IESE Business School
University of Navarra

Working Paper
WP-202
December, 1990

LOADING RULES WITH A CONSTANT NUMBER OF JOBS IN PROCESS:
CAPACITY ANALYSIS AND APPROXIMATE CALCULATION OF SYSTEM PROPERTIES

Josep Riverola[1]

Jaume Ribera[1]

[1] Professor, Production, Technology and Operations Management, IESE

IESE Business School – University of Navarra
Av. Pearson, 21 – 08034 Barcelona, Spain. Phone: (+34) 93 253 42 00 Fax: (+34) 93 253 43 43
Camino del Cerro del Águila, 3 (Ctra. de Castilla, km 5,180) – 28023 Madrid, Spain. Phone: (+34) 91 357 08 09 Fax: (+34) 91 357 29 13

LOADING RULES WITH A CONSTANT NUMBER OF JOBS IN PROCESS:
CAPACITY ANALYSIS AND APPROXIMATE CALCULATION OF SYSTEM PROPERTIES

Shantikumar (1990), introduced the load function as a method for loading a plant. The idea is that the decision maker loads the plant by looking at the work in process and trying to keep it "right" in some way. Other forms of this rule are known as CONWIP rules (Hopp et al., 1989; Knessl and Tier, 1990; Lazowska et al., 1984). In practice, a variation of those rules is more easily implemented. The procedure, which we feel is one of the most common rules in practice, tries to maintain the *number of jobs in the factory roughly constant*. This is the rule we are going to analyze in this paper.

The reasons for delivering just a fixed number of all orders to the shop floor are many, and can be found in the desire to minimize confusion in the plant, avoiding trade-offs in order-processing by the decision makers at lower levels in the organization, etc.

In addition, often all classes of orders should not be pooled together without taking in account, for instance, their routing behavior. Thus the number of jobs in the factory may be a vector, since several classes of orders may be in process at the same time. For instance, orders for several products, or several types of customers, may be simultaneously considered in the loading of the plant. One may be interested in the individual performance of each class, delivery times being one of the most important measures of service in many cases.

Thus, we are led to the study of loading rules that try to maintain the number of jobs in *each of several classes roughly constant*. These are input-output rules, in the sense that their implementation does not need an actual knowledge of the state of the factory. The rule it is easily implemented by monitoring the outputs and sending a new job into the factory every time a job finishes processing and leaves. Obviously the job should belong to the same class of goods as the one leaving the factory. Like the CONWIP rules, the rule is also in line with Just in Time approaches, since a job starts just when another job finishes.

We will model the factory by a closed network of M/M/1 FIFO queues. This is a well known modeling approach, and sophisticated tools are available for its analysis (Walrand, 1988; Whitt, 1984). We are deliberately choosing a very simple model, although powerful modeling extensions are available in the literature, especially in the field of computer performance evaluation. Variable service rates, other priority disciplines and types of stations can be added to the model, if so desired.

Even if the decision rule of a constant number of jobs is not strictly enforced, we feel that modeling a factory as a closed network of queues, and analyzing its behavior as a function of the network population, is the right approach for estimating production times. Indeed, when modeling a production system as an open network of queues, and then using elementary queuing theory to estimate production times, or delivery lags, one runs into problems with the heavily loaded centers. A heavily loaded center has utilization of nearly one, and therefore an elementary queuing approach announces waiting time equal to infinity, not giving any intuitive hints of the magnitude of the times involved. This hinders the utilization of queuing formulas for crude estimation, and leaves practitioners without the benefits of a simple approach that can provide intuitive, even if approximate, estimations to the delivery lags involved.

Computing algorithms (Bruell and Balbo, 1980; Reiser and Lavenberg, 1980; Shantikumar and Gocmen, 1983; Walrand, 1988; Whitt, 1984) are readily available to analyze the behavior of such systems. From the point of view of the operations manager, concerned with the structure and performance of its factory, most of them suffer from two main drawbacks.

First, they are very analytical. They are good at computing system performance measures once given the system configuration and parameters, but the concepts involved are remote from the operations domain. They provide little insight into understanding the way the design of the system relates to the performance measures. In addition they do not have any improvement concepts built in.

