1997

An Aggregation Procedure for Simulating Manufacturing Flow Line Models

Paul Savory
University of Nebraska at Lincoln, psavory2@gmail.com

Gerald Mackulak
Arizona State University at the Tempe Campus

P. Savory and G.T. Mackulak (1997), "An Aggregation Procedure for Simulating Manufacturing Flow Line Models," ***Computers and Operations Research***, Volume 24, No. 11, pp. 1963-1073.

AN AGGREGATION PROCEDURE FOR SIMULATING MANUFACTURING FLOW LINE MODELS

SCOPE AND PURPOSE

In developing a discrete-event simulation model, most of the actual features of a system under study are ignored and an abstraction is developed. If done correctly, this idealization provides a useful approximation of the real system. Aggregation is one of the available techniques for abstracting a system. Potential benefits for developing an aggregate simulation model include a reduced run length, a less complex model, and decreased demand for simulation resources. This paper presents a quantitative method for creating an aggregate discrete-event simulation model of a flow line manufacturing system. Computational experiments indicate that the average part cycle time through a flow line can be approximated with minimal error.

ABSTRACT

We develop a formal method for specifying an aggregate discrete-event simulation model of a production flow line manufacturing system. The methodology operates by aggregating production stations or resources of a flow line. Determining the specifications for representing the aggregated resources in a simulation model is the focus of our presentation. We test the methodology for a set of flow lines with exponentially distributed arrival and service times. Comparisons between analytical and simulation results indicate the aggregation approach is quite accurate for estimating average part cycle time.

P. Savory and G.T. Mackulak (1997), "An Aggregation Procedure for Simulating Manufacturing Flow Line Models," ***Computers and Operations Research***, Volume 24, No. 11, pp. 1963-1073.

AN AGGREGATION PROCEDURE FOR SIMULATING MANUFACTURING FLOW LINE MODELS

Paul A. Savory[1]† and Gerald T. Mackulak[2]‡

[1]Industrial and Management Systems Engineering, University of Nebraska, Lincoln, NE 68588-0518, USA
[2]Industrial and Management Systems Engineering, Arizona State University, Tempe, AZ 85287-5906, USA

† PAUL A. SAVORY is an Assistant professor in Industrial and Management Systems Engineering at the University of Nebraska - Lincoln (UNL). Prior to joining the faculty of UNL, he earned a Ph.D. in Industrial Engineering from Arizona State University, a Master's degree in Operations Research and a Bachelor's degree in Computer Science, both from Oregon State University. In addition to having taught university and industrial courses in simulation, operations research, and applied statistics, he has held positions as a software engineer and a quality control inspector.

‡ GERALD T. MACKULAK is an Associate Professor in the Department of Industrial and Management Systems Engineering at Arizona State University. He is co-Director of the Systems Simulation Laboratory. Dr. Mackulak received all his degrees from Purdue University in the area of Industrial Engineering. Before joining the ASU faculty in 1980, he held positions with U.S. Steel, Burroughs Corporation, and Pritsker and Associates. Dr. Mackulak has published extensively in the areas of CIM integration methods, simulation methodology, and production control. He has taught both public and private seminars on CIM system integration, simulation, and economic justification of CIM implementations.

1 INTRODUCTION

Abstraction is a technique for reducing a system description to a level of detail that can be more easily managed. All discrete-event simulation models contain some level of abstraction [1]. One of the primary abstraction techniques, aggregation, involves lumping details into a single, approximately equivalent function.

The majority of aggregation research for simulation centers on developing conceptual frameworks for allowing modeling objects to be combined and/or extended [2, 3, 4, 5, 6]. Limited research has investigated aggregating a system that is modeled using a discrete-event world-view. Rogers *et al.* [7] develops a general framework for aggregation and disaggregation and explores its potential for solving linear programming and network flow optimization problems. Friedman [8] presents a reduction procedure based on the dominance of a queue's impact on the other queues of a flow line. Gershwin [9] presents a decomposition method for evaluating performance measures of tandem queueing systems with finite buffers in which blocking and starvation are important. This algorithm was later expanded [10] to consider the case of unreliable tandem queueing systems. A similar procedure works by comparing adjacent nodes [11]. Schweitzer and Altiok [12] developed an aggregation procedure for modeling tandem queues that lack intermediate buffers. Additional research results for the decomposition of tandem queueing model include Hunt [13], Hillier and Boling [14], Altiok [15], and Gun and Makowski [16]. The majority of these efforts focus on the mathematical approximation of exponential queueing systems with finite buffers.

The objective of this paper is to develop a formal method for specifying an aggregate discrete-event simulation model of a production flow line manufacturing system. The methodology operates by combining production stations or resources of a flow line. Determining

the specifications for representing the *aggregation resources* in an aggregate simulation model is the focus of our presentation. The success of the aggregation procedure is judged by how well it estimates the average cycle time (*i.e.*, sojourn time) of a part to wait and be serviced by all resources of the flow line. This performance variable is important for planning delivery dates [17], and is also useful for reducing costs [18]. This paper discusses five aspects of the aggregation methodology. Section 2 defines the set up, which includes the flow line system, the model assumptions, and the notation. Section 3 discusses the steps of the aggregation methodology. Section 4 demonstrates the aggregation process for a simple two resource flow line system. Section 5 summarizes testing the methodology for a set of flow lines with exponentially distributed arrival and service time distributions. Section 6 provides a summary of the research results.

