MULTI-CLASS MULTI-SERVER QUEUEING NETWORKS FOR PRODUCTION SYSTEMS DESIGN

Pasquale Legato[(a)], Rina Mary Mazza[(b)]

[(a),(b)]Department of Informatics, Modelling, Electronics and Systems Engineering
University of Calabria, 87036 Rende (CS), ITALY

[(a)]legato@dimes.unical.it, [(b)]rmazza@dimes.unical.it

ABSTRACT
Product form queueing networks with multiple customer classes and multiple server stations arise in the design and performance evaluation of stochastic job-shop models of manufacturing, warehousing and logistic systems. Real-size models of this type cannot be solved by the classical Mean Value Analysis (MVA) algorithm, due to its exponential computational complexity in the number of classes. Consolidated (pseudo) polynomial approximation methods have been proposed in literature since some decades. They are based on the transformation of the recursive MVA equations in a system of nonlinear equations to be solved iteratively. Unfortunately these contributions do not cover the case of stations with multiple servers. A new technique based on the idea of class aggregation to cope with the latter case, under a first-come-first-served policy is presented. Preliminary numerical experiments are encouraging upon comparison against the exact MVA algorithm.

Keywords: production systems, decision making, queueing networks, mean value analysis

1. INTRODUCTION

Queueing networks are well consolidated as performance models of systems where congestion phenomena need to be quantified since the earlier stage of system design. This allows to address any successive choice of design by focusing on a restricted number of alternatives. Most consolidated examples of the above systems are assembly, machining, warehousing and other modern flexible manufacturing systems (FMS) in industrial engineering, (Buzacott and Shantikumar 1993), (Zijm, Adan, Buitenhek, and van Houtum 2000), (Fukunari and Malmborg 2009), (McGinnis and Wu 2012). This paper deals with the approximate analytical solution of multi-class queueing network models of the BCMP type (Baskett, Chandy, Muntz, and Palacios 1975) as models of dynamic job-shop systems under stochastic routing among service stations and stochastic duration of services. In this modeling framework, the "processor sharing" service discipline also covered by BCMP networks is not useful and customers (jobs) belonging to different classes are usually serviced according to a first-come-first-served (FCFS) rule. Moreover, service stations are typically equipped with multiple identical servers (Stecke and Kim 1989) (i.e. manufacturing machines and/or automated guided vehicles) each having a fixed-service rate (i.e. not load-dependent). Whatever be the number of servers at each station, to apply the BCMP modeling framework under a FCFS discipline, one has to assume that the average service time is the same for all the job classes and it is referred to an exponential probability distribution function. In the multi-class modeling framework, the exponential assumption for service times over all classes, at any given station, corresponds to a mixture of many short service durations for some classes of jobs with a few long service durations for other job classes. A sample of such a mixture with an average duration equal to one time unit is shown in Figure 1. The exponential assumption is considered suitably "safe" for a first-order estimation of queueing phenomena in automated manufacturing systems where no interruptions during service are expected (Hopp and Spearman 2000), (Manitz 2015). In such a case, a less than exponential variability in service durations is typically estimated (Tempelmeier 2003) and this leads to a (safe) overestimation of the actual queue lengths.

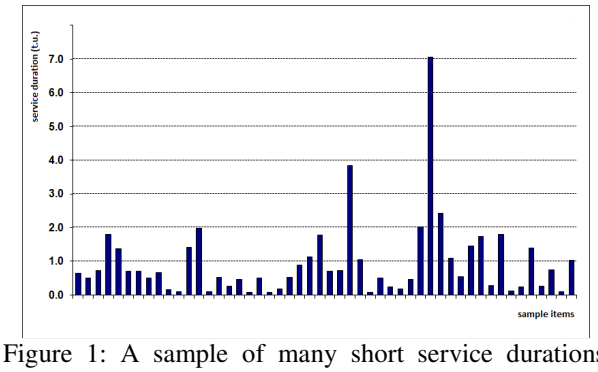

Figure 1: A sample of many short service durations mixed with a few long ones

It is well recognized since a long time that the solution of a BCMP model of a manufacturing system (Solot and Bastos 1988), besides its direct adequateness (Zhuang and Hindi 1990), can also be required as a nested step in approximate iterative methods for solving more

realistic, but non-BCMP models (Jain, Maheshwari, and Baghel 2008). One such pioneering example can be found in (Legato 1993), where a BCMP network (under a FCFS policy) must be solved repeatedly, to estimate the time spent by jobs waiting for admission to the machine-shop section of the whole job-shop model. This reasoning is at the basis of the problem focused in this paper.

In principle, the exact solution of a BCMP model could be obtained by the classical MVA algorithm (Reiser and Lavenberg 1980), but in practice, computational costs become prohibitive for models with more than three or four classes and not restricted to a few customers per class. This occurs because the recursive MVA algorithm is characterized by a dependency between some variables being updated at the current iteration and some others that have been calculated at the previous iteration and whose number rapidly increases with the number of classes. Furthermore, let N_c be the number of class c customers and $\mathbf{N}=(N_1,\ldots, N_c,\ldots,N_C)$ the network population, then the MVA iteration must be repeated from $(0_1,\ldots,0_C)$ to $(N_1,\ldots,N_C)$. Therefore, in a multi-class queueing network the computational complexity of MVA is exponential in both the number of iterations and the number of variables.

For networks with a few service stations and many customer classes, there is an MVA variant proposed ten years after the classic MVA. It is called the Mean Value Analysis by Class (MVAC) algorithm (Conway, de Souza e Silva, and Lavenberg 1989). It exhibits computational costs much lower than MVA, but, as the number of stations increases, MVAC becomes more costly than MVA. On the other hand, since the 1980's, several computer scientists have focused on developing heuristic approximations to the MVA algorithm for BCMP networks aimed at reducing the computational complexity. We have critically examined their work and, in particular, we show in the appendix that one such heuristic algorithm, i.e. the Bard-Schweitzer heuristics (Bard 1979, Schweitzer 1979), may be also derived by a probabilistic argument, i.e. from the assumption that customers circulate within the queueing network according to a semi-Markov process. However, the main contribution of this paper is the proposal of a new approximate, but (pseudo) polynomial MVA algorithm covering the multi-server case under FCFS.

