Throughput of a constant work in process manufacturing line subject to failures

WALLACE J. HOPP† and MARK L. SPEARMAN†

We consider a production line consisting of several machines in tandem operating under a constant work in process (CONWIP) control strategy. We assume that processing times are deterministic but machines are subject to exponential failures and repairs. We model this system as a closed queueing network and develop an approximate regenerative model (ARM) for estimating throughput and average cycle time as a function of WIP level. We compare ARM with mean value analysis (MVA) and develop readily computable tests of the suitability of the two approaches to a given production system. Through comparison with simulations, we show that ARM gives better predictions than MVA in a range of realistic situations.

1. Introduction

We consider a production line composed of several machines in tandem in which a constant number of identical jobs are maintained, as illustrated in Fig. 1. This is motivated by the desire to model and control effectively the hybrid push/pull control strategy known as constant work in process (CONWIP) which has been described by Spearman et al. (1990b). Under CONWIP, work cannot begin on a line until the line WIP has fallen below a specified level. Figure 1 illustrates a system in which the total number of jobs is regulated by requiring each job to be accompanied by a 'production card' (which may actually be an electronic signal). When both a card and suitable raw
materials are available at the front of the line, the next job specified by the 'CONWIP backlog' is started. Unlike traditional kanban systems, CONWIP cards are not part number specific, so the backlog can consist of a changing product mix as well as short runs of small lots. Also unlike kanban, in CONWIP work is 'pushed' at all stations downstream from the front of the line. In this way a CONWIP line achieves the WIP limiting benefits of a kanban line without the blocking and changing product mix problems that have limited the application of kanban (see Spearman and Zazanis 1990). CONWIP has also been found to be superior to push systems as material requirement planning (MRP) in simulation studies (see Roderick et al. 1990, Spearman et al. 1990b). A hierarchical control architecture using CONWIP as a means of shop floor control has been described by Spearman et al. (1990a) and is currently being implemented in a large circuit board plant of a major computer manufacturer.

If the processing time on a machine for all jobs is roughly the same, we can model a CONWIP line as a 'tandem closed queueing network' with a single class of jobs by treating the job that has finished and the job that is started as being the same. Even if the CONWIP approach is not used explicitly, it has often been noted that closed queueing networks are often more appropriate than open networks in modelling manufacturing lines since, in most production systems, WIP is not without control (Jackson 1963, Whitt 1984).

The majority of queueing network models assume that processing times are exponential or other 'Poisson driven' distributions, such as the phase-type distributions. (see e.g. Disney and König (1985) for a survey of queueing network models). This particularly true for the literature dealing with closed queueing networks. The assumption of exponential processing times is needed in order to obtain a 'product-form' solution for the joint probability distribution of the system state vector (i.e. the vector of the number of jobs at each machine). Gordon and Newell (1967) pioneered this work within the context of closed queueing networks (CQN). Reiser and Lavenberg (1980) pointed out that the joint probability distribution is more information than is really needed for performance evaluation. Their approach, known as mean value analysis (MVA) iteratively computes the throughput, average cycle time (i.e. the time to traverse the line) and average WIP at each station for increasing WIP levels and is exact for systems with all exponential processing times.

In order to realistically model the performance of serial discrete manufacturing operations, we feel that in many cases it is more reasonable to assume that processing times are deterministic, but the machines are subject to random outages. In our experience, most tools (especially automated ones) have what are effectively constant processing times when the processing times do not depend on part characteristics (e.g. printed circuit manufacturing). The majority of the variability present in the system is due to either random failures, periodic adjustments, or inattention from the operator. It seems reasonable to model these lapses as random, memoryless (i.e. exponential) outages. Also, when combining repair times (that are typically long) with other shorter adjustment times, the exponential distribution again appears to be appropriate. Thus, we are interested in analysing the closed queueing network with deterministic processing times and machines subject to exponential failures and repairs. In particular, we want to be able to compute the throughput and average cycle time as a function of WIP level, in order to provide guidance in setting WIP levels in CONWIP systems.

Modelling the deterministic processing with random outages (DPRO) system is not a new approach. There is a considerable literature analysing the role of interstage
buffers when machines work at a constant rate but are subject to stochastic failures (see e.g., Buzzacott 1971, Meyer et al. 1979, Wijngaard 1979, Hopp et al. 1989). Several researchers have also studied the problem of flow control in systems with machines satisfying this assumption (see e.g., Kimemia and Gerswin 1983, Akella and Kumar 1986, Sharifnia 1988). However, we are not aware of any work that has explicitly considered the discrete parts manufacturing case where the system is closed (e.g. as in a CONWIP line) under the DPRO assumption.

Certainly one approach to modelling the DRPO system would be to use simulation. However, simulation can be tedious to use and require many runs when used as a decision aid. Furthermore, Suri (1988) stresses the timeliness aspect of using analytic models in place of more detailed simulation models by observing that: 'A rough model today is worth more than a refined model in a few months'. For this reason, many analytical tools for modelling manufacturing systems have been developed in recent years. These include: CAN-Q developed by Solberg (1980), RESQ described by Tucci and MacNair (1982) and by Sauer et al. (1982), PANACEA described by Ramakrishnan and Mitra (1982), QNA described by Whitt (1983), and MANUPLAN II developed by Suri et al. (1986). A survey of these and other packages is given by Snowdon and Ammons (1988).

An analytical alternative to simulation for analysing the system is to use MVA with the exponential processing times chosen to match the mean processing times in the actual system. MVA has been shown to be fairly robust in a wide variety of situations (Suri 1983). Unfortunately, this robustness is not guaranteed. Whitt (1984) has demonstrated that the performance of MVA in modelling a non-exponential system can be made arbitrarily bad by increasing the variability of the processing times for the machines. This is particularly relevant to the DRPO system since the coefficient of variation for a deterministic machine subject to long but infrequent failures can be very large.