2 The Setup

2.1 Definition and Assumptions for a Manufacturing Flow Line

The manufacturing system that this analysis explores is a production flow line (or flow shop) system [19]. Systems of this type are widely used in industry to represent situations in which parts arrive at a service area, obtain the service they require, and then move on to the next service area or leave the system. Flow line manufacturing systems have been widely studied in operations research as serial, series or tandem queueing systems [see references 20-36].

The flow line definition used for this analysis is: *The single part type is processed at N production stations (resources) with the ordering of processing at a production station (resource) being the same for all parts* (based on Pinedo [30]). A flow line is a sequence of N production steps or resources (R_i), consisting of a machine and associated buffer (or queue) area. This relationship is illustrated in Figure 1. A description of the notation and definitions are given

in Table 1. The assumptions associated with the flow line production system are summarized in Table 2 (based on Hendricks [26]).

<<< Figure 1 Approximately Here >>>

<<< Table 1 Approximately Here >>>

<<< Table 2 Approximately Here >>>

Table 3 describes the parameters of a flow line system based on the flow line description and the listing of assumptions. A flow line (FL) consists of three primary components, the receiving area (R), the shipping area (S), and N production resources (R_i). The receiving area (R) is described by the mean time between arrivals ($1/\lambda$), where λ is the arrival rate and Z is the maximum number of parts that can arrive from the storage area. The shipping area (S) is characterized by its storage capacity (U). From the assumptions of Table 2, the mean time between arrivals follows an exponential distribution, and Z and U are assumed to be infinite.

<<< Table 3 Approximately Here >>>

Each production step or resource (R_i) is composed of a queue (Q_{i-1}) and a machine (M_i) which is to service (*i.e.*, process or machine) a part. The queue component of a resource represents the waiting space preceding the machine at which a part waits until a server becomes available to process it. It is characterized by its buffer capacity (x_{i-1}) which is assumed to be infinite (*i.e.*, $x_{i-1} = \infty$). The service time to process a single part on each machine (M_i) is specified by a probability distribution (f_i) and its corresponding mean service time (m_i). A machine is also characterized by the number of parallel, identical servers ($s_i \geq 1$) that perform the machine's task.

2.2 An Aggregate Flow Line Representation

This analysis proposes that in an aggregation representation of a flow line, all resources with a given server capacity are aggregated together to form an *aggregate resource*, AR_i, where i represents the aggregate resource service capacity. For example, AR_1, represents all single-server resources from the original system and AR_2 represents all two-server resources. Obviously, many other characteristics of a flow line system can be used as a basis for aggregation. The decision to aggregate on the service capacity of a resource is based on the fact that service capacity is the one characteristic that remains constant for any type of flow line system.

Table 4 summarizes the parameters for the aggregate flow line. An aggregated flow line consists of the receiving area (R), the shipping area (S), and a collection of G aggregation resources (AR_i), where G is the maximum number of parallel, identical servers used by any machine in the original flow line. Each aggregation resource represents the aggregation of the P_i resources with a given server capacity. Those aggregation resources that exist (*i.e.*, not the empty set), are characterized by a queue and are associated with a machine with server capacity i.

<<< Table 4 Approximately Here >>>

The queue (Q_i^*) component of an aggregation resource is defined by its storage capacity. As with the original system, the buffer capacity is assumed to be infinite (*i.e.*, $x_i = \infty$). The machine, M_i^*, represents all the machines of the original system with capacity i. That is,

$$M_i^* = \left\{ M_j : s_j = i \right\} \begin{matrix} i = 1, \ldots, G \\ j = 1, \ldots, N \end{matrix}$$

The machine of an aggregation resource is characterized by its service time distribution (f_i^*) and its corresponding service mean (δ_i^*). Once this service mean is estimated, a process for

generating variables from the service time distribution will be possible. The procedure for accomplishing these tasks is presented in the next section.

A description of how aggregation resources will be modeled in an aggregate simulation model is given in Figure 2. When a part arrives to the aggregate flow line it is sent to each of the aggregation resources. Note that the order of resources is removed and the arrival process of a part to each of the aggregate resource is the same Poisson process. This representation is based on the work of Burke [38] who showed that the output process of a M/M/S queue is itself Poisson, with a mean equal to its arrival mean. Because the original resources have an infinite buffer capacity, one resource does not impact another, and thus it can be surmised, in estimating the cycle time of a part, that the resources act independently of one another. As a result, each aggregation resource is also independent and goes through the same arrival process in the aggregate flow line representation.

<<< Figure 2 Approximately Here >>>

3 AGGREGATION METHODOLOGY

3.1 *Step 1:* Computing Cycle Time

The first task in estimating the aggregate resource parameters is to compute the expected cycle time of all N resources in the flow line. That is, calculate the total steady-state processing/service and waiting time that a part will experience at each of the resources.