The paper is organized as follows. Background and previous work are provided in section 2. The new approximate algorithm is presented in section 3. Numerical experiments for accuracy validation are given in section 4. Conclusions and issues for further work are in section 5.

2. BACKGROUND AND PREVIOUS WORK

To make the paper self-contained, the MVA algorithm is first resumed.

2.1. Multi-class multi-server MVA algorithm

The following notation is used from now on.

M = number of stations; j =1,...,M as station index.
m_j = number of servers at station j.
C = number of classes; c =1,...,C as class index.
N_c = population of class c customers.
n_c = current class c population, from 0 to N_c.
$\mathbf{N}$ = population vector, i.e. $\mathbf{N}=(N_1,\ldots,N_c,\ldots,NC)$.
V_{jc} = expected number of passages through station j by a class c customer in its trip through the network.
R_j = expected service duration at station j.
$\mathbf{n}$ = current population vector, $n = (n_1,\ldots,n_c,\ldots,n_C)$.
$\mathbf{n}\text{-}1_c = \mathbf{n}$ with one less customer of class c.
$D_{jc}(\mathbf{n})$ = expected sojourn time per visit at station j by a class c customer, under $\mathbf{n}$.
$Q_{jc}(\mathbf{n})$ = expected number of customers at station j, under $\mathbf{n}$.
$T_c(\mathbf{n})$ = expected network throughput for class c customers, under $\mathbf{n}$.
$T_{jc}(\mathbf{n})$ = expected station j throughput for class *c* customers, under $\mathbf{n}$.
$P_j(l|\mathbf{n})$ = long-run (marginal) probability that *l* customers are present at station *j*, under $\mathbf{n}$.

MVA algorithm under FCFS discipline:

For j=1 to M **Do**
  $Q_j(0_1,\ldots,0_C)=0$, $P_j(0_1,\ldots,0_C)=1$
  For l=1 to m_j-1 **Do**
    $P_j(l|0_1,\ldots,0_C)=0$
For $\mathbf{n}=(0_1,\ldots,0_C)$ to $\mathbf{N}=(N_1,\ldots,N_C)$ **Do**
  For c=1 to C **Do**
    For j=1 to M **Do**

(1) $$D_{jc}(\mathbf{n}) = \frac{R_j}{m_j}\left[1+\sum_{d=1}^{C} Q_{jd}(\mathbf{n}-1_c)+\sum_{l=1}^{m_j-1}(m_j-l)P_j(l-1|\mathbf{n}-1_c)\right]$$

  For c=1 to C **Do**

(2) $$T_c(\mathbf{n}) = n_c \,/\, \sum_{j=1}^{M} V_{jc} D_{jc}(\mathbf{n})$$

  For j=1 to M **Do**
    For c=1 to C **Do**

(3) $$Q_{jc}(\mathbf{n}) = V_{jc} T_c(\mathbf{n}) D_{jc}(\mathbf{n})$$

    For l=1 to m_j-1 **Do**

(4) $$P_j(l\,|\,\mathbf{n}) = (1/l)\cdot\sum_{c=1}^{C} R_{jc} V_{jc} T_c(\mathbf{n}) P_j(l-1\,|\,\mathbf{n}-1_c)$$

(5) $$P_j(0\,|\,\mathbf{n}) = 1-\frac{1}{m_j}\left[\sum_{c=1}^{C} R_{jc} V_{jc} T_c(\mathbf{n}) + \sum_{j=1}^{m_j-1}(m_j-l)P_j(l\,|\,\mathbf{n})\right]$$

It is easy to recognize that both the sojourn time equation (1) and the marginal probability equation (4) are responsible for the exponential computational complexity of MVA. In fact, due to the recursive dependence of both $Q_j(\mathbf{n})$ and $P_j(l|\mathbf{n})$ from the corresponding measures related to the population $\mathbf{n}$ with one class c customer removed, the complexity of the algorithm under just a few servers per station is:

$$O(M\cdot C\cdot\prod_{c=1}^{C}(N_c+1)), \qquad (6)$$

both in time and space requirements.

Observe that the marginal probabilities are needed for computing the expected values of the customer sojourn time at each station only in the multi-server case. In the

particular case of just one server per station, the sojourn time (fundamental) equation (1) of the MVA algorithm reduces to the following:

$$D_{jc}(\mathbf{n}) = R_j\left[1+\sum_{d=1}^{C} Q_{jd}(\mathbf{n}-1_c)\right] \quad (7)$$

whenever the user is not interested in computing marginal probabilities, but expected performance measures only (e.g. sojourn times, queue lengths and throughputs for each station).

2.2 Heuristic relationships

If we had an approximate relationship between $Q_{jc}(\mathbf{n}\text{-}1_c)$ and $Q_{jc}(\mathbf{n})$ and, furthermore, a second relationship between $P_j(l\text{-}1|\mathbf{n}\text{-}1_c)$ and $P_j(l\text{-}1|\mathbf{n})$, then we would be able to circumvent the recursive nature of the MVA algorithm. This recursion requires the "**For** $\mathbf{n}=(0_1,\ldots,0_C)$ to $\mathbf{N}=(N_1,\ldots,N_C)$ **Do**" loop which, in turn, is responsible of the exponential complexity in computation. Hence, at the price of introducing an approximation, one may collect equations (1), (2), (3) and (4) into a set of nonlinear equations to be solved simultaneously (Ortega and Rheinboldt 1970) under the (fixed) population **N**. The previous work that appeared in literature immediately after the earlier publication of the MVA algorithm has been guided by this idea and is resumed in a unifying view. Let us start with the identity:

$$\frac{Q_{jc}(\mathbf{n}-1_c)}{n_c-1}-\frac{Q_{jc}(\mathbf{n})}{n_c}=\frac{Q_{jc}(\mathbf{n}-1_c)}{n_c-1}-\frac{Q_{jc}(\mathbf{n})}{n_c} \quad (8)$$

valid for whatever station-class couple. By introducing a measure (δ) of the change in the fraction of the total number of customers found in station j resulting from the removal of one class c customer out of the (current) network population **n**:

$$\delta_{jc}(\mathbf{n}) \triangleq \frac{Q_{jc}(\mathbf{n}-1_c)}{n_c-1}-\frac{Q_{jc}(\mathbf{n})}{n_c} \quad (9)$$

a relationship amenable to heuristic particularizations of the δ measure is obtained:

$$Q_{jc}(\mathbf{n}-1_c) = (n_c-1)[(Q_{jc}(\mathbf{n})/n_c)+\delta_{jc}(\mathbf{n})] \quad (10)$$