We note that the system we wish to model is further complicated by the fact several independent processes are taking place so that processing time variability is not the only issue. For example, the throughput in a closed tandem network with only one job and no machine failures is simply the reciprocal of the sum of the mean processing times, regardless of their distribution so long as the machines behave independently. However, we will see that this is not true in a system with failures. In fact, the reciprocal of the sum of the average processing times (i.e. the constant processing time divided by the availability) is an upper bound for throughput in this system.

In the present paper, we develop a quantitative measure of the suitability of MVA to a particular DPRO system. Because we find that in many realistic situations MVA does not accurately predict the throughput versus WIP curve, we develop an alternative model, called the 'approximate regenerative model' (ARM), that explicitly considers the DPRO. We show that ARM is effective in many cases where MVA is not.

The paper is organized as follows. In section 2 we review the mechanics of MVA as applied to the DPRO. In section 3 we describe the ARM. In section 4 we develop quantitative measures of the suitability of both MVA and ARM to specific DPRO systems. In section 5 we provide numerical comparisons of the two methods with simulation results. Conclusions are given in section 6.

2. Mean value analysis

Before reviewing the mechanics of MVA as applied to the DPRO, we give some notation used throughout the paper.
We assume that the system is composed of \( K \) machines in tandem operating with a constant number of identical jobs, \( m \). For each machine \( i \), we have the following parameters:
\[ \tau_i \quad \text{constant processing time}; \]
\[ \lambda_i \quad \text{failure rate}; \] and
\[ \mu_i \quad \text{repair rate}. \]
It is well known that the mean time between failures (MTBF), mean time to repair (MTTR), and stand-alone availability of machine \( i \) are given by \( 1/\lambda_i \) and \( \mu_i/(\lambda_i + \mu_i) \), respectively.

We denote the index of the slowest machine (the bottleneck) to be \( b = \arg \max_i \{ \tau_i \} \). We further define \( \theta \) to be the average throughput in the system and \( C \) to be the average cycle time. Note that \( \theta, C \) and \( m \) are related by Little's law: \( \theta = m/C \). During the development of MVA we denote the average number of jobs in queue at machine \( i \) as \( Q_i \) and make use of the quantity,
\[ t_i = \frac{\tau_i \mu_i + \lambda_i}{\mu_i} \tag{1} \]
which is, of course, simply the constant process time divided by the availability of the machine.

We now review the algorithm used for MVA for one type of job in a tandem line. The basic relation on the fact that the average number of jobs that a marked job sees, upon arrival to a station in steady state, is equal to the average number at the station with one fewer jobs in the system. For a more general treatment involving multiple job classes see Reiser and Lavenberg (1980). Algorithm 1 presents a procedure for computing the average cycle time and the throughput of a tandem line with \( m \) jobs.

**Algorithm 1:** mean value analysis.
\[
\begin{align*}
n &\leftarrow 1 \\
Q_i &\leftarrow 0, \quad \forall i \\
\text{while } n \leq m \text{ do} & \\
& C \leftarrow \sum_i (1 + Q_i) \tau_i \\
& \theta \leftarrow n/C \\
& Q_i \leftarrow \theta(1 + Q_i) \tau_i, \quad \forall i \\
& n \leftarrow n + 1 \\
\text{end \{of while\}}
\end{align*}
\]
The steps within the loop are simple. The first computes the average cycle time by summing the response times at each station. The second computes the throughput from Little's law. The third step computes the average number of jobs seen by an additional job at each station by multiplying the response time at the station by the throughput. At the conclusion of the algorithm, the throughput, average cycle time, and WIP levels are contained in \( \theta, C \) and \( m \), respectively. The algorithm is so simple it can easily be adopted for a spreadsheet.

We will see that the MVA algorithm is asymptotically correct for large WIP levels, regardless of the system parameters, but typically overestimates throughput for small WIP levels and sometimes underestimates it for larger values.
3. An alternate model using regenerative analysis

In this section we derive an alternate model that is explicitly designed for estimating throughput for the DRPO. The name 'approximate regenerative model' comes from the use of a regenerative approach in its derivation. As we show later, ARM is most effective when the failure rates are low with respect to processing rates.

The objective of the ARM is to provide an approximation of $\theta(m)$, the throughput of the DPRO as a function of WIP level. To derive the model, we make two levels of approximation. First, we observe that

$$\theta(m) \approx \frac{\mu_b}{\lambda_b + \mu_b} \hat{\theta}(m)$$

(2)

where $\hat{\theta}(m)$ is the throughput of the system with $m$ jobs when the bottleneck machine is completely reliable.

To explain the second level of approximation, we need to consider the behaviour of the system when the machines are not allowed to fail. Under this condition, the system will eventually reach what we call 'deterministic steady state' (DSS), in which the work in the system will move through the same cycle over and over again. We discuss the rate at which DSS is reached in Section 4.

If the number of jobs in the system is greater than or equal to $M_0$, where $M_0 = \sum \tau_i / \tau_b$, then, in DSS, the bottleneck machine never becomes starved and the DSS cycle is the same regardless of the initial distribution of WIP. We call $M_0$ the 'critical WIP' level. Regardless of the WIP level, all non-bottleneck machines will be periodically starved in DSS.

If $m < M_0$, then a DSS cycle will be reached, but this cycle may depend on the initial distribution of WIP. In this case, the bottleneck, as well as the non-bottleneck machines, will be periodically starved. If we start the system with all jobs at the bottleneck, then the DSS cycle will starve the bottleneck only once during the round trip time of a marked job. Depending on the system and the initial WIP distribution it may be possible to have multiple starvation periods during the round trip time of a marked job. Regardless of whether it is contiguous, the total time of starvation during a round trip time is always $\sum \tau_i - m \tau_b$.