Because the arrival process to a flow line is Poisson and all resources have exponential service times, determining a resource's cycle time requires the use of the M/M/S queuing formula. This formula, written in terms of the flow line terminology, is:

$$E\left[T_j\right] = \frac{\lambda\rho_j\left(\lambda/\mu_j\right)^{s_j} P_{o_j}}{s_j!\left(1-\rho_j\right)^2} + m_j \qquad j = 1,\ldots,N$$

where: $E[T_j]$ Expected cycle time of resource j (j = 1,...,N)

λ Arrival rate of parts to the flow line

s_j Number of parallel, identical servers for resource j (j = 1,...,N)

m_j Mean service time of resource j (j = 1,...,N)

ρ_j Traffic intensity of resource j (j = 1,...,N): $\rho_j = \frac{\lambda m_j}{s_j}$

P_{0_j} The probability that zero services are busy for resource j (j = 1,...N):

$$P_{0_j} = \frac{1}{\left[\sum_{n=0}^{s_j-1}\frac{(\lambda m_j)^n}{n!} + \frac{(\lambda m_j)^{s_j}}{s_j!(1-\lambda m_j/s_j)}\right]}$$

Once cycle time estimates have been computed for all N resources, the next task is to determine the average cycle time of the G aggregation resources in the aggregate flow line system. This is the sum of all resource cycle times represented by the aggregation resource divided by the number of resources aggregated (P_i). This is computed by first summing the cycle time (T_j) of the resources aggregated by AR_i, giving the total aggregate cycle time (T_i^*) represented by an aggregation resource. That is,

$$E\left[T_i^*\right] = \sum_{R_j \in AR_i} T_j \qquad \begin{matrix} i = 1,\ldots,G \\ j = 1,\ldots,N \end{matrix}$$

Next, the average cycle time for each aggregation resource is the total aggregate cycle time divided by the number of resources combined at the aggregation resource. That is,

$$E\left[\overline{T}_i^*\right] = \frac{T_i^*}{P_i} \qquad i = 1,\ldots,G$$

3.2 *Step 2:* Service Mean of an Aggregation Resource

The next step is to use the average cycle time of each aggregation resource to estimate its corresponding service mean. Each estimated service mean will later be used by the aggregation methodology to weight the original resource service time distributions of its corresponding aggregation resource.

The procedure for solving for the mean service time of an aggregation resource involves applying queueing formulas backwards. Most uses of queueing formula involve specifying the parameters (*i.e.*, arrival rate, service mean, and capacity) of a resource or queueing system and computing the cycle or waiting time (such as was done in the previous section). This analysis differs in that it seeks to specify the arrival rate, capacity, and cycle time of an aggregation resource with the objective of computing the mean service time. Hence, given the cycle time, it is solving for the mean service time.

Estimating the service mean (δ_i^*) for an aggregation resource with an estimated average cycle time ($\overline{T_i^*}$) requires solving the following M/M/S queueing formula for δ_i^*:

$$E\left[\overline{T_i}\right] = \frac{\lambda\rho_i\left(\lambda\delta_i^*\right)^i P_{0_i}^*}{i!\left(1-\rho_i^*\right)^2} + \delta_i^* \quad i = 1,...,G \qquad (*)$$

where:

$E\left[\overline{T_i^*}\right]$ Expected average cycle time of aggregate resource i (i = 1,...,G)

λ Arrival rate of parts to the flow line

δ_i^* Mean service time of aggregate resource i (i = 1,...,G)

ρ_i^* Traffic intensity of aggregation resource i (i = 1,...,G): $\rho_i^* = \frac{\lambda\delta_i^*}{i}$

$P_{o_i}^*$ $$P_{o_i}^* = \frac{1}{\left[\sum_{n=0}^{i-1}\frac{(\lambda\delta_i^*)^n}{n!} + \frac{(\lambda\delta_i^*)^i}{i!(1-\lambda\delta_i^*/i)}\right]} \quad i = 1,...,G$$

The previous section used the M/M/S formula for computing the average cycle time of each aggregation resource. This step of the methodology specifies the expected or average cycle time for an aggregation resource, the arrival rate, and the number of servers aggregated by the aggregation resources and solves for the average service mean (δ_i^*). This is one of the two parameters we are attempting to determine for each of the aggregation resources. The other is the aggregate resource service time distributions. The next section provides the basis for estimating this distribution by using the average service mean of an aggregation resource to determine a weighting relationship between the resources that are represented by each aggregation resource.

3.3 *Step 3:* Resource Weighting Procedure

The weights developed in this section represent the percentage contribution of each resource service mean towards an aggregation resource's service mean. The weights must satisfy two conditions: (1) the sum of all the resource weights multiplied by the original resource mean service times is equal to the average service time of the aggregation resource (δ_i^*) and (2) the sum of the weights is equal to one. More formally, these two conditions are:

$$(1)\ \sum_{R_j \in AR_i} w_j^* m_j = \delta_i^* \quad \begin{matrix} i = 1,\ldots,G \\ j = 1,\ldots,N \end{matrix} \quad \text{and} \quad (2)\ \sum_{R_j \in AR_i} w_j^* = 1 \quad \begin{matrix} i = 1,\ldots,G \\ j = 1,\ldots,N \end{matrix}$$

This convex relationship determines the proportional weight that each resource service mean contributes towards the average service time of the aggregation resource.

Distribution weights can most easily be determined for the case in which an aggregate resource represents a single resource. In such a case, the aggregate resource service mean (δ_i^*) is merely the resource service mean, m_j, where $R_j \in AR_i$. Thus, the distribution weight, w_j^*, for the

resource service mean of resource R_j is 1.0. Clearly this satisfies the two weighting conditions: (1) $w_j^* \times m_j = \delta_i^*$ and (2) $w_j^* = 1.0$.