Clearly, the first option consists in setting:

$$\delta_{jc}(\mathbf{n}) = 0 \quad (11)$$

thus obtaining a proportionality assumption ($(n_c\text{-}1/n_c)$ as proportionality factor) on the relationship between $Q_{jc}(\mathbf{n}\text{-}1_c)$ and $Q_{jc}(\mathbf{n})$. This is known as the Bard-Schweitzer (BS) heuristic relationship (Bard 1979, Schweitzer 1979). According to these authors, one has to further assume that removing one class c customer does not affect the expected queue length of customers belonging to different classes, hence:

$$Q_{jd}(\mathbf{n}-1_c) = Q_{jd}(\mathbf{n}), \quad d=1,\ldots,C; d\neq c \quad (12)$$

Hence, the BS approximation to equation (7) of the exact MVA algorithm for (fixed rate) single server stations is obtained:

$$D_{jc}(\mathbf{n}) = R_j\left[1+\frac{n_c-1}{n_c}Q_{jc}(\mathbf{n})+\sum_{d=1,d\neq c}^{C} Q_{jd}(\mathbf{n})\right] \quad (13)$$

Provided that marginal probabilities are not required but only expected performance measures are required, formula (13) is all the user needs to define a (fixed-point) iterative heuristic algorithm (called BS_MVA) for queueing networks with fixed-rate single-server stations only. This because the expected system throughput equation (2) can be inserted in the formula for the expected station queue length:

$$Q_{jc}(\mathbf{n}) = V_{jc}T_c(\mathbf{n})\,D_{jc}(\mathbf{n}) \quad (14)$$

thus obtaining the following equation:

$$Q_{jc}(\mathbf{n}) = n_c \cdot V_{jc}D_{jc}(\mathbf{n})/\sum_{i=1}^{M} V_{ic}D_{ic}(\mathbf{n}) \quad (15)$$

Equations (13) and (15) are at the basis of a fixed-point iteration algorithm.

The computational complexity of BS_MVA is O(*MC*) in both space and time (per iteration) requirements. An extensive theoretical study on the existence and uniqueness of the solution of BS_MVA, as well as on convergence, has been carried out in (Pattipati, Kostreva, and Teele 1990). In particular, the existence of the solution is established for monotonic, but single-class networks, i.e. networks where the service rates are monotonically non decreasing functions of the number of customers at the stations. Uniqueness and convergence results have been obtained only under the limiting condition that the number of customers of each class grows to infinity.

An improvement over BS_MVA may be based on the assumption that the change in the fraction of class r customers present at station *j* resulting from the removal of one class *c* customer is constant around the current population vector **n**. This corresponds to replacing the null setting (11) by the constant setting:

$$\delta_{jc}(\mathbf{n}) = \delta_{jc}(\mathbf{n}-1_c) \quad (16)$$

Hence, at the price of introducing a nested fixed-point iteration within the BS_MVA (to estimate the constant δ measure), the so-called Linearizer heuristics, first proposed by Chandy and Neuse (1982), has been obtained. The new assumption on the $\delta_{jc}(\mathbf{n})$ yields an improvement over the BS_MVA algorithm (called Lin_MVA) at the expense of an acceptable increase of the space complexity $O(MC^2)$ and the time (per

iteration) complexity $O(MC^3)$. Some years later, Zahorjan, Eager and Sweillam (1988) proposed working with cumulative (i.e. over all classes rather than per class) queue lengths in formula (10). This allows reducing the space complexity of Lin_MVA from $O(MC^2)$ to O(MC) and time/iteration complexity from $O(MC^3)$ to $O(MC^2)$, without significantly affecting the accuracy of the algorithm. They were not able to establish formal convergence results for the original Lin_MVA algorithm, nor for their own variant to it. Since then, some other variants to both BS_MVA and Lin_MVA and more sophisticated implementations have been proposed (Wang and Sevcick 2000), (Wang, Sevcick, Serazzi, and Wang 2008), but always restricted to fixed-rate services in single-server stations.

To the focus of this paper, (fixed-rate services in multi-server stations) it is worth mentioning that using the following further assumption:

$$P_j(l-1 \mid \mathbf{N}-1_c) = P_j(l-1 \mid \mathbf{N}) \tag{17}$$

within equation (4) allows to define a straightforward version of Lin_MVA capable of solving networks with load-dependent (thus multi-server) service rate stations. Unfortunately, this choice has been exploited by Krzesinski and Greyling (1983), but indicated as the source of several failures for Lin_MVA (i.e. convergence failure, convergence errors and numerical instabilities). Among more successful proposals, it is worth recalling that, before presenting the Linearizer heuristics, Neuse and Chandy (1981) introduced the SCAT algorithm for the approximate solution of multi-class queueing networks with load-dependent service rate stations. SCAT was based on the idea of reconstructing the profile of the distribution of the marginal probability of having one customer at station j from the estimate of the mean queue length at the same station. To this purpose, Chandy and Neuse were assigning the whole marginal probability "mass" to the first two integer values neighboring the estimated (generally fractional) value of the mean queue length. This idea was later refined by Akyildiz and Bolch (1988), who proposed to scatter the assignment of probability mass to a wider range of neighboring integer values. Their scattering was carried out according to a (pseudo) normal distribution function. Akyildiz and Bolch (1988) were able to achieve significant improvements upon the original SCAT in some numerical instances. Unfortunately, other numerical evidence for not heavy-loaded networks states that the profile of marginal queue length probability strongly departs from a normal-like one.

More recently Suri, Sahu and Vernon (2007) have pursued the idea of multiplying $Q_{jc}(\mathbf{n}\text{-}1_c)$ by a correction factor within the BS formula (13), in order to cope with the multi-server case. Their factor is aimed to capture the reduction of the expected queue length resulting from the presence of many servers instead of one at the same station. The above factor is defined as an empirical function of both the number of servers and their utilization factor for each station at hand. Unfortunately, this is not appealing for the applications of generic queueing networks in real practice.

3. A NEW APPROXIMATION

Our approach for reducing the computational complexity of the multi-class multi-server MVA from exponential to (pseudo) polynomial when computing both queue lengths and marginal probability distributions in multi-server networks under the FCFS discipline is presented in this section.