Second, they work better for small populations. Indeed the exact analysis of a large system is hopeless for all of them. For instance, the MVA analysis would be impossible in terms of computer time or storage, for more than 10 classes and 10 individuals per class, since it involves the recursive calculation of all solutions for less than the final number of individuals in each class, 10 solutions in our case. In practice, a variety of ad-hoc approximating techniques are used for large populations (Sauer and Chandy, 1981). Most techniques obtain good approximations and reasonable computing time, but most of the time at the expense of further obscurities.

In this paper we try to provide an approximate, albeit somewhat crude, approach that we feel does not have some of the problems above. It is asymptotic and it is mostly inspired in (Knessl and Tier, 1990; MacKenna and Mitra, 1984, 1986; Walrand, 1988). The approach combines capacity planning ideas, resource constrained flow calculations with elementary queuing concepts to construct an integrated framework where waiting times are analytically related to the goodness of decisions on loading the factory.

Also, its degree of approximation is good for large populations (say for more than 20-40 jobs among all classes in the system) and degrades for small populations. Thus the proposed approach is complementary, in population terms, to the standard one. It also allows an easy analysis of the load-delivery lag trade-off. To this end, the computational approach we propose computes a whole one parameter load curve.

Throughout the paper, the intuitive reasoning needed is just an extension of the classical deterministic capacity analysis. We are led to identify bottlenecks and production flow rates in the production centers, compatible with the capacity constraints. Then we determine the loads on the centers, their utilization in terms of the network population, and use the asymptotic character of our results to compute the delays via some M/M/1 reasoning. Finally, we aggregate the populations involved in a composite population, and use this aggregation to adjust the actual delays due to congestion to the case of a finite population.

The paper is organized as follows: Section 1 is just a reminder of the basic MVA technique that forms the starting point of our approach. Section 2 develops an imbedding of the MVA equations. We get a set of functional equations for the desired system properties. The equations are complicated, but by introducing two slow variation hypotheses, satisfied among other instances when the population vector grows unbounded, we get simple approximate approaches to the full set of functional equations. In section 3 we study some properties of the asymptotic solutions derived above and show how MVA analysis integrates both with classical capacity analysis and elementary queuing analysis. Section 4 dwells in computational issues. Section 5 presents an "engineering procedure" that combines all concepts in a systematic three step procedure for the selection of a rule. Finally, section 5 presents a fully worked example.

1. MVA Analysis of a Closed Queuing System

We summarize here the classical MVA Analysis. Consider a closed Jackson network of M/M/1 queues, with several classes of customers. Each class of customers follows a route in the network of queues, visiting several centers and waiting FIFO whenever the center is busy. Once the job is finished, it is replaced by an identical one. The number of classes will be denoted by m. The number of centers is n. The service time S_i, depends only on the center and is identical for all classes in each center. More general systems can also be analyzed, but we feel the above model suffices for most applications, without introducing unnecessary complexities. The usual assumptions on those systems (Walrand, 1988) are assumed to hold.

Some additional notation:

$\mathbf{n}$	:	Population vector, with components n^i
$N_j^i(\mathbf{n})$:		Number of jobs in system for class i and center j
$W_j^i(\mathbf{n})$:		Waiting time for the class i in center j
$\lambda_j^i(\mathbf{n})$:		Arrival rate for class I at center j
V_j^i	:	Visit ratio

Let N stand for the sum of the elements of **n**, i.e. the total number of items in all classes. As usual, the visit ratios of class i to center j are given for each class by a solution of the routing equations (Walrand, 1988), stating the long range equilibrium of each class in the network. The visit ratios are given up to a constant factor that can be dependent on the class.

The theory of these networks is well understood, being the main result the so-called Arrival Theorem (see below). From this main result, one can derive the Mean Value Algorithm (MVA) (Reiser and Lavenberg, 1980), a procedure for computing mean values of numbers in system, waiting times and flow rates.

The Mean Value Analysis relies on three basic properties of closed Jackson networks of M/M/1 queues. The first is the *Arrival Theorem*: At jump times the customer that jumps sees the others with their invariant distribution, i.e. with the steady state distribution resulting from a population vector with the jumping customer removed. The second and third properties are

versions of *Little's Law*, applied to the *whole system* and *to each center,* respectively. The procedure proceeds to a solution via a recursion on the population vector.