To illustrate the behaviour of a system in DSS, we consider a simple example with three machines where $\tau_i = 1, 1, 2$ for $i = 1, 2, 3$. The critical WIP level for this system is $M_0 = 2$. Table 1 shows the allocation of work in the system at intervals of one time unit for WIP levels of 1, 2 and 4 jobs. Note that in all cases in Table 1, the allocation of WIP

<table>
<thead>
<tr>
<th>$\tau$</th>
<th>$W_1^1$</th>
<th>$W_1^2$</th>
<th>$W_1^3$</th>
<th>$W_2^1$</th>
<th>$W_2^2$</th>
<th>$W_2^3$</th>
<th>$W_3^1$</th>
<th>$W_3^2$</th>
<th>$W_3^3$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>4</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>3</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>0</td>
<td>2</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>3</td>
<td>1</td>
<td>1</td>
<td>3</td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>3</td>
</tr>
<tr>
<td>5</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>3</td>
</tr>
<tr>
<td>6</td>
<td>0</td>
<td>0</td>
<td>2</td>
<td>1</td>
<td>0</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>3</td>
</tr>
<tr>
<td>7</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>3</td>
</tr>
<tr>
<td>8</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>3</td>
</tr>
</tbody>
</table>

$W_i^j$, total number of time units at workstation $i$ at time $t$.

Table 1. Example convergence to DSS.
eventually converges to a DSS cycle, which can be constant or cyclic. When \( m = 1 \),
which is less than \( M_0 \), the bottleneck machine (machine 3) is regularly starved. When
\( m = 2 \) or \( m = 4 \), the bottleneck never starves. Furthermore, when \( m \geq M_0 \), jobs arriving
at the bottleneck always 'see' the same amount of work ahead of them. In the example
of Table 1, in DSS with \( m = 2 \) jobs arriving at the bottleneck encounter zero time units
of work, so they never have to wait. In DSS with \( m = 4 \), jobs always see two time units of
work at machine 3.

If \( m \geq M_0 \), the system throughput will be \( 1/\tau_b \), since the bottleneck is always busy.
Because the bottleneck will act to pace the line, emitting one job every \( \tau_b \) time units and
the non-bottleneck stations can all keep up with this pace, it follows that non-
bottleneck stations will be busy \( \tau_i/\tau_b \) of the time (see Fig. 2).

If \( m < M_0 \) the bottleneck will always starve in DSS and, therefore, the non-
bottleneck stations will be starved even more frequently. If we start this system with all
jobs at the bottleneck, then the DSS cycle will involve exactly one starvation period of
length \( g = \Sigma \tau_i - m \tau_b \) of the bottleneck per round trip of job. The effect on a non-
bottleneck machine is illustrated in Fig. 2. For the purposes of our calculations below,
we will assume that the DSS cycle for the case with \( m < M_0 \) looks like this since it
simplifies the exposition. Since WIP levels below \( M_0 \) are unlikely to be used in practical
situations and since, as we show later, our simple approximation works well, we refrain
from more detailed analysis of this case.

3.1. Throughout above \( M_0 \)

Now, with the concept of DSS in hand, we turn to the problem of approximating
\( \theta(m) \) for the case where \( m \geq M_0 \). This approximation combined with Equation (2) gives
us the ARM approximation of \( \theta(m) \).
We estimate the throughput of the system with a completely reliable bottleneck, \( \delta(m) \), as follows:

\[
\delta(m) \approx \frac{\delta(m)}{\tau_b}
\]  

where \( \delta(m) \) is the fraction of time for which the bottleneck machine is not starved in the system where the bottleneck is completely reliable and the following assumption holds:

(A1): Each time a non-bottleneck machine fails, no other non-bottleneck machine fails until the machine is repaired and DSS is reached.

The accuracy of this approximation depends on the degree to which assumption (A1) is satisfied naturally by the system. If failure rates are low relative to repair rates and rate at which DSS is reached, then assumption (A1) will generally characterize the behaviour of the system. We quantify the extent to which assumption (A1) approximates reality in Section 4.

To use assumption (A1) to derive an expression for \( \delta(m) \), we define a stochastic process \( A = \{A_t; t \geq 0\} \), where

\[
A_t = \begin{cases} 
1 & \text{if work is available at machine } b \text{ at time } t \\
0 & \text{otherwise}
\end{cases}
\]

Assumption (A1) implies that the instants just prior to failures constitute regeneration points for the stochastic process \( A \). Assumption (A1) also simplifies the possible behaviour between these regeneration points, since we need only consider one failure at a time.

Expressing \( \delta(m) \) in terms of \( A \) and using standard results from renewal theory (Karlin and Taylor 1975, p. 203), we can write

\[
\delta(m) = \lim_{t \to \infty} \frac{E \left[ \int_0^t A_t \, dt \right]}{E[T_1]} = \frac{E[P_1]}{E[T_1]} 
\]

where \( T_1 \) is the time of the first regeneration (we assume the system starts uniformly on DSS) and

\[
E[P_1] = E \left[ \int_0^{T_1} A_t \, dt \right]
\]

that is, \( E[P_1] \) is the expected time work is available at the bottleneck during a regenerative cycle.

By defining \( \Lambda = \sum_{i \neq b} \lambda_i \), we compute the expected length of the regenerative cycle as the sum of the expected down time (which depends on which machine fails) plus the expected up time of all machines, which yields

\[
E[T_1] = \sum_{i \neq b} \frac{\lambda_i}{\Lambda} \frac{1}{\mu_i} + \frac{1}{\Lambda}
\]

We approximate the expected time work is available at the bottleneck during a cycle by first noting that while the system is in DSS the time an arriving job will spend in the queue in front of the bottleneck will be,

\[
z = m\tau_b - \sum \tau_i
\]
This time represents a cushion against WIP gaps caused by failures of non-bottleneck machines. Under assumption (A1) there is no interaction between failures, so this WIP cushion is the only protection of the bottleneck against such failures. Specifically, we are acting as though any delay of a job by failure of a non-bottleneck machine is passed along to the bottleneck (i.e. is not erased by the reallocation of WIP caused by failure of a second machine).