Let us introduce $P_j(l|\mathbf{N})$, l=1,…,N as the probability that l customers are present at station j under a single class population of $N=N_1+N_2\ldots+N_C$ customers circulating within the queueing network. The idea is that of using $P_j(l\text{-}1|\mathbf{N}\text{-}1)$ in place of $P_j(l\text{-}1|\mathbf{N}\text{-}1_c)$ l=1,…,N to get an approximate sojourn time equation for an MVA algorithm which adopts the BS approximation for queue lengths. To this purpose, the concept of representative class is now introduced.

We associate a single-class network to a multi-class queueing network characterized by a population vector $\mathbf{N}=[N_1,\ldots,N_c]$, a matrix of visits $\mathbf{V} = [V_{jc}, j=1,\ldots,M; c=1,\ldots,C]$ and a vector of service requests $\mathbf{R} = [R_j, j=1,\ldots,M]$. The single class, here called "representative class", is defined by the following formulas:

$$N = \sum_{c=1}^{C} N_c \tag{18a}$$

$$V_j = \sum_{c=1}^{C} V_{jc} T_c(\mathbf{N}) \Big/ \sum_{c=1}^{C} T_c(\mathbf{N}) \qquad j = 1,\ldots,M \tag{18b}$$

The expected throughput values per class, $T_c(\mathbf{N})$, required by the (aggregation) formulas (18) to define the representative class are computed iteratively by *Large*, a fixed-point procedure based on the BS approximation for queue lengths coupled with the marginal probabilities w.r.t. the representative class:

Procedure: *Large*

Use $P_j(l \mid N-1) \qquad N = 1,\ldots,N;\ j = 1,\ldots,M$

Initialize $D_{jc}(\mathbf{N}) = R_j \quad c = 1,\ldots,C;\ j = 1,\ldots,M$

Repeat

For c=1 to C **Do**

(19) $T_c(\mathbf{N}) = N_c \Big/ \sum_{i=1}^{M} V_{ic} D_{ic}(\mathbf{N})$

For j=1 to M **Do**

(20) $Q_{jc}(\mathbf{N}) = V_{jc} T_c(\mathbf{N}) D_{jc}(\mathbf{N})$

(21) $$D_{jc}(\mathbf{N}) = \frac{R_j}{m_j}\Big[1 + \frac{N_c - 1}{N_c} Q_{jc}(\mathbf{N}) + \sum_{d=1, d\neq c}^{C} Q_{jc}(\mathbf{N}) + \sum_{l=1}^{m_j - 1} (m_j - l) P_j(l-1 \mid N-1)\Big]$$

Until convergence upon $D_{jc}(\mathbf{N}), \quad c = 1,\ldots,C;\ j = 1,\ldots,M$

The representative class and the *Large* procedure are used in the iterative algorithm called *Approx*, presented in the following.

Approx is based on the following reasoning. If we had the exact throughput values $T_c(\mathbf{N})$ c=1,…,C then the expected number of visits to define the representative class would be determined from (18b) once and for all. Instead, since this is not the case and moreover the above throughputs depend in turn from the marginal probabilities of the representative class, we have to resort to a nested fixed-point algorithm where both the expected throughputs and the marginal probabilities are iteratively refined. Thus, *Approx* requires an initial estimate of the marginal probabilities of the representative class at each station and then it iteratively updates the initial estimate until convergence is achieved. *Large* operates as a nested "repeat until" statement. Given the current estimate of the marginal probabilities, *Large* computes the corresponding expected throughput values per class which are needed to update the current parameters defining the representative class.

Procedure: *Approx*

Initialize $P_j(l\text{-}1\|N\text{-}1)$, l=1,…,$m_j$-1; j=1,…,M
Repeat
1. Compute class throughput by Large
2. Define the representative class
3. Solve the multi-server network under the representative class, by polynomial single class (polynomial) MVA
4.Update $P_j(l\text{-}1\|N\text{-}1)$, l=1,…,$m_j$-1; j=1,…,M
Until *convergence upon* $P_j(l\text{-}1\|N\text{-}1)$, l=1,…,$m_j$-1; j=1,…,M
Return *expected performance measures per class and station*

Returned performance measures for each (original) class of customers, e.g. expected throughputs and queue lengths at each station, are computed by the following formulas:

$$T_{jc}(\mathbf{N}) = V_{jc}N_c \Big/ \sum_{i=1}^{M} V_{ic}D_{ic}(\mathbf{N}) \qquad (22)$$

$$Q_{jc}(\mathbf{N}) = T_{jc}(\mathbf{N}) \cdot D_{jc}(\mathbf{N}) \qquad (23)$$

where $D_{jc}(\mathbf{N})$ values, for j=1,…,M and c=1,…,C, are those returned by *Large* after convergence.

Computational complexity of *Approx* is that of an exact single-class MVA, i.e. $O(m_j MN)$, because step 3. is more costly than step 1. under the BS approximation.

3.1 Convergence

In a large number of numerical experiments we have always observed a very robust behavior. Just a few iterations (outer iterations) are usually needed to determine the marginal probabilities for the representative class at each station. Vice versa, for the nested procedure (*Large*), we have observed that at most some tens of (inner) iterations are needed, in the worst case, to achieve convergence with the initial estimate of the marginal probabilities. But, after the first updates of the marginal probabilities (i.e. the first outer iterations) the number of inner iterations decreases significantly. Typically, it ranges from almost ten to no more than twenty and it remains constant or decreases further during the next few outer iterations. Nevertheless, in principle, we cannot exclude the existence of some (pathological) cases under heavy-loaded networks in which inner, outer or both convergence conditions are not achieved.

4. NUMERICAL EXAMPLES

In this section *Approx* is applied to the solution of the two queueing network models of the job-shop systems shown in Figure 2 and Figure 3. Each station of the job shop is equipped with a pool of identical facilities (see the numbered circles) bearing a common input buffer whose size is big enough to prevent the occurrence of a full buffer. Services are provided according to a FCFS discipline and their individual duration cumulated over all classes well fits an exponential probability distribution. Multiple types of jobs circulate within the network according to a routing path described by a different Markov chain for each different type of jobs. The resulting network configuration is known as Central Server Model (CSM) to highlight the role of the central station within the network topology.