Determining the distribution weights for when two resources are aggregated together is similar. For example, suppose aggregation resource (*e.g.*, AR_3) represents the aggregation of two three-server resources (*e.g.*, R_2 and R_5). Solving for distribution weights involves deciding how to weight the two individual service resource means (m_2 and m_5) such that they equal the aggregate service mean (δ_2^*). Applying the two weighting conditions results in the following equations:

$$(w_2^* \times m_2) + (w_5^* \times m_5) = \delta_3^*$$
$$w_2^* + w_5^* = 1$$

Since the values of m_2, m_5, and δ_3^* are known, the task of solving for w_2^* and w_5^* involves applying standard algebraic procedures for solving two equations with two unknowns.

By similar logic, consider aggregation resource two (AR_2) that represents the aggregation of (say) five resources (*e.g.*, R_1, R_3, R_4, R_6, and R_7). This aggregation resource is depicted in part (a) of Figure 3. The solution technique for determining the distribution weights requires solving:

$$(w_1^* \times m_1) + (w_3^* \times m_3) + (w_4^* \times m_4) + (w_6^* \times m_6) + (w_7^* \times m_7) = \delta_2^*$$
$$w_1^* + w_3^* + w_4^* + w_6^* + w_7^* = 1$$

In this instance, the solution can only be reduced to a set of relationships among the variables. To determine the service time weighting requires the techniques of the previous sections (determining total cycle time and deriving the average aggregate resource service mean) with a recursive algorithm to reduce (by aggregating) the resources of an aggregation resource to only two resources.

A detailed description of the algorithm is presented in Savory [37]. In essence, the algorithm incrementally aggregates within the aggregation resource to reduce the number of

resources represented by the aggregation resource to only two, in much the same manner that the original system of N resources were combined. This is demonstrated in parts (b) and (c) of Figure 3. Once only two resources are presents, part (d) of Figure 3, determining the distribution weights is derived by solving a set of two equations with two unknowns.

<<< Figure 3 Approximately Here >>>

The reason this approach has been termed recursive is that once distribution weights can be found for two resources (one of which is an aggregation resource), the procedure works incrementally backwards using the current solution to solve the prior levels of aggregation. Thus, since a value for w_7^* is known from solving the equations describing Figure 3(d), the equations representing Figure 3(c) reduce to a set of two equations with two unknowns. This backward solution process continues until all original resources represented by the aggregation resource have distribution weights. The weights determined by the algorithm will be used in the next section to specify a procedure for estimating the service time distributions of the aggregation resources (f_i^*).

3.4 *Step 4:* Specifying the Aggregate Simulation Model

Consider any aggregate resource AR_i. This resource is characterized by f_i^*, the probability density function for the aggregation resource service time distribution, which represents all of the service time distributions (f_i) that have been aggregated to form AR_i. The objective of this and the prior three steps has been to somehow represent f_i^*. One approach is to mathematically determine this distribution; but in practice, this is a difficult, if not impossible,

task. An alternative is to represent this unspecified service time distribution not as a mathematical function, but rather, as a relationship from which random numbers can be sampled.

Recall that f_i^* represents all the service time distributions of an aggregation resource. As such, we know the individual service time distributions (f_i) composing f_i^* (whose mathematical representation is unspecified), plus the weights (w_i^*) specifying the importance (i.e., contribution) of each resource service time mean (m_i) toward an aggregation resource. Using this information, an effective solution is to use a procedure known as the *composition* or *mixture method* for generating random variables [39].

Kronmal and Peterson [40] explain that some continuous distribution are efficiently generated by representing them as mixtures of several other (continuous) distributions that are easy to generate. In the case of a production flow line, the aggregate service distribution (f_i^*) must be estimated. It is a very difficult task, but it can be done by using the weights and the original service time distribution (f_i) that make up the individual aggregation resources. That is,

$$f_j^*(x) = \sum_{k \in R_i} w_k^* f_k(x) \qquad \begin{matrix} i = 1, \ldots N \\ j = 1, \ldots, G \end{matrix}$$

Thus, the aggregate resource service time distribution is never specified, but rather, values from it will be sampled during the execution of the aggregate simulation model. By repeated sampling, each of the component (*i.e.*, original service time) distributions is selected in accordance with their weights and, hence, the samples are generated in accordance with the (unspecified) aggregate service time distribution [41].

With a procedure to model aggregation resources, the final need is to collect part cycle time estimates from the simulation model. Recall that, in the aggregate simulation model, each of the aggregation resources is explicitly modeled to represent the *average* of all the resources it

has aggregated. Figure 2 described this system. To determine the average cycle time for a part, assume that a total of r parts "flow" through the model when the aggregate simulation model is run. Table 5 summarizes the four statistics that must be collected or computed.

<<< Table 5 Approximately Here >>>

Since an aggregate resource is an average of all the original resources (R_i) it aggregates, the cycle time of a part through an aggregation resource (AR_i) is really an average of the true aggregate resource processing time. The true processing time of an aggregate resource is $Y_{ij} = P_i \times \overline{Y_{ij}}$, where P_i is the number of resources aggregated in AR_i. Correspondingly, the total cycle time of part number j (j = 1,...,r) is equal to the sum of the processing and waiting time that part number j spends at all the G aggregate resources. Hence,

$$Z_j = \sum_{i=1}^{O} Y_{ij} \qquad j = 1,\ldots,r$$

where Z_j is the total cycle time of a part. The final statistic to be collected or computed is the average cycle time of all r parts which "flow" through the simulation model. That is,

$$\overline{Z} = \frac{\sum_{i=1}^{r} Z_j}{r}$$

Applying the four steps to a manufacturing flow line results in the weighting relationship between the service time distributions for each of the aggregation resources.