To make the details of our calculation explicit, we define $X$ to be a random variable representing the length of an outage at the bottleneck caused by the failure of non-bottleneck machine $i$. Also, we let $U$ be the time of the failure (on our cyclic clock as seen in Fig. 2) and $V$ represent the length of time until the next job will arrive at machine $i$ if $U$ occurs while the machine is idle. We denote the time that machine $i$ is down by $Y$ and note that it is exponentially distributed with a mean time of $1/\mu_i$. Then, given $U$, $V$ and $Y$, we can write the expected value of $X$ as

$$E[X|U,V,Y] = \begin{cases} 
\max \{0, Y-z\} & \text{if } U \in B \\
\max \{0, Y-z-V\} & \text{if } U \in I 
\end{cases}$$

Unconditioning on $Y$ yields

$$E[X|U] = \begin{cases} 
(1/\mu_i) \exp (-\mu_i z) & \text{if } U \in B \\
(1/\mu_i) \exp [-\mu_i (z + V)] & \text{if } U \in I 
\end{cases}$$

Noting that $P(U \in B) = \tau_i/\tau_b$, $P(U \in I) = (\tau_b - \tau_i)/\tau_b$, and that $\{U \in I\}$ implies that $V$ is uniformly distributed on $[0, \tau_b - \tau_i]$, and defining $\beta_i(m)$ to be the expected value of $X$ given that machine $i$ fails yields we can uncondition on $U$ and $V$ to get

$$\beta_i(m) = E[X] = \frac{\exp(-\mu_i z)}{\mu_i \tau_b} \left[ \tau_i + \frac{1}{\mu_i} \left( \frac{1}{\mu_b} - \exp \left[ -\mu_i (\tau_b - \tau_i) \right] \right) \right]$$

(6)

Hence, the expected time work is available at the bottleneck during a regenerative cycle is

$$E[P_i] = \sum_{i \neq b} \frac{\lambda_i}{\Lambda} \left( \frac{1}{\mu_i} - \beta_i(m) \right) + \frac{1}{\Lambda}$$

(7)

Using these expressions for $E[T_i]$ and $E[P_i]$ we can write $\delta(m)$, the fraction of the time that the bottleneck is not starved by non-bottleneck failures, as

$$\delta(m) = \frac{1 + \sum_{i \neq b} \lambda_i \left( \frac{1}{\mu_i} - \beta_i(m) \right)}{1 + \sum_{i \neq b} \frac{\lambda_i}{\mu_i}}$$

(8)

Using Equation (8) with Equations (2) and (3) gives us an estimate of average throughput.

3.2. Throughput Below $M_0$

If the number of jobs is less than the critical WIP level, $M_0$, the bottleneck will periodically starve in DSS. We can still use Equation (2) to estimate $\theta(m)$, but we need to modify Equation (3) to estimate $\delta(m)$. Because the starvation of the bottleneck causes
the production rate in DSS (i.e. with all reliable machines) to be \( m/\sum \tau_i \), we can write the analogue to Equation (3) as

\[
\theta(m) = \varepsilon(m) \frac{m}{\sum \tau_i} \tag{9}
\]

where \( \varepsilon(m) \) represents the fraction of time work is available at the bottleneck under assumption (A1). This assumption again ensures that \( A \) is a regenerative process, which allows us to use renewal theory in a similar fashion to that used previously to derive the following expression for \( \varepsilon(m) \) (see Appendix for details):

\[
\varepsilon(m) = \frac{1 + \sum_{i \neq b} \lambda_i \left( \frac{1}{\mu_i} - \gamma_i(m) \right)}{1 + \sum_{i \neq b} \lambda_i / \mu_i} \tag{10}
\]

where \( \gamma_i(m) \) in Equation (10) is given by

\[
\gamma_i(m) = \frac{1}{m \tau_i + \beta_i / \mu_i} \left[ m(\tau_i + 1/\mu_i) - \exp \left[ -\mu_i(\tau_i - \tau_b) \right] \frac{1}{\mu_i} (m - 1 + \exp(-\mu_i \beta)) \right] \tag{11}
\]

and \( g = \sum \tau_i - m \tau_b \), which is the negative of \( z \) and represents the length of the ‘gap’ seen at the bottleneck.

Using expressions (10) and (11) with Equations (2) and (9) yields an approximation of \( \theta(m) \) for \( m < M_0 \).

4. Robustness of the Models

In the above sections two models for estimating throughput as a function for WIP level in DPRO systems are described: MVA and ARM. Both models are approximations, so the question is now: When can we expect each to work well? Below, we discuss the robustness of each model with respect to parameter values that would be typically encountered in a real production situation and to violations of the underlying assumptions.

4.1. Mean Value Analysis

MVA is exact for systems having exponential processing times and no failures. Hence, we would expect MVA to be robust whenever the coefficients of variation of processing time are near unity (see Bondi and Whitt 1986). An indication of how well MVA will perform is given by how close the effective coefficients of variation are to unity. We can compute an approximate value for these coefficients by assuming that the machine is either busy or up whenever a job arrives. In either case, the arriving job will see the machine up whenever it becomes available for processing. This approximation should be good at moderate and high WIP levels (i.e. \( m > M_0 \)). In the following analysis we drop the subscript indicating the machine number.

Let \( T \) be the total process time (including down time) at the machine, \( N \) be the number of failures that occur, and \( U_i \) be the length of the \( i \)th failure. Note that \( E[N] = \lambda_i \tau_i \) and \( E[U_i] = 1/\mu_i \). Then

\[
E[T] = \tau_i + \sum_{j=1}^{N} U_j
\]

\[
E[T] = \tau_i (1 + \lambda_i / \mu_i) \tag{12}
\]
which is equivalent to Equation (1). Similarly,

\[
E[T^2 | N] = \tau_i^2 + \sum_{j=1}^{N} U_j^2 + 2\tau_i \sum_{j=1}^{N} U_j + \sum_{k \neq j} U_k U_j
\]

\[
E[T^2] = \tau_i^2(1 + \lambda_i/\mu_i)^2 + 2\lambda_i \tau_i/\mu_i^2
\]  

(13)

After some algebra, the squared coefficient of variation is

\[
C_v^2 = \frac{2\lambda}{\tau(\mu + \lambda)^2}
\]  

(14)

Thus, Equation (14) provides a useful check on the suitability of MVA to the DPRO system. If most of the tools have \( C_v^2 \approx 1 \) then MVA should work well. If, on the other hand, \( C_v^2 \gg 1 \) or if \( C_v^2 \ll 1 \) then MVA could have significant error.