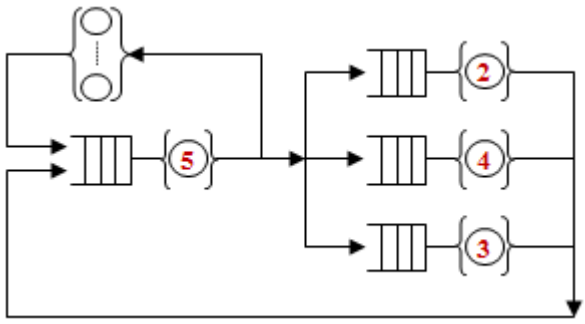

Figure 2: CSM with a pure delay station modelling the delay between finished jobs and new jobs

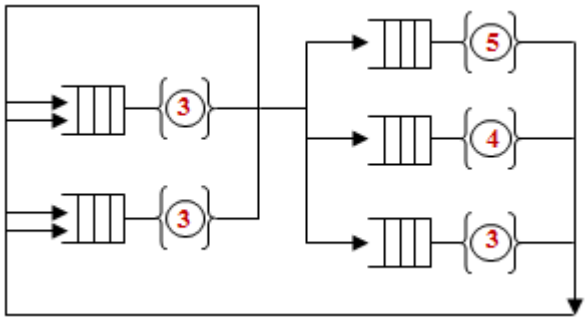

Figure 3: CSM with two central stations and no delay between finished jobs and new jobs

A concrete example of a CSM for an FMS is now given according to the authors' past experience with a German automotive industry in Salzgitter. The material handling system is represented as the central station whose multiple servers are automated guided vehicles (carts) which transfer the workpieces (parts) among the

manufacturing centers. Each manufacturing center is a peripheral station whose servers are the flexible machine tools and the queue is the (local) input storage area. The (local) output storage areas of the manufacturing centers are coalesced into the unique queue of the central (transport) station. A multi-class model must be used when different types of parts have non-compatible geometrical properties and, thus, require different types of support devices (pallets). The parts are mounted on pallets and carried by carts. They follow different routes in the network, according to the class to which they belong. Finally, a fixed number of circulating pallets is assumed and modeled as the fixed multi-class population of customers of the queueing network. The model is closed since no pallet enters the system from outside, nor leaves it; rather, a finished part is removed from the pallet and immediately replaced by a new (raw) part or after some delay. In the latter case, the related average delay is captured by the pure delay station in the CSM (see Figure 2); in the former, the pure delay station is short-circuited.

The numerical results presented in the next two subsections refer to CSM configurations under a few stations and a few classes with a limited number of customers per class. This allows a relatively fast exact solution by the MVA algorithm. Moreover, from the probabilistic argument at the basis of the derivation of our approximation in the Appendix, it should be clear that our experiments under a low-moderate load condition at the central station coupled with a heavy load at one or more peripheral stations should correspond to a worst-case validation.

4.1 Results from the first model

The first set of numerical results presented here is referred to the CSM with a finite source of customers (Figure 1). There are 3 peripheral stations (station 3, station 4 and station 5), 1 central station (station 2) and 3 classes of customers. The population of class 1 is 12, 15 for class 2 and 18 for class 3; therefore, 45 servers are at the pure delay station (station 1), which is visited only once per customer passage through the system. The average delay time differs per class (40 t.u. for class 1, 60 t.u. for class 2 and 80 t.u. for class 3). The central station is equipped with 5 servers, the first peripheral with 2, the second peripheral with 4 and the last peripheral with 3. Using the average number of visits and the average service time per visit shown in Table 1, we tune (by output results) a set of (server) utilization factors ranging from 67% at the first peripheral station (station 3) to 95% at the second peripheral station (station 4). The expected queue lengths per class and the expected throughput per class at each station are reported in Tables 3a-3c. The same performance indices computed by the exact solution of the queueing network returned by the classical MVA algorithm are in these tables as well. The comparison between approximate and exact results for the average queue lengths shows a level of accuracy which is widely acceptable in practice; this accuracy is even higher for the throughput results.

It is well known that queueing approximations are less accurate in the case of a specific station acting as a strong system bottleneck within an unbalanced and heavily loaded system. So, we show in Tables 4a-4c a second set of results obtained under the condition that the second peripheral station is utilized at 99% while the first is around 83%. The new condition has been obtained by reducing to a half the expected delay per class at the pure delay station. From Tables 4a-4c one may recognize that the only significant loss of accuracy (about 10%) arises in the evaluation of the average queue length at station 4. This is the station that has been forced, by our input data setting, to become a very strong bottleneck (utilization level = 99%) for the entire network. Fortunately, we believe that in this case a more accurate analytical evaluation of the average queue length is unnecessary under a similar system bottleneck.

Table 1: Remaining data of the first model

Class c_i	Service time per station s_i case 1/case 2				
	s_1	s_2	s_3	s_4	s_5
c_1	40/20	0.5	1.0	2.0	1.5
c_2	60/30	0.5	1.0	2.0	1.5
c_3	80/40	0.5	1.0	2.0	1.5
Class c_i	Visits per station				
c_1	1	9	1	3	4
c_2	1	7	3	2	1
c_3	1	5	1	2	1

Table 2: Remaining data of the second model

Station	Visits per class c_i				Service times	
	c_1	c_2	c_3	c_4	case 1	case 2
1	5	7	0	0	0.10	0.35
2	0	0	12	11	0.25	0.25
3	1	3	4	2	0.70	0.50
4	2	1	4	3	0.50	0.30
5	1	2	3	5	0.20	0.20

Table 3a: Class 1 results of the first model – case 1

Station	Throughput		Queue length	
	exact	approx	exact	approx
1	0.1982	0.1977	7.9294	7.9105
2	1.7841	1.7798	0.9079	0.9026
3	0.1982	0.1977	0.2575	0.2583
4	0.5947	0.5932	1.4780	1.4910
5	0.7929	0.7910	1.4269	1.4374

Table 3b: Class 2 results of the first model – case 1

Station	Throughput		Queue length	
	exact	approx	exact	approx
1	0.2019	0.2016	12.115	12.098
2	1.4134	1.4115	0.7199	0.7197
3	0.6057	0.6049	0.7757	0.7810
4	0.4038	0.4032	1.0161	1.0247
5	0.2019	0.2016	0.3726	0.3759