4 NUMERICAL EXAMPLE

To demonstrate the aggregation process, consider three M/M/1 queues in series. Assume that arrivals are Poisson and occur at a rate of λ=.5. The service rate for each the resources is .75, .6, and .7, respectively. For this simple example, each of the single server resources is

aggregated together to form AR_1. The first step of the methodology estimates the cycle time for the resources:

$$E[T_1] = \frac{1}{.75-.5} = 4,\ E[T_2] = \frac{1}{.6-.5} = 10, \text{ and } E[T_3] = \frac{1}{.7-.5} = 5$$

Thus, the total aggregate cycle time to be represented by AR_1 is:

$$E[T_1^*] = E[T_1] + E[T_2] + E[T_3] = 4 + 10 + 5 = 19$$

Since AR_1 will represents the average aggregation of the two resources, the average cycle time is computed as:

$$E[\overline{T}_1^*] = \frac{E[T_1^*]}{3} = \frac{19}{3} = 6.3333$$

The next step of the methodology estimates the mean service time required of an aggregation resource to have average cycle time, $\overline{T}_1^*$. Since this is a single server system, equation (*) of Section 3.2 results in:

$$\delta_1^* = \frac{\overline{T}_1^*}{1 + \lambda \overline{T}_1^*} = \frac{6.3333}{1 + (.5)(6.3333)} = 1.52$$

Next, the aggregate resource's service time mean is used to determine a weighting relationship between the original resource service time means. Since three resources are aggregated, the following weighting relationship needs to solved:

$$\begin{gathered}(w_1^* \times m_1) + (w_2^* \times m_2) + (w_3^* \times m_3) = \delta_1^* \\ w_1^* + w_2^* + w_3^* = 1\end{gathered} \qquad (1)$$

To find the distribution weights, the recursive algorithms must be applied. The algorithm first aggregates two resources (say R_1 and R_3). This is done by summing the total cycle time of the two resources (T_1 and T_3) and dividing this by two to find the average cycle time of the "new" aggregate resource. That is, the average cycle time of aggregate resource $A_{1|3}$ (an aggregate resource within an aggregation resource) is $\overline{T}_{1|3}^*$:

$$\overline{T}^*_{1|3} = \frac{T_1 + T_3}{2} = \frac{4+5}{2} = 4.5$$

Next, the mean service time ($\delta^*_{1|3}$) for $A_{1|3}$ is calculated:

$$\delta^*_1 = \frac{\overline{T}^*_1}{1+\lambda\overline{T}^*_1} = \frac{4.5}{1+(.5)(4.5)} = 1.38462$$

Aggregating R_1 and R_3 within AR_1 results in equations (1) reducing to:

$$(w^*_{1|3} \times \delta^*_{1|3}) + (w^*_2 \times m_2) = \delta^*_1 \qquad (2)$$
$$w^*_{1|3} + w^*_2 = 1$$

Since the value for m_2 is known $\left(\frac{1}{.6} = 1.66667\right)$ and the values of $\delta^*_{1|3}$ and δ^*_1 have been computed, the above equations can easily be solved. Solving for the weights results in $w^*_{1|3}$ = .520014 and w^*_2 = .479986. By knowing the value of w^*_2, equations (1) can be solved and yield the following results: w^*_1 = .239987 and w^*_3 = .280027.

The final step of the aggregation methodology is to specify the aggregate simulation model using composite sampling and to appropriately collect all the necessary statistics. Figure 4 displays a subset of the SLAM II simulation model for this situation.

<<< Figure 4 Approximately Here >>>

5 COMPUTATIONAL EXPERIMENTS

To test the effectiveness of applying the aggregation methodology to a flow line with all exponential service times, ten random flow line scenarios were generated by a software program [37]. Table 6 summarizes each of these test scenarios. For example, Scenario 1 is a flow line consisting of 16 resources with the first resource in the flow line having seven servers and the second resource having six servers. The average utilization of the sixteen resources (or queueing systems) is 40.28%. These sixteen resources are combined into seven aggregation resources

(AR_1, AR_2, AR_4, AR_5, AR_6, AR_7, AR_8), with each aggregation resource representing the combination of all similar capacity resources. For instance, the two seven server resources will be grouped together. A complete description of the distribution parameters and the simulation results from the aggregate simulation models can be found in Savory [37].

An aggregate simulation model was written in the SLAM II simulation language [39] for each of the test scenarios. Thirty replications of each of the aggregate simulation models were run under steady-state conditions. Table 6 compares the simulation cycle time estimates to the analytical steady-state results (which can be computed for an exponential system). The results indicate that the average relative error,

$$RE = 100\% \times \left[\frac{\left| \text{average aggregate cycle time - steady state estimate} \right|}{\text{steady state estimate}} \right],$$

associated with this diverse set of test cases is only 1.1390%. A 95% confidence interval computed on the average relative error of the cycle time is (.5955%, 1.6825%). Overall, it appears that the aggregation methodology is accurate in estimating the cycle time.