Another source of error in the MVA approximation occurs at low WIP levels. This is caused by the fact that MVA computes the average cycle time with one job to be simply \( \sum t_i \), where \( t_i \) is given by Equation (1). The following analysis shows that this value is a lower bound for average cycle time so that \( 1/\sum t_i \) is an upper bound for throughput.

Consider a set of single machines in a tandem closed queueing network with a single job. Let \( T_i \) represent the total time spent at station \( i \) during one cycle and define \( R_i \) to be the total repair time within the \( \tau_i \) processing time. As before, \( E[R_i] = \lambda_i \tau_i/\mu_i \). Also let \( U_i \) be the time of the first failure if the machine is down when the job arrives. Since the repair times are exponential, \( E[U_i] = 1/\mu_i \). Because the machine must be up whenever the job departs, the probability that the machine is down upon arrival after departing \( t \) time units earlier will be

\[
\phi(t) = P\{\text{station } i \text{ is down at time } t \mid \text{it is up at } t=0\}
\]

Solving the Chapman–Kolmogorov equations yields (Ross 1983, p. 150)

\[
\phi(t) = \frac{\lambda_i}{\lambda_i + \mu_i} \left(1 - \exp \left[\left(\frac{\lambda_i + \mu_i}{\mu_i}\right) t\right]\right)
\]  

(15)

Then

\[
T_i = \begin{cases} \tau_i + R_i + U_i & \text{if down upon arrival} \\ \tau_i + R_i & \text{otherwise} \end{cases}
\]

So that

\[
E[T_i | t] = \tau_i(1 + \lambda_i/\mu_i) + \phi(t)/\mu_i
\]

We can obtain \( E[T_i] \) by allowing \( t \) to be a random variable and solving

\[
E[T_i] = E[E[T_i | t]] = \tau_i(1 + \lambda_i/\mu_i) + E[\phi(t)/\mu_i]
\]

Unfortunately, \( \phi \) depends on the random vector, \( T \). However, since Equation (15) is concave everywhere, by Jensen's inequality

\[
E[\phi(T)] \leq \phi(E[T])
\]

Hence, solving the system

\[
\ell_i = \tau_i(1 + \lambda_i/\mu_i) + \frac{1}{\mu_i} \frac{\lambda_i}{\lambda_i + \mu_i} \left(1 - \exp \left[\left(\frac{\lambda_i + \mu_i}{\mu_i}\right) \sum t_i\right]\right)
\]
yields an upper bound for $E[T_i]$. Also, since the smallest value for $\phi(t)$ is zero, $t_i = \tau_i(1 + \lambda_i/\mu_i)$ is a lower bound for $E[T_i]$. Thus

$$\frac{1}{\sum t_i} \leq \bar{\theta} \leq \frac{1}{\sum I_i}$$

so that MVA will always provide an upper bound on throughput for $m = 1$.

Our experience with simulation has shown that the throughput lower bound is typically tighter than the upper bound. Hence, whenever the bounds are far apart and $m \leq M_0$, MVA tends to work poorly.

4.2. The approximate regenerative model

The suitability of the ARM to a particular system depends primarily on the extent to which assumption (A1) holds. One measure of this is the probability that the system is in DSS at the time of a non-bottleneck failure given that it was in DSS at the time of the last non-bottleneck failure. Unfortunately, it is clear that this probability depends on the WIP level (e.g. a system with one job is always in DSS when all machines are working, which is clearly not true for $m > 1$). Since we are interested in characterizing the entire throughput vs. WIP curve, we would like a single suitability measure for the system. We do this by considering a worst case analysis.

To determine the maximum time for the the system to return to DSS following a non-bottleneck failure, we first note the behaviour of the system under these conditions. If (non-bottleneck) machine $i$ fails, then jobs accumulate in front of machine $i$ until it is repaired. When it comes back up, since it works faster than the bottleneck, which is pacing the line, it will eventually work off this backlog. The system will be in DSS when the first job that encounters machine $i$ with an empty queue reaches the bottleneck queue. (Note that DSS is not reached as soon as machine $i$ clears its backlog, because the jobs between machine $i$ and the bottleneck are more closely spaced than in DSS.) Since the time it takes machine $i$ to clear its backlog after a failure is monotonically increasing in the number of jobs in the system, the worst case is when $m = \infty$. In this case, DSS will find the system with an infinite queue in front of the bottleneck and non-bottleneck machines alternating between 0 and 1 jobs, as depicted in Fig. 2.

We will denote by $P_{A_i}$ the probability that the system with a completely reliable bottleneck and $m = \infty$ is in DSS at the time of a failure given that it was in DSS at the time of the last failure. We could compute $P_{A_i}$ exactly for this system, but the discreteness of the jobs greatly complicates the resulting expression. So, instead, we approximate $P_{A_i}$ by making a flow approximation. Formally, we suppose that (non-bottleneck) machine $i$ fails at time 0 and is down for $Y$ time units. Using our flow approximation, work builds up in front of machine $i$ at rate $1/\tau_i$, until time $Y$, after which it depletes this queue at rate $1/\tau_b - 1/\tau_i$. Simple algebra shows that the time machine $i$ will clear its backlog, provided that no further failures occur, can be written

$$T = \frac{\tau_b}{\tau_b - \tau_i} Y$$

Letting $D_i$ represent the time DSS is reached after a failure of machine $i$, assuming, without loss of generality, that the bottleneck machine is the last machine, $n$, and defining $r_i = \sum_{j=1}^{n-i} \tau_i$, for $i < n$ we can write

$$D_i = \frac{\tau_b}{\tau_b - \tau_i} Y + r_i$$
DSS will be reached before the next failure provided that:

1. a failure of non-bottleneck machines other than \( i \) does not occur in the time interval \([0, Y]\); and
2. a failure of any non-bottleneck machine does not occur in the interval \([Y, D_i]\).