Table 3c: Class 3 results of the first model – case 1

Station	Throughput		Queue length	
	exact	approx	exact	approx
1	0.1983	0.1981	15.868	15.853
2	0.9918	0.9908	0.5055	0.5071
3	0.1983	0.1981	0.2582	0.2596
4	0.3967	0.3963	1.0008	1.0094
5	0.1983	0.1981	0.3663	0.3699

Table 4a: Class 1 results of the first model – case 2

Station	Throughput		Queue length	
	exact	approx	exact	approx
1	0.2334	0.2266	4.6688	4.5325
2	2.1009	2.0396	1.1349	1.0640
3	0.2334	0.2266	0.4368	0.4162
4	0.7003	0.6798	3.5357	3.8191
5	0.9337	0.9065	2.2236	2.1680

Table 4b: Class 2 results of the first model – case 2

Station	Throughput		Queue length	
	exact	approx	exact	approx
1	0.2886	0.2837	8.6592	8.5112
2	2.0204	1.9859	1.0918	1.0399
3	0.8659	0.8511	1.5433	1.5344
4	0.5772	0.5674	2.9824	3.2166
5	0.2886	0.2837	0.7231	0.6977

Table 4c: Class 3 results of the first model – case 2

Station	Throughput		Queue length	
	exact	approx	exact	approx
1	0.3135	0.3088	12.543	12.352
2	1.5679	1.5440	0.8490	0.8123
3	0.3135	0.3088	0.5817	0.5677
4	0.6271	0.6176	3.2411	3.5072
5	0.3135	0.3088	0.7847	0.7601

4.2 Results from the second model

In the second model (Figure 3) the pure delay station has been eliminated (i.e. short-circuited), two central stations (station 1 and station 2) are coupled with three peripheral stations (stations 3-5) and the number of customer classes has been increased from 3 to 4. Class 1 population is 12, class 2 is 10, class 3 is 6 and class 4 is 8. There are 3 servers in each of the two central stations, 4 at the first peripheral, 3 at the second and 2 at the last one. All the customers visit all the peripheral stations, but only class 1 and class 2 customers visit the first central station and, in turn, only class 3 and class 4 customers visit the second central station. In terms of the FMS of reference, this means that some carts are dedicated to some types of parts and other carts to other types of parts.

Using the data in Table 2, a double strong bottleneck condition in a heavily loaded system is obtained. In fact, the first and second peripheral stations result to be utilized both at 99% and the less loaded station (central station 1) is at 72%. The results in Tables 5a-5d confirm the good accuracy achieved in practice by our approximate algorithm. Finally, we eliminate one server from each of the two central stations and modify service times as shown in Table 2 (case 2 column). This leads to moving the double bottleneck condition from the two peripheral stations to the two central stations (the first central at 99% and the second at 97%, the first peripheral at 83% and the second at 69%). Results are in Tables 6a-6d. As expected, the accuracy does not depend from the position of the bottleneck stations but, more importantly, no further loss of accuracy is observed w.r.t. the previously discussed results from the first model (where only one bottleneck station was forced to exist by our input data tuning).

Table 5a: Class 1 results of the second model – case 1

Station	Throughput		Queue length	
	exact	approx	exact	approx
1	7.0665	6.9051	0.7665	0.7288
2	0.0000	0.0000	0.0000	0.0000
3	1.4133	1.3810	4.4237	4.2038
4	2.8266	2.7620	6.3748	6.6603
5	1.4133	1.3810	0.4348	0.4069

Table 5b: Class 2 results of the second model – case 1

Station	Throughput		Queue length	
	exact	approx	exact	approx
1	5.7237	5.4921	0.6218	0.5802
2	0.0000	0.0000	0.0000	0.0000
3	2.4530	2.3537	6.7243	7.0202
4	0.8176	0.7845	2.1540	1.9390
5	1.6353	1.5691	0.4996	0.4605

Table 5c: Class 3 results of the second model – case 1

Station	Throughput		Queue length	
	exact	approx	exact	approx
1	0.0000	0.0000	0.0000	0.0000
2	2.7426	2.7246	0.8472	0.8336
3	0.9142	0.9082	2.7259	2.7475
4	0.9142	0.9082	2.2157	2.2180
5	0.6856	0.6811	0.2111	0.2007

Table 5d: Class 4 results of the second model – case 1

Station	Throughput		Queue length	
	exact	approx	exact	approx
1	0.0000	0.0000	0.0000	0.0000
2	4.8636	4.8405	1.4846	1.4632
3	0.8843	0.8801	2.7220	2.6814
4	1.3264	1.3201	3.1335	3.2170
5	2.2107	2.2002	0.6599	0.6382

Table 6a: Class 1 results of the second model – case 2

Station	Throughput		Queue length	
	exact	approx	exact	approx
1	3.1715	3.1429	10.474	10.567
2	0.0000	0.0000	0.0000	0.0000
3	0.6343	0.6285	0.3374	0.3306
4	1.2686	1.2571	0.4091	0.4006
5	0.6343	0.6285	0.7787	0.7008

Table 6b: Class 2 results of the second model – case 2

Station	Throughput		Queue length	
	exact	approx	exact	approx
1	2.5427	2.5287	8.4283	8.5155
2	0.0000	0.0000	0.0000	0.0000
3	1.0897	1.0837	0.5776	0.5661
4	0.3632	0.3612	0.1176	0.1159
5	0.7264	0.7224	0.8764	0.8023

Table 6c: Class 3 results of the second model – case 2

Station	Throughput		Queue length	
	exact	approx	exact	approx
1	0.0000	0.0000	0.0000	0.0000
2	3.5358	3.4282	3.9019	4.1091
3	1.1786	1.1427	0.6233	0.5910
4	1.1786	1.1427	0.3799	0.3610
5	0.8839	0.8570	1.0947	0.9387

Table 6d: Class 4 results of the second model – case 2

Station	Throughput		Queue length	
	exact	approx	exact	approx
1	0.0000	0.0000	0.0000	0.0000
2	4.3407	4.2689	4.9401	5.1395
3	0.7892	0.7761	0.4199	0.4060
4	1.1838	1.1642	0.3826	0.3695
5	1.9730	1.9404	2.2572	2.0848

5. CONCLUSIONS

The proposed fixed-point algorithm seems to be a practical alternative to the classical MVA algorithm for large multi-class queueing networks with multi-server stations. It applies under a FCFS discipline for the exponentially distributed service durations. It does not require restrictive assumptions on the probabilistic shape of the marginal queue length at each station, nor complicated tuning of empirical functions. This should encourage using it in real practice.