<<< Table 6 Approximately Here >>>

6 FINAL COMMENTS

The aggregation methodology presented in this paper is a formal analytical technique for creating an aggregate discrete-event simulation model. The objective of the aggregation methodology is to generate the specifications necessary for creating an aggregate simulation model for approximating the average cycle time of a part through a flow line. The aggregation procedure works by defining the flow line system, computing the weighting time for the flow line and aggregating the resources of the flow time based on the number of parallel servers into aggregation resources. Next, it estimates the aggregation service mean, estimates the service

mean of each aggregate resource, and weights the original service time means. The final step specifies the aggregate simulation model using composite random number sampling. The testing of the aggregation methodology on a diverse group of flow lines with Poisson arrivals and exponentially distributed service times shows that the methodology works quite well for estimating part cycle time.

REFERENCES

1. Pegden, C. D., R. E. Shannon, and R. P. Sadowski, 1995. *Introduction to Simulation Using SIMAN*. New York: McGraw-Hill, Inc.

2. Zeigler, B., 1986. "Hierarchical Modular Modeling/Knowledge Representation." *SCS 1986 Winter Simulation Conference*, pp. 129-137.

3. Zeigler, B. P., 1987. "Hierarchical, Modular Discrete-Event Modelling in an Object-Oriented Environment." *Simulation*. 49: 219-230.

4. Portier, F. J., 1987. "Implementing the Product Automaton Formalism." *SCS 1987 Winter Simulation Conference*, pp. 544-553.

5. Overstreet, C. M. and R. E. Nance, 1985. "A Specification Language to Assist in Analysis of Discrete Event Simulation Models." *Communications of the ACM*. 28: 190-201.

6. Derrick, E. J., O. Balci, and R. Nance, 1989. "A Comparison of Selected Conceptual Frameworks for Simulation Modeling." *SCS 1989 Winter Simulation Conference*, pp. 711-718.

7. Rogers, D. F., R. D. Plante, R. T. Wong, and J. R. Evans, 1991. "Aggregation and Disaggregation Techniques and Methodology in Optimization." *Operations Research*. 39: 553-582.

8. Friedman, H. D., 1965. "Reduction Methods for Tandem Queuing Systems." *Operations Research* 13: 121-131.

9. Gershwin, S. B., 1987. "An Efficient Decomposition Method for the Approximate Evaluation of Tandem Queues with Finite Storage Space and Blocking." *Operations Research*, 35: 291-305.

10. Gershwin, S. B. 1989. "An Efficient Decomposition Algorithm for Unreliable Tandem Queueing Systems with Finite Buffers." H. G. Perros and T. Altiok, ed., *Queueing Networks with Blocking.* North-Holland, pp. 127-146.

11. Takahashi, Y. 1989. "Aggregate Approximation for Acyclic Queueing Networks with Communication Blocking." H. G. Perros and T. Altiok, ed., *Queueing Networks with Blocking.* North-Holland, pp. 33-46.

12. Schweitzer, P. J. and T. Altiok 1989. "Aggregate Modelling of Tandem Queues Without Intermediate Buffers." H. G. Perros and T. Altiok, ed., *Queueing Networks with Blocking.* North-Holland, pp. 47-72.

13. Hunt, G. C., 1956. "Sequential Arrays of Waiting Lines." *Operations Research*. 4: 674-683.

14. Hillier, F. S. and R. W. Boling, 1967. "Finite Queues in Series with Exponential or Erlang Service Times - A Numerical Approach." *Operations Research*. 15: 286-303.

15. Altiok, T., 1982. "Approximate Analysis of Exponential Tandem Queues and Blocking." *European Journal of Operations Research*. 11: 390-398.

16. Gun, L. and A. M. Makowski 1989. "An Approximation Method for General Tandem Queueing Systems Subject to Blocking." H. G. Perros and T. Altiok, ed., *Queueing Networks with Blocking.* North-Holland, pp. 147-174.

17. Cheng, T. C. E., 1990. "Analysis of Material Flow in a Job Shop with Assembly Operations." *International Journal of Production Research*. 28: 1369-1383.

18. Mott, J. and K. Tumay, 1992. "Developing a Strategy for Justifying Simulation." *Industrial Engineering*. 24: 38-42.

19. Aneke, N. A. G. and A. S. Carrie, 1984. "A Comprehensive Flowline Classification Scheme." *International Journal of Production Research*, 22: 281-297.

20. Altiok, T., 1989. "Approximate Analysis of Queues in Series with Phase-Type Service Time and Blocking." *Operations Research*. 37: 601-610.

21. Brandwajin, A. and Y. L. Jow, 1988. "An Approximation Method for Tandem Queues With Blocking." *Operations Research*. 36: 73-83.

22. Graves, S. C., 1986. "A Tactical Planning Model for a Job Shop." *Operations Research*. 34: 522-533.

23. Hillier, F. S. and R. W. Boling, 1966. "The Effect of Some Design Factors on the Efficiency of Production Lines with Variable Operation Times." *Journal of Industrial Engineering*. 17: 651-658.

24. Konig, D. and V. Schmidt, 1984. "Relationship Between Time/Customer Stationary Characteristics of Tandem Queues Attended by a Single Server." *Journal of the Operations Research Society of Japan*. 27: 191-204.

25. Ku, P. S. and S. C. Niu, 1986. "On Johnson's Two-Machine Flow Shop with Random Processing Times." *Operations Research*. 34: 130-36.

26. Hendricks, K. B., 1992. "The Output Processes of Serial Production Lines of Exponential Machines With Finite Queues." *Operations Research*. 40: 1139-1147.