The mean value analysis formulas for an m-class, n-center closed queuing M/M/1 system are (Reiser and Lavenberg, 1980):

$$W_j^i(\mathbf{n}) = S_j\left(1 + \sum_{k=1}^{m} N_j^k(\mathbf{n} - e_i)\right) \tag{1}$$

$$\lambda_j^i(\mathbf{n}) = \frac{n^i V_j^i}{\sum_{k=1}^{n} V_k^i W_k^i} \tag{2}$$

$$N_j^i(\mathbf{n}) = \lambda_j^i(\mathbf{n})\ W_j^i(\mathbf{n}) \tag{3}$$

for i = 1,..., m, and j = 1,...,n.

2. An Imbedding of the MVA Equations

In this section we derive an imbedding of (1)-(3) that is central to our approach. The idea is that for large values of the populations, it is possible to replace a discrete state space by a continuous one by taking fractions of the total population. This is a usual approach in asymptotic analysis (MacKenna and Mitra, 1984).

In the spirit of (Knessl and Tier, 1990; MacKenna and Mitra, 1984, 1986), we introduce a small parameter ε (going to zero) and define a vector b, with components $b^i = n^i \varepsilon$. One possible choice is ε = 1 / N, with N growing unbounded, in which case the b's are the proportions of each class in the total population, and they sum to 1. We do not restrict the b's to sum to one. Denoting their sum by B, we have ε = B / N.

To derive the alternative form of (1)-(3), we start by substituting b^i/ε for n^i in them, and defining[1]:

$$\nu_j^i(b) = \varepsilon N_j^i\left(\frac{b}{\varepsilon}\right) \qquad x_j^i(b) = \lambda_j^i\left(\frac{b}{\varepsilon}\right) \qquad \omega_j^i(b) = \varepsilon W_j^i\left(\frac{b}{\varepsilon}\right)$$

Equations (1), (2) and (3) become:

$$\omega_j^i(b) = \varepsilon S_j + S_j \sum_{k=1}^{m} \nu_j^k(b - \varepsilon e_i) \tag{4}$$

[1] All functions of b are at the same time functions of ε. Thus, we will normally suppress any explicit reference to ε. We also suppress the explicit dependence on b when it is clear from the context.

$$x_j^i(b) = \frac{b^i V_j^i}{\sum_{k=1}^{n} V_k^i \omega_k^i(b)} - \tag{5}$$

$$\nu_j^i = x_j^i \omega_j^i \tag{6}$$

If B = 1, it is easy to interpret the meaning of the above quantities. The ν are the proportions of the total population in each class and in each of the centers. They sum to B and for each class their sum is b^i. The x preserve their interpretation as flow rates, whereas the w are waiting times of class i customers in center j, per customer in the whole network.

Now, for convenience, define

$$y_j^i = \nu_j^i + \frac{\varepsilon}{m}$$

and

$$y_j = \sum_{k=1}^{m} y_j^k$$

Except for an additional term on ε, the y_j are the percentages of the total population accounted for all classes in center j. The unnatural term ε, will be justified on notational grounds later on. Observe that the denominator of (5) is independent of j. Therefore we can define the *flow rate of class i* as:

$$x^i = \frac{b^i}{\sum_{k=1}^{n} V_k^i \omega_k^i}$$

In terms of the flow rate of the class, it is possible to recover all other flow rates, by substituting in (5), as:

$$x_j^i = x^i V_j^i$$

Finally, define the matrix of loadings, $A = [a_{ij}]$ by:

$$a_{ij} = V_j^i S_j$$

Now substitute (6) in (4) and (5), and use the above definitions to get:

$$y_j^i(b) - x^i(b)\, a_{ij}\, y_j(b - \varepsilon e_i) = \frac{\varepsilon}{m} \tag{7}$$

$$x^i(b)\sum_{j,k} a_{ij}\, y_j^k(b-\varepsilon e_k) = b^i \tag{8}$$

Equations (7) and (8) give rise to the desired imbedding of the basic MVA ones (1)-(3). To this end, we allow any positive real values of the vector b and the parameter ε. Thus we are replacing the equations (1)-(3), defined on a countable set of values of (x,y), (one for each possible value of the class population vector **n**, a vector of integers) by a pair of functional equations (7)-(8) over the real vector $b \geq 0$ and real parameter $\varepsilon \geq 0$, having as unknowns two vector functions of b and ε: x(b,ε) and y(b,ε).