Letting \( Z_i \) represent an exponentially distributed random variable with parameter \( \Lambda_i = \sum_{j \neq i} \lambda_j \) and \( Z \) represent an exponentially distributed random variable with parameter \( \Lambda = \sum_{j \neq i} \lambda_j \), we can express the probability that DSS is reached before the next non-bottleneck failure as

\[
\pi_i = P(Z_i > Y)P(Z > D_i - Y)
\]

Substituting the expression for \( D_i \) into this equation and using the fact that \( Y \) is exponentially distributed with parameter \( \mu_i \), we can compute the probability that DSS is reached after a failure of machine \( i \) before the next failure occurs as

\[
\pi_i = \frac{\mu_i}{(\Lambda_i + \mu_i)} \frac{\mu_i (\tau_b - \tau_i) \exp(-\Lambda \tau_i)}{\mu_i (\tau_b - \tau_i) + \Lambda \tau_i} \tag{16}
\]

Observe that \( \pi_i \) has the correct qualitative behaviour. As we would expect, \( \pi_i \to 1 \) as \( \Lambda \to 0 \), since DSS is certainly reached as the failure rates of the non-bottleneck machines approach zero. (Note that \( \Lambda \to 0 \) implies that \( \Lambda_i \to 0 \).) In contrast, \( \pi_i \to 0 \) as \( \tau_b - \tau_i \to 0 \), since a balanced system never erases the deviations caused by failures. These observations suggest that the ARM should work well for systems with

1. failure rates that are low relative to repair rates, and
2. a well-defined bottleneck machine.

Finally, unconditioning on \( i \), we can express our approximation of \( P_{A_i} \) as

\[
P_{A_i} \approx \sum_{j \neq i} \frac{\lambda_j}{\Lambda} \pi_j \tag{17}
\]

We now have a set of criteria for choosing a model. If \( P_{A_i} \) is high while \( C^2 \) is either much greater or much smaller that unity of if \( \Sigma \tau_i \ll \Sigma \delta_i \), then the ARM should be used. On the other hand, if \( C^2 \approx 1 \) while \( P_{A_i} \) is near zero, MVA should be chosen.

5. Comparisons

We now consider the questions of how likely a realistic system is to satisfy the suitability conditions for MVA and ARM and how much these conditions can be violated without severely affecting the accuracy of the models. To do this we compare their performance with data from simulation. We start by comparing the models using "typical" processing time, failure, and repair data. We then compare by setting specific values for \( P_{A_i} \), and \( C^2 \). For each configuration and WIP level combination (i.e. 90 separate cases) we ran 25 replications of a simulation model. Because the variance of the cycle time appeared to increase linearly in the number of jobs, each replicate was run until the number of cycle time observations equalled 40 times the number of jobs. Throughput was estimated using Little's law. The error bars in the figures represent 95% confidence intervals on the mean throughput from the 25 independent observations.
5.1. Comparisons using ‘typical’ parameters

For the comparisons using ‘typical’ parameters, we considered two basic cases with four machines each.

I. Slow machines:

(a) MTBF is three orders of magnitude larger than processing time;
(b) MTTR is two orders of magnitude larger than processing time; and
(c) Typical values would be 1 h processing time with a MTBF of 100 h and a MTTR of 10 h.

II. Fast machines:

(a) MTBF is two orders of magnitude larger than processing time;
(b) MTTR is a single order of magnitude larger than processing time; and
(c) Typical values would be 6 min processing time with a MTBF of 100 h and a MTTR of 10 h.

The actual parameters were generated from uniform distributions about desired means and are shown in Table 2. The processing time characteristics are shown in Table 3. From these characteristics it would appear that MVA should work well for case I since the coefficients of variation are close to unity. Case II, however, has coefficients of variation around 3. Although, in both case I and case II the ARM assumption probability is fairly low, the model appears to work reasonably well. Figures 3 and 4 show the fit of the models against the simulated data. In Fig. 3 we see good agreement.

<table>
<thead>
<tr>
<th>Case</th>
<th>Parameter</th>
<th>Range</th>
<th>Machine</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>I</td>
<td>( \tau )</td>
<td>1–2</td>
<td>1</td>
<td>1.83</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>2</td>
<td>1.45</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>3</td>
<td>1.65</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>4</td>
<td>1.62</td>
</tr>
<tr>
<td></td>
<td>MTBF</td>
<td>80–120</td>
<td>1</td>
<td>107:00</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>2</td>
<td>89:00</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>3</td>
<td>118:00</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>4</td>
<td>93:00</td>
</tr>
<tr>
<td></td>
<td>MTTR</td>
<td>8–12</td>
<td>1</td>
<td>11:70</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>2</td>
<td>9:80</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>3</td>
<td>9:00</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>4</td>
<td>10:20</td>
</tr>
<tr>
<td>II</td>
<td>( \tau )</td>
<td>1–2</td>
<td>1</td>
<td>1.83</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>2</td>
<td>1.45</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>3</td>
<td>1.65</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>4</td>
<td>1.62</td>
</tr>
<tr>
<td></td>
<td>MTBF</td>
<td>800–1200</td>
<td>1</td>
<td>1070:00</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>2</td>
<td>890:00</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>3</td>
<td>1180:00</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>4</td>
<td>930:00</td>
</tr>
<tr>
<td></td>
<td>MTTR</td>
<td>80–120</td>
<td>1</td>
<td>117:00</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>2</td>
<td>98:00</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>3</td>
<td>90:00</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>4</td>
<td>102:00</td>
</tr>
</tbody>
</table>

Table 2. Data for ‘typical’ comparisons.
Table 3. Processing time characteristics for 'typical' comparisons.