27. Lee, Y. J. and P. Zipkin, 1992. "Tandem Queues With Planned Inventories." *Operations Research*. 40: 936-947.

28. Maaloe, E., 1973. "Approximations Formulae for Estimation of Waiting-Time in Multiple-Channel Queueing System." *Management Science*. 19: 703-710.

29. McCormick, S. T., M. L. Pinedo, S. Shenker, and B. Wolf, 1989. "Sequencing in an Assembly Line with Blocking to Minimize Cycle Time." *Operations Research*. 37: 925-935.

30. Pinedo, M., 1982. "Minimizing the Expected Makespan in Stochastic Flow Shops." *Operations Research*. 30: 148-162.

31. Shalmon, M. and M. A. Kaplan, 1984. "A Tandem Network of Queues with Deterministic Service and Intermediate Arrivals." *Operations Research*. 32: 753-773.

32. Suresh, S. and W. Whitt, 1990. "Arranging Queues in Series: A Simulation Experiment." *Management Science*. 36: 1080-1091.

33. Wittrock, R. J., 1988. "An Adaptive Scheduling Algorithm For Flexible Flow Lines." *Operations Research*. 36: 445-453.

34. Whitt, W., 1983. "Comparison Conjectures about the M/G/S Queue." *Operations Research Letters*. 2: 203-209.

35. Whitt, W., 1984. "Departures from a Queue with Many Busy Servers." *Mathematics of Operations Research*. 9: 534-544.

36. Wolff, R. W., 1982. "Tandem Queues with Dependent Service Times in Light Traffic." *Operations Research*. 30: 619-635.

37. Savory, P. A., 1993. "A Robust Aggregation Approach to Simplification of Manufacturing Flow Line Models." Ph.D. Dissertation, Arizona State University.

38. Burke, P. J., 1956. "The Output of a Queueing System." *Operations Research*. 4: 699-704.

39. Pritsker, A. A. B., 1986. *Introduction to Simulation and Slam II*. West Lafayette, Indiana: Systems Publishing Corporation.

40. Kronmal, R. A. and A. V. Peterson, 1979. "On the Alias Method for Generating Random Variables From a Discrete Distribution." *The American Statistician*. 33: 214-218.

41. Law, A. M. and W. D. Kelton, 1991. *Simulation Modeling and Analysis*. Second Edition. New York: McGraw Hill.

Table 1: Description of flow line system.

R	Receiving area
S	Shipping area
N	Number of production steps to produce a part
R_i	A resource or production step consisting of a queue and associated machine (i = 1 to N)
M_i	Machine i (i = 1 to N)
Q_j	Queue or buffer proceeding M_{j+1} (j = 0 to N-1).

Table 2: Basic assumptions of the manufacturing flow line (based on Hendricks [26]).

1. The production line (flow line) is a series arrangement of a finite number of N resources. The machine component of a resource has s_i parallel servers and each server can operate on one part at a time and has internal storage for that part.
2. All parts are processed by each resource in the flow line.
3. The production line is operating under steady-state conditions.
4. Parts leave the receiving area (arrive to Q_0) following an exponential distribution with density function $f(t) = \lambda e^{(-\lambda t)}$.
5. The machines M_i (i = 1,...,N) have mutually independent processing times which are exponentially distributed.
6. The shipping area has unlimited storage capacity, and the receiving area has an unlimited supply of parts.
7. Parts are selected from all queues following a first-in-first-out priority scheme.
8. All machines are reliable and produce no bad (or scrap) parts.
9. No batching and no setup times are allowed.
10. All queues between machines have infinite storage capacity.
11. The flow line does not allow for feedback or rework.

Table 3: Parameters of a flow line system (from Savory [37]).

$$FL = \langle R, R_1, ..., R_N, S \rangle$$
$$R = \langle 1/\lambda, Z \rangle$$
$$S = \langle U \rangle$$
$$R_i = \langle Q_{i-1}, M_i \rangle \quad i = 1, ..., N$$
$$Q_{i-1} = \langle x_{i-1} \rangle \quad i = 1, ..., N$$
$$M_i = \langle f_i, m_i, s_i \rangle \quad i = 1, ..., N$$

Table 4: Parameters of an aggregate flow line.

$$AFL = \langle R, AR_1, ..., AR_G, S \rangle$$
$$AR_i = \langle \varnothing, \langle Q_i^*, M_i^* \rangle \rangle \quad i = 1, ..., G$$
$$Q_i^* = \langle x_i^* \rangle \quad i = 1, ..., G$$
$$M_i^* = \langle F_i^*, \delta_i^* \rangle \quad i = 1, ..., G$$

Table 5: Statistics that need to be collected or computed.

$\overline{Y_{ij}}$ = Average cycle time of part number j at AR_i $\quad i = 1, ..., G$, $j = 1$ *to* r

Y_{ij} = True cycle time of part number j at AR_i $\quad i = 1, ..., G$, $j = 1$ *to* r

Z_j = Total cycle time of part number j $\quad j = 1$ *to* r

$\overline{Z}$ = Average cycle time for all r parts

Table 6: Relative error from comparing the aggregate simulation model estimate of cycle time to the analytical, steady-state estimate.

