlin, 1993] there exists a feasible allocation if the following condition is satisfied

\[
\lambda_M(M_i, \ldots, M_1) \leq \lambda_e(M)
\]

(19)
Thus if equation (19) is satisfied for all sequences \( \{M, M_i, \ldots, M_1\} \in \mathcal{M}^{i+1} \), then there exists a feasible combination of values for the probability distributions for the queueing model. For \( \lambda_{M \mathcal{K}} (M_{\mathcal{K}-1}, \ldots, M_1) \) it is clear that (19) holds. If equation (19) holds for all \( \lambda_{M(M_{j}, \ldots, M_1)} \) with \( j = i + 1, \ldots, \mathcal{K} \) then the following holds for \( \lambda_{M(M_{j}, \ldots, M_1)} \):

\[
\lambda_{M}(M_i, \ldots, M_1) = \frac{\lambda e - \lambda U(M_i, \ldots, M_1)}{\sum_{m \in \mathcal{M}} \lambda U(M_i, \ldots, M_1) s_{M}(M_i, \ldots, M_1)} \leq \frac{\lambda e - \lambda U(M_i, \ldots, M_1)}{\sum_{m \in \mathcal{M}} \lambda_{M}(M_i, \ldots, M_1) \lambda_{e}(M)} = \lambda e(M)
\]

Thus equation (19) holds for all sequences \( \{M, M_i, \ldots, M_1\} \in \mathcal{M}^{i+1} \).

## 5 Conclusions and further research

In this paper we presented a large class of multi-machine queueing models that have a product form solution if a certain condition is satisfied, the so-called condition A. This condition A can be satisfied by adjusting the control parameters of the model appropriately. These control parameters determine to which machine an arriving job is sent if there is a choice between two or more machines.

If the values of the control parameters deviate such that condition A is not satisfied, then the solution to the equilibrium equations is no longer a product form. Numerical studies of the simple two-machine model in figure 1 (a) suggest that the found product form solution is part of a possibly more complex solution: the only part that remains if the control parameters are chosen appropriately. Further research is, however, needed to determine what the structure of this more complex solution is.

The found product form solution can be used to calculate performance characteristics of the model, like the throughput and mean waiting time. It is clear however that in a lot of cases numerical problems can occur (summing a large quantity of very small numbers leads to instability). Closer examination of the product form solution and the structure of specific models may lead to ways of avoiding the numerical problems and calculating the performance measures in an efficient way.

The found product form solution in cases that condition A holds has given some insight into the solution of these models in general. It is, however, not clear what the solution is in case that condition A is not satisfied. We can use the product form solution as a product form approximation for values of the control parameters that do not satisfy condition A. Since the control parameters only directly influence states where two or more machines are idle and not the states where all machines are busy, it may be expected that this leads to reasonable approximations, especially in cases with a high machine load.

## References


GREEN, L. [1985], A queueing system with general-use and limited-use servers, *Operations Research* 33, 168–182.

KELLY, F.P. [1979], *Reversibility and Stochastic Networks*, Wiley.