Preliminary validation experiments against the exact solution provided by the MVA algorithm state a good accuracy on queueing network models of job-shop systems. The complete exact results of our instances highlight that accuracy lies in the slight variability of the customer sojourn time per class, under a FCFS discipline. The different number of visits to the same station by customers belonging to different classes is responsible for the variability of the waiting time per class. The different average number of visits per class produces different sampling patterns according to which the queue length at that station is observed and suffered by arriving customers.

Polynomial complexity per iteration has been obtained by aggregating all the customer classes into a unique representative class. Marginal queue length probabilities are computed for the representative class at each station by using the (polynomial) single-class MVA algorithm. After a disaggregation step, the expected performance measures per customer class are obtained. A formal proof of convergence seems difficult and even unlikely, considering the past efforts put into this type of iterative schema. Failure in convergence cannot be excluded for pathological instances yet to be classified. Refinements of the algorithm and extensive numerical experiments are also the subject of a future work.

APPENDIX

Here a probabilistic argumentation on the BS heuristics for queue lengths is provided.

Introducing $PF_{jc}(\mathbf{N})$ as the probability of finding one class c customer at station j (i.e. one out of class c population) given that it circulates in a network together with other N_1, N_2, …N_{c-1}, …N_C other customers, we assume that:

$$Q_{jc}(\mathbf{N}) = Q_{jc}(\mathbf{N}-1_c) + PF_{jc}(\mathbf{N}), \quad c = 1,...,C; \quad j = 1,...,M. \tag{24}$$

Relationship (24) lies on the hypothesis that adding a unique class c customer to a network population $\mathbf{N}$-1_c does not produce the effect of redistributing preexisting customers in station j among other stations and, therefore, it does not affect the preexisting average queue lengths. Rather, on the long run we expect a very limited and purely additive marginal contribution to queue lengths. Precisely, this contribution is uniquely determined by the average proportion of the time spent at each queue with respect to the average time spent within the entire network by the (marginal) customer just added.

At this point, we estimate $PF_{jc}(N)$ as follows:

$$PF_{jc}(\mathbf{N}) = V_{jc}D_{jc}(\mathbf{N}) / \sum_{i=1}^{M} V_{ic}D_{ic}(\mathbf{N}), \quad c = 1,...,C; j = 1,...,M. \tag{25}$$

The approximation underlying relationship (25) consists in evaluating the average time spent over all visits at station *j* by a class *c* customer as the product of the average number of visits multiplied by the average time spent per visit. This is not exact, unless the customer circulates within the network according to a semi-Markov process (Cinlar 1975) and, as a consequence, the sojourn time per visit by any customer at any given station is independent from the specific visit (e.g. the first, the second…or the last). Only under this independence assumption the numerator of formula (25) corresponds to the exact evaluation of the cumulative (i.e. over all visits) sojourn time spent by a class c customer at station j during its lifetime within the queueing network, while the denominator corresponds to the average sojourn time within the entire network. Otherwise the correlation among the sojourn time per visit arises, thus reducing formula (25) to an approximation formula. Actually, we have verified by simulation experiments that a significant correlation may exist among the sojourn times experienced by a class *c* customer at station *j*. This qualifies our semi-Markov assumption as an approximation, but at the same time, it indicates the way to improve our approximation. To give some details, by simulating the CSM model in Figure 2, we have observed point

estimates of the correlation factor among sojourn times experienced by a specific (sample) customer at the second peripheral station (station 4). They typically range from 0.15 to 0.25 under a very high-loaded central station coupled with a moderate-loaded station 4. On the other hand, the same correlation factor typically ranges from 0.55 to 0.80 under a moderate-loaded central station coupled with a high-loaded station 4.

After this discussion, the BS heuristics is derived here using some algebra.

By our approximate relationship:

$$Q_{jc}(\mathbf{N}-1_c) = Q_{jc}(\mathbf{N}) - V_{jc}D_{jc}(\mathbf{N}) / \sum_{i=1}^{M} V_{ic}D_{ic}(\mathbf{N}), \quad \forall (j,c) \tag{26}$$

recalling the sojourn time equation of MVA for a (fixed rate) single-server station under FCFS:

$$D_{jc}(\mathbf{N}) = R_j[1 + Q_{jc}(\mathbf{N}-1_c) + \sum_{d=1,d\neq c}^{C} Q_{jd}(\mathbf{N}-1_c)], \quad \forall (j,c) \tag{27}$$

we assume the following:

$$\sum_{d=1,d\neq c}^{C} Q_{jd}(\mathbf{N}-1_c) = \sum_{d=1,d\neq c}^{C} Q_{jd}(\mathbf{N}) \tag{28}$$

Hence, it suffices inserting first (26) in (27):

$$D_{jc}(\mathbf{N}) = R_j\left[1 + Q_{jc}(\mathbf{N}) - \frac{\forall_{jc} D_{jc}(\mathbf{N})}{\sum_{i=1}^{M} V_{ic}D_{ic}(\mathbf{N})} + \sum_{d=1,d\neq c}^{C} Q_{jd}(\mathbf{N}-1_c)\right] \tag{29}$$

and then repeating equation (15), with reference to population **N**:

$$Q_{jc}(\mathbf{N}) = N_c \cdot V_{jc}D_{jc}(\mathbf{N}) / \sum_{i=1}^{M} V_{ic}D_{ic}(\mathbf{N}), \quad \forall (j,c) \tag{30}$$

in equation (29) to obtain the BS heuristic approximation:

$$\begin{aligned} D_{jc}(\mathbf{N}) &= \\ &= R_j\left[1 + N_c \cdot \frac{V_{jc}D_{jc}(\mathbf{N})}{\sum_{i=1}^{M} V_{ic}D_{ic}(\mathbf{N})} - \frac{\forall_{jc} D_{jc}(\mathbf{N})}{\sum_{i=1}^{M} V_{ic}D_{ic}(\mathbf{N})} + \sum_{d=1,d\neq c}^{C} Q_{jd}(\mathbf{N}-1_c)\right] \\ &= \left[1 + \frac{N_c - 1}{N_c} Q_{jc}(\mathbf{N}) + \sum_{d=1,d\neq c}^{C} Q_{jd}(\mathbf{N}-1_c)\right] \end{aligned} \tag{31}$$

This concludes our alternative derivation of the BS heuristics based on the core assumption that customer travel through any sequence of network stations is described by a semi-Markov process.