Scenario	Number of Resource	Number of Servers for each Resource (corresponding to the flow line sequence)	Average Utilization	Relative Error
1	16	7,6,6,5,1,4,5,5,6,7,4,2,8,5,1,1	40.28%	.8763%
2	11	4,3,3,8,3,5,2,2,5,2	49.98%	2.0448%
3	20	3,8,8,3,1,5,4,4,4,2,8,6,5,2,2,4,1,6,4,7	51.55%	1.3367%
4	6	7,8,5,8,4,5	34.13%	.0093%
5	18	8,6,5,3,1,1,1,8,3,4,7,4,6,6,3,3,4 2	55.83%	.9189%
6	11	1,6,1,8,2,8,7,6,8,5,4	40.53%	.6956%
7	21	7,6,4,3,8,3,5,8,1,6,2,5,2,6,4,4,3,2,1,7,4	51.63%	1.6219%
8	20	7,3,7,3,7,2,8,6,4,7,3,3,7,6,8,6,5,4,7,1	47.60%	1.3367%
9	8	4,5,6,8,1,3,7,1	49.97%	.9282%
10	19	2,1,7,7,4,1,7,3,3,8,3,7,4,5,1,8,3,4,5	51.58%	1.6219%

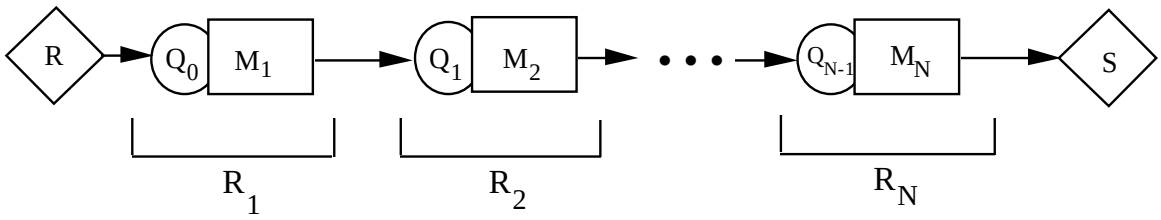

Figure 1: A flow line has N production steps, where each resource or production step consists of a machine and associated waiting or buffer area.

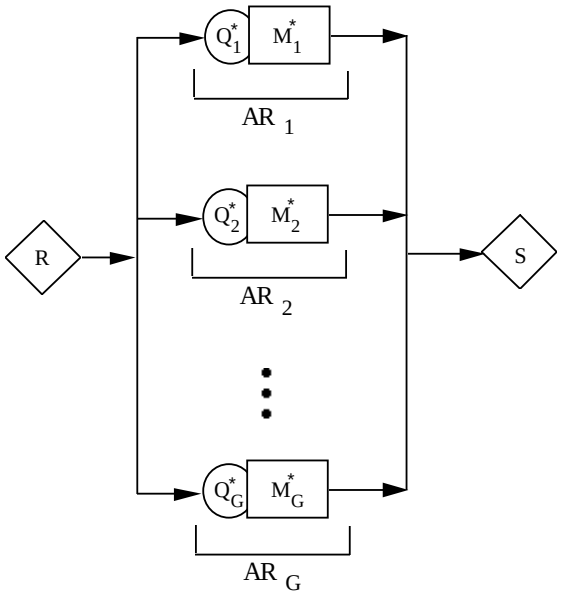

Figure 2: Representation of an aggregate flow line to estimate cycle time.

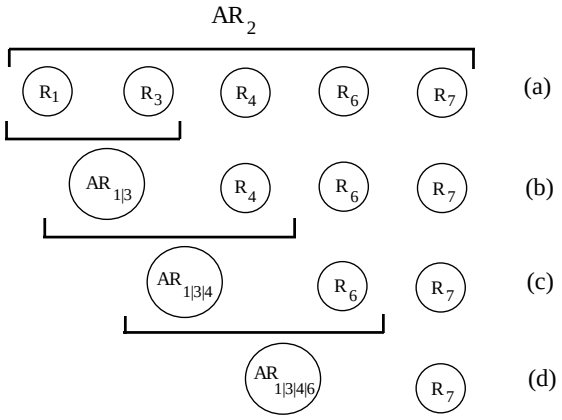

Figure 3. Recursive procedure to determine the distribution weight for an aggregation resource consisting of three or more resources.

```
        CREATE,EXPON(2),,1;              set atrib(1) = tnow
AR1     GOON,1;
                ACT,,.239987,A11;
                ACT,,.479986,A12;
                ACT,,.280027,A13;
A11     ASSIGN, ATRIB(3)=EXPON(1.33333);
                ACT,,,D1;
A12     ASSIGN, ATRIB(3)=EXPON(1.66667);
                ACT,,,D1;
A13     ASSIGN, ATRIB(3)=EXPON(1.42857);
                ACT,,,D1;
;
D1      Queue(1);
                ACT(1)/1,ATRIB(3);
                ASSIGN,ATRIB(2)=TNOW-ATRIB(1)-ATRIB(3);
                ASSIGN,ATRIB(4)=ATRIB(2)+ATRIB(3);
                COLCT,ATRIB(3),AR1 SERVICE TM;
                COLCT,ATRIB(4),AR1 CYCLE TM;
                ASSIGN,ATRIB(5)=ATRIB(4)*3;
                COLCT,ATRIB(5),AR1 TOTAL CYCLE;
```

Figure 4. SLAM II code for modeling the aggregation resource represent the series of three M/M/1 queues.