For future reference we record (7)-(8) for the single class case:

$$\begin{aligned} &y_j(B) - x(B)\, a_j\, y_j(B-\varepsilon) = \varepsilon \\ &x(B)\sum_j a_j\, y_j(B-\varepsilon) = B \end{aligned} \tag{9}$$

Notice that, in justifying the imbedding (7)-(8), implicit use has been made of the asymptotic character of the analysis.

Equations (7)-(8) are very difficult to solve, but by extended asymptotic considerations, approximated solutions can be obtained to a degree of approximation suitable for most practical production purposes.

There are several ways of simplifying (7) and (8), leading to approximate solutions. The two basic assumptions we will be considering in this paper are:

Hypothesis H0.- *The changes in the y's, solution of (7)-(8), produced by a (small enough) decrease in the i-th class population proportion, i.e. changing b^i by a small amount, are **independent** of the class i being changed, and only depend on the change of the B in the network.*

Hypothesis H1.- *The y's in the solution of (7)-(8) are **insensitive** to small enough decreases in the i-th class population proportion.*

Observe that H0 is weaker than H1, since H1 postulates insensitivity to changes, whereas H0 simply states that the change should be independent of the population being decreased.

H1 can be easily proved to be true when *the network population grows unbounded*. Thus our approach is basically sound and the approximation is good for large populations. However, the introduction of H0 extends the range of the approximation. Hypothesis H0 can hold in cases other than the large population case, for example in cases of populations with similar flow rates and sojourn times in the network.

H0 and H1 correspond to two fundamental ways to deal with a large network population. One is to assume that removing a customer changes the system in a way that does not depend on the class of the customer being removed. Thus, when a customer jumps, the state of the remaining network is roughly the same for all classes of jumping customers. The second is to assume that, to a first order, when the population is large enough, the effect of the removal of one customer is negligible.

a) *Hypothesis H0: The single class approximation*

In H0 we formally mean that, to a first order approximation:

$$y_j^i (b - \varepsilon e_k) = y_j^i (b - \varepsilon e_1) \qquad \text{for all values of } k.$$

The net result of the hypothesis is that the system behaves as a single class network, with loadings defined as:

$$a_j = \sum_{i=1}^{m} a_{ij} x^i \tag{10}$$

the x^i being the (unknown) rates in each of the individual classes. To see this, sum (7) and (8) over i and make use of the hypothesis to derive

$$y_j(b) - y_j(b - \varepsilon e_1) \; a_j = \varepsilon$$

$$\sum_{j=1}^{n} a_j \; y_j(b - \varepsilon e_1) = B$$

which shows that the flow rate for the reduced single class problem is x = 1, and the y's are identical to the ones in the original n class system.

Notice the following implication of the above. Assume a proportionality to be maintained in the flow rates of the network require them to be proportional to a vector v. Define the revised one class loadings as:

$$a_j = \sum_{i=1}^{m} a_{ij} v^i \tag{11}$$

Then, equations (9) for the reduced single class would solve the m class problem, providing an H0-approximate solution to the full system. Of course, if we started with a desired set of population proportions, b, this proportion may not hold for this solution. The desired proportionality of the populations, b, is only obtained by using the exact flow rates x^i in the calculation of the loadings (11). The closer the v's to the actual x's, the closer the solution of the aggregate one class problem will be to the exact solution of the initial m-class network. In this sense, H0 leads to an aggregation result, by showing how to aggregate classes in a single class, preserving in the process the properties of the whole system.