<table>
<thead>
<tr>
<th>Case</th>
<th>Machine</th>
<th>$\tau\mu/(\mu + \lambda)$</th>
<th>$C_v$</th>
</tr>
</thead>
<tbody>
<tr>
<td>I</td>
<td>1</td>
<td>2.03</td>
<td>1.07</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>1.61</td>
<td>1.10</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>1.78</td>
<td>0.85</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>1.80</td>
<td>1.06</td>
</tr>
<tr>
<td>Probability $A_1=0.2677$</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>II</td>
<td>1</td>
<td>2.03</td>
<td>3.37</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>1.61</td>
<td>3.48</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>1.78</td>
<td>2.68</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>1.80</td>
<td>3.35</td>
</tr>
<tr>
<td>Probability $A_1=0.2923$</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Figure 3. Comparison of models with simulation data for case I.

Figure 4. Comparison of models with simulation data for case II.
between the simulated data and both models with the ARM fitting somewhat better than MVA for low WIP levels and MVA fitting somewhat better for high WIP levels. In Fig. 4 the ARM fits much better than MVA, although the data is somewhat more noisy. Note that MVA approaches the asymptotic throughput value much more quickly than the simulated data. Note also that the amount of variance in the simulation estimates is large even with up to 22000 observations for a single point indicating a clear need for an analytical method for predicting throughput.

5.2. Comparisons with respect to model assumptions

We conclude our comparisons with a systematic perturbation of model parameters that yield precise values for $P_{A_3}$ and $C_v^2$. (We have chosen these examples such that $C_v$ is the coefficient of variation of all three machines, so that MVA requires only this single suitability measure.) We present seven cases, each labelled according to how well the model assumptions are satisfied in the ARM and MVA, respectively. We consider three levels of satisfaction: excellent (E), where $P_{A_3} = 0.9$ and $C_v = 1.0$, good (G) where $P_{A_3} = 0.6$ and $C_v = 2.0$ (G1) or 0.2 (G2), and bad (B) where $P_{A_3} = 0.05$. No B cases were considered for MVA. We label the seven combinations we generated: EE, GE, BE, GG1, GG2, BG1, and BG2. The parameters for these cases are given in Table 4, while the processing time characteristics are shown in Table 5.

In each case there are three machines, two identical machines and a distinct bottleneck. The processing parameters were chosen to achieve the desired values of $C_v$ and $P_{A_3}$ while maintaining a reasonable 'unbalance' to the system. By this we mean that the ratio of the average process time of the bottleneck to the average process time of the

<table>
<thead>
<tr>
<th>Parameter</th>
<th>EE</th>
<th>GE</th>
<th>GG1</th>
<th>GG2</th>
<th>BE</th>
<th>BG1</th>
<th>BG2</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\tau_1$</td>
<td>1.000</td>
<td>1.000</td>
<td>1.000</td>
<td>1.000</td>
<td>1.000</td>
<td>1.000</td>
<td>1.000</td>
</tr>
<tr>
<td>$\tau_2$</td>
<td>1.000</td>
<td>1.000</td>
<td>1.000</td>
<td>1.000</td>
<td>1.000</td>
<td>1.000</td>
<td>1.000</td>
</tr>
<tr>
<td>$\tau_3$</td>
<td>1.750</td>
<td>1.200</td>
<td>1.500</td>
<td>1.200</td>
<td>1.200</td>
<td>1.200</td>
<td>1.200</td>
</tr>
<tr>
<td>$1/\lambda_1$</td>
<td>658.00</td>
<td>181.80</td>
<td>182.40</td>
<td>17.140</td>
<td>4.067</td>
<td>10.700</td>
<td>1.3340</td>
</tr>
<tr>
<td>$1/\lambda_2$</td>
<td>658.00</td>
<td>181.80</td>
<td>182.40</td>
<td>17.140</td>
<td>4.067</td>
<td>10.700</td>
<td>1.3340</td>
</tr>
<tr>
<td>$1/\lambda_3$</td>
<td>658.00</td>
<td>181.80</td>
<td>182.40</td>
<td>17.140</td>
<td>4.067</td>
<td>10.700</td>
<td>1.3340</td>
</tr>
<tr>
<td>$1/\mu_1$</td>
<td>18.6525</td>
<td>10.0618</td>
<td>21.3336</td>
<td>0.6062</td>
<td>2.1960</td>
<td>8.1492</td>
<td>0.1861</td>
</tr>
<tr>
<td>$1/\mu_2$</td>
<td>18.6525</td>
<td>10.0618</td>
<td>21.3336</td>
<td>0.6062</td>
<td>2.1960</td>
<td>8.1492</td>
<td>0.1861</td>
</tr>
<tr>
<td>$1/\mu_3$</td>
<td>18.6525</td>
<td>10.0618</td>
<td>21.3336</td>
<td>0.6062</td>
<td>2.1960</td>
<td>8.1492</td>
<td>0.1861</td>
</tr>
</tbody>
</table>

Table 4. Data for example perturbations.

<table>
<thead>
<tr>
<th>Case</th>
<th>$C_v$</th>
<th>$P_{A_3}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>EE</td>
<td>1.0</td>
<td>0.90</td>
</tr>
<tr>
<td>GE</td>
<td>1.0</td>
<td>0.60</td>
</tr>
<tr>
<td>BE</td>
<td>1.0</td>
<td>0.05</td>
</tr>
<tr>
<td>GG1</td>
<td>2.0</td>
<td>0.60</td>
</tr>
<tr>
<td>GG2</td>
<td>0.2</td>
<td>0.60</td>
</tr>
<tr>
<td>BG1</td>
<td>2.0</td>
<td>0.05</td>
</tr>
<tr>
<td>BG2</td>
<td>0.2</td>
<td>0.05</td>
</tr>
</tbody>
</table>

Table 5. Suitability measures for example perturbations.
non-bottlenecks is maintained between 1.1 and 1.75. With all these constraints, it is no surprise that the parameters are somewhat atypical. This analysis, therefore, should only be used to judge the effect of relaxing the basic model assumptions.