REFERENCES

Akyildiz I.F., Bolch G., 1988. Mean value analysis approximation for multiple server queueing networks. Performance Evaluation 8: 77-91.

Bard J., 1979. Some extensions to multi-class queueing network analysis. In: M. Arato, A. Butrimenko, E. Gelenbe, eds. Performance of Computer Systems, Amsterdam: North Holland, 51-62.

Baskett F., Chandy K.M., Muntz R.R., Palacios F., 1975. Open, closed and mixed networks of queues with different classes of customers. Journal of the ACM 22 (2): 248-260.

Buzacott J.A., Shanthikumar G.J., 1993. Stochastic models of manufacturing systems. Englewood Cliffs, NJ: Prentice-Hall, Inc.

Chandy K.M., Neuse D., 1982. Linearizer: a heuristic algorithm for queueing network models of computing systems. Communications of the ACM 25 (2): 126-134.

Cinlar E., 1975. Introduction to stochastic processes. Engl. Cliffs NJ: Prentice-Hall.

Conway A.E., de Souza e Silva E., Lavenberg S.S. 1989. Mean value analysis by chain of product form queueing networks. IEEE Transactions On Computers 38 (3): 432-442.

Fukunari M., Malmborg C. J., 2009. A network queuing approach for evaluation of performance measures in autonomous vehicle storage and retrieval systems. European Journal of Operational Research 193: 152–167.

Hopp W.J., Spearman M.L., 2000. Factory physics—Foundations of manufacturing management (2nd ed.). New York: Irwin/McGraw-Hill.

Jain H., Maheshwari S., Baghel K.P.S., 2008. Queueing network modelling of flexible manufacturing systems using mean value analysis. Applied Mathematical Modelling 32: 700–711.

Krzesinski A., Greyling J., 1984. Improved linearizer methods for queueing networks with queue dependent centres. ACM SIGMETRICS Performance Evaluation Review 12 (3): 41-51.

Manitz M., 2015. Analysis of assembly/disassembly queueing networks with blocking after service and general service times. Annals of Operations Research 226 (1): 417-441.

McGinnis L., Wu K., 2012. Performance evaluation for general queueing networks in manufacturing systems: characterizing the trade-off between queue time and utilization. European Journal of Operational Research 221: 328–339.

Ortega J.M., Rheinboldt W.C. 1970. Iterative solution of nonlinear equations in several variables, New York: Academic Press.

Neuse D., Chandy K.M., 1981. Scat: a heuristic algorithm for queueing network models of computing systems. ACM SIGMETRICS Performance Evaluation Review 10 (3): 59-79.

Pattipati K.R., Kostreva M.M., Teele J.L., 1990. Approximate mean value analysis algorithms for queueing networks: existence, uniqueness and

convergence results. Journal of the ACM 37 (3): 643-673.
Reiser M., Lavenberg S.S., 1980. Mean value analysis of closed multichain queueing networks. Journal of the ACM 27(2): 313-322.
Schweitzer P.J., 1979. Approximate analysis of multi-class closed networks of queues. Proceedings of the International Conference on Stochastic Control and Optimization, 25-29. April 5-6, Amsterdam.
Solot P., Bastos J.M., 1988. MULTIQ: a queueing model for FMSs with several pallet types. Journal of the Operational Research Society, 39(9):811-821.
Stecke K.E., Kim I., 1989. Performance evaluation for systems of pooled machines of unequal sizes: unbalancing versus balancing. European Journal of Operational Research 42: 22-38.
Suri R., Sahu S., Vernon M., 2007. Approximate mean value analysis for closed queuing networks with multiple-server stations. Proceedings of the 2007 Industrial Engineering Research Conference, 1-6. December 2-4, Nashville (Tennessee, USA)
Tempelmeier H., 2003. Practical considerations in the optimization of flow production systems. International Journal of Production Research, 41 (1): 149–170.
Wang H., Sevcick K.C., 2000. Experiments with improved approximate mean value analysis algorithms. Performance Evaluation 39: 189–206.
Wang H., Sevcik K.C., Serazzi G., Wang S., 2008. The general form linearizer algorithms: a new family of approximate mean value analysis algorithms. Performance Evaluation 65: 129–151.
Zahorjan J., Eager J.D., Sweillam H:M., 1988. Accuracy, speed and convergence of approximate mean value analysis. Performance Evaluation 8: 255-270.
Zhuang L., Hindi K.S., 1990. Mean value analysis for multiclass closed queueing network models of FMS with limited buffers. European Journal of Operational Research 46: 366-379.
Zijm W.H.M., Adan I.J.B.F., Buitenhek, R., Houtum, van, G.J., 2000. Capacity analysis of an automated kit transportation system. Annals of Operations Research 93 (1): 423-446.

AUTHORS BIOGRAPHY

PASQUALE LEGATO is an Associate Professor of Operations Research in the Department of Informatics, Modeling, Electronics and System Engineering (DIMES) at the University of Calabria, Rende (CS, Italy). He has been a member of the Executive Board of the University of Calabria as well as university delegate for the supervision of associations and spin-offs from the University of Calabria. He has been involved in several EEC funded research projects aimed at the technological transfer of SO procedures and frameworks in logistics. He is a member of the INFORMS Simulation Society. His research activities focus on queuing network models, stochastic simulation and the integration of simulation techniques with combinatorial optimization algorithms. His e-mail address is legato@dimes.unical.it and his web-page can be found at http://wwwinfo.dimes.unical.it/legato.

RINA MARY MAZZA is the Research Manager of the Department of Informatics, Modeling, Electronics and System Engineering (DIMES) at the University of Calabria, Rende (CS, Italy). She graduated in Management Engineering and received a PhD in Operations Research from the above university. She has been a consultant for operations modeling and simulation in container terminals. Her current research interests include discrete-event simulation and optimum-seeking by simulation in complex systems. Her e-mail address is rmazza@dimes.unical.it.