Figures 5 to 8 provide a comparison of the simulated data and the models. In both the EE and GE cases the ARM fits significantly better than MVA even though $C_v^2 = 1$. This, of course, reflects the fact that even though the coefficient of variation is equal to unity, the process times are not exponential. In the BE case MVA provides a significantly better fit than the ARM, as expected. The ARM fit somewhat better than MVA in the GG1 case and much better in the GG2 case. Neither model fits very well in the BG1 case. Fortunately, there should not be many real systems with such characteristics (few machines have mean times to repair that are of the same order as the mean times to failure). Finally, the ARM model fits significantly better in the BG2 case.

![Graphs comparing ARM, MVA, and simulated data for EE and GE cases.](image)

Figure 5. Comparison of models with simulation data for the EE and GE cases.
Figure 6. Comparison of models with simulation data for the BE and GG1 cases.
Figure 7. Comparison of models with simulation data for the GG2 and BG1 cases.
6. Conclusions

In many realistic production settings the DPRO assumption is a good characterization of reality. However, exact analysis of a closed tandem queueing model of the DPRO system under a CONWIP control strategy is extremely difficult. Simple, but reliable, approximate techniques are needed to predict throughput and cycle times in such systems in order to determine appropriate WIP levels. Both MVA and ARM are simple techniques that can be incorporated in a spreadsheet or straightforward computer program. Moreover, by using the quantitative suitability measures defined here, the user can predict whether or not they are likely to give good results for a particular system.

Our comparisons show that ARM is more robust than MVA. For MVA to yield a throughput versus WIP curve that matches the simulation well, we noted that coefficients of variation had to be quite close to 1. ARM, on the other hand, worked quite well even when the assumption (A1) was strongly violated.

We also observed that in order to get MVA to work significantly better than ARM, we had to choose parameters that are quite unrealistic. For instance, we required average repair times to be of the same order of magnitude as the average times between failures. For the common situation where times between failures are substantially longer than repair times and repair times are substantially longer than processing times (e.g. GG2 Case), ARM works much better than MVA.

In short, the ARM approach to approximating throughput in a DPRO system under a CONWIP control system appears promising. In order to provide appropriate analytical tools for dealing with the range of situations encountered in industry, further work in this direction is needed. In particular, attention must be given to

1. the situation where workstations consist of multiple machines, which fail independently, and
2. the case where jobs are of different types and have significantly different processing times on the various machines.

Although we feel that the general approach used here will still be effective, these cases are considerably more complex and will substantial extension of this research.
Acknowledgment

This work has been supported in part by Grant NO. DDM-8905638 of the National Science Foundation.

References


SOLBERG, J. J., 1980, CAN-Q user’s guide. Report No. 9 (Revised). School of Industrial Engineering, Purdue University, West Lafayette, Indiana.


Appendix

To derive expressions (A1) and (A2) for the case where \( m < M_0 \), we define \( I \) to represent the idle times at machine \( i \) with duration \( \tau_i - \tau_1 \) and \( \bar{I} \) to represent the single idle time at machine \( i \) with duration \( \tau_i - \tau_1 + g \), where \( g = \sum \tau_i - m\tau_1 \). As in section 3, we let \( X \) be the length of the outage at the bottleneck caused by failure of machine \( i \), \( Y \) be the downtime of machine \( i \), \( U \) be the time of failure on the cyclic clock of Fig. 2, and \( V \) be the time until the next job arrives if \( U \) occurs during an idle period (of either duration). Conditioning on \( U \), \( V \) and \( Y \), we can express \( X \) as

\[
E[X|U, V, Y] = \begin{cases} 
Y & \text{if } U \in B \\
\max\{0, Y-V\} & \text{if } U \in I \\
\max\{0, Y-V\} & \text{if } U \in \bar{I}
\end{cases}
\]

Unconditioning on \( Y \) yields

\[
E[X|U, V] = \begin{cases} 
1/\mu_i & \text{if } U \in B \\
(1/\mu_i) \exp(-\mu_i V) & \text{if } U \in I \\
(1/\mu_i) \exp(-\mu_i V) & \text{if } U \in \bar{I}
\end{cases}
\]

Noting that \( P(U \in B) = m\tau_1/(m\tau_1 + g) \), \( P(U \in I) = (m-1)(\tau_b - \tau_1)/(m\tau_1 + g) \), \( P(U \in I) = \tau_i - \tau_1 + g)/(m\tau_1 + g) \), and that \( \{U \in I\} \) implies that \( V \) is uniformly distributed on \([0, \tau_b - \tau_1]\), and \( \{U \in \bar{I}\} \) implies that \( V \) is uniformly distributed on \([0, \tau_b - \tau_1 + g]\), we can define \( \gamma_i(m) \) to be the expected value of \( X \) given that machine \( i \) fails and unconditioned on \( U \) and \( V \) to get

\[
\gamma_i(m) = \frac{1}{m\tau_1 + g} \cdot \frac{1}{\mu_i} \left[ m\tau_i + \frac{1}{\mu_i} \exp \left( -\frac{\mu_i (\tau_b - \tau_1)}{\mu_i} \right) (m-1 + \exp(-\mu_i g)) \right] \quad (A1)
\]

Hence, the expected time work is available at the bottleneck during a regenerative cycle is

\[
E[P_1] = \sum_{i \neq b} \frac{\lambda_i}{\Lambda} \left( \frac{1}{\mu_i} - \gamma_i(m) \right) + \frac{1}{\Lambda}
\]

Using these expressions for \( E[T_1] \) and \( E[P_1] \) we can write \( \delta(m) \), the fraction of the time that the bottleneck is not starved by non-bottleneck failures, as

\[
\delta(m) = \frac{1 + \sum_{i \neq b} \frac{\lambda_i}{\mu_i} \left( \frac{1}{\mu_i} - \gamma_i(m) \right)}{1 + \sum_{i \neq b} \frac{\lambda_i}{\mu_i}} \quad (A2)
\]