b) *Hypothesis H1: The asymptotic analysis*

Formally, H1 means we can replace the functions of $b - \varepsilon e_k$ by their corresponding functions of b in (7)-(8). Performing the replacement, (7)-(8) become:

$$
\begin{aligned}
& y_j \left(1 - \sum_{i=1}^{m} a_{ij}\, x^i \right) = \varepsilon \\
& x^i \sum_{j=1}^{n} a_{ij}\, y_j = b_i \\
& v_j^i = a_{ij}\, x^i\, y_j
\end{aligned}
\qquad (12)
$$

All variables in (12) are to be positive. The third equation is just a recipe to compute the v's. Thus, when we refer to (12), we usually mean the first two equations of (12), for all values of i and j. We call equations (12) the *asymptotic equations* and refer to their solution as an **asymptotic solution**. By summing the first and second equations, and replacing the second in the first, we see that the sum of the y_j is $B + n\,\varepsilon$, in accordance with their interpretation.

Notice that the term $a_{ij}\, x^i$ is the utilization of center j accounted by class i, ρ_{ij}, as easily derived from the definition of the loadings, a_{ij}, and of the class flow rates, x^i. The global utilization of center j denoted, as usual, by ρ_j is:

$$
\rho_j = \sum_{i=1}^{m} \rho_{ij} = \sum_{i=1}^{m} a_{ij}\, x^i
$$

The expression in parentheses in the first group of equations of (12) is thus $1 - \rho_j$. A (trivial) lemma in the appendix, asserts that $z_j = 1 - \rho_j \geq 0$, i.e. the slack in each center utilization must be non-negative. The slack is strictly positive whenever the center is not completely saturated, i.e. when the center is not a bottleneck for the current set of x's.

By including the above conditions into (12), using the vector of slacks z, and a new variable vector P, to be interpreted below, one gets the following alternative formulation of (12):

$$
\begin{aligned}
& A^T x + z = 1 \\
& A\, y - P = 0 \\
& y_j\, z_j = \varepsilon \quad , \quad j = 1, \ldots, n \\
& P^i\, x^i = b^i \quad , \quad i = 1, \ldots, m
\end{aligned}
\qquad (13)
$$

Letting $\varepsilon = 0$ in (13) we get the conditions for the **limiting solution**, i.e. the solution when the population goes to infinity.

Conditions (13) play a central role in our development and deserve additional interpretation. Again, we assume that the b's have been normalized to sum to 1. First, let us remark that conditions (13) for a given normalization vector b and ε, as we will see shortly, determine uniquely the values of the x's and the y's. The x's represent the (asymptotic) flows of each class, in the operations parlance the *production of each product class*. We usually want all x's to be strictly greater than zero, if any production of every class is ever to take place. The productions are constrained by the capacities of the production centers. This is the meaning of the first set of conditions in (13), restricting the flow of production in each center below its capacity (i.e. requiring the positivity of the slacks).

For a finite population ($\varepsilon > 0$) and more than one production center, no center will have utilization $\rho = 1$, i.e. $z = 0$. This is required by the third equation of (12) asking for the product of z_j and y_j to be equal to $\varepsilon > 0$. Thus, we can algebraically solve for y_j giving

$$y_j = \frac{1}{z_j} = \frac{\varepsilon}{(1 - \rho_j)}$$

Now, substituting the value of the y's in the second equation of (12) and replacing the definition of a_{ij} we get:

$$P^i = \sum_{j=1}^{n} a_{ij}\, y_j = \varepsilon \sum_{j=1}^{n} a_{ij}\, \frac{1}{z_j} = \varepsilon \sum_{j=1}^{n} V_j^i \frac{S_j}{(1 - \rho_j)} = \frac{1}{N} \sum_{j=1}^{n} V_j^i\, W_j(M/M/1)$$

The last equality comes from the presence in the next to last expression of the formula for the waiting time in an open M/M/1 queue, $W_j(M/M/1)$. Thus, the last sum gives the sojourn time of class i items in a system with every queue behaving as an open M/M/1 queue. Replacing the value of P^i in the last condition of (13), we get:

$$b^i = x^i P^i = \frac{1}{N} \sum_{j=1}^{n} x^i V_j^i\, W_j(M/M/1) = \frac{1}{N} \sum_{j=1}^{n} x_j^i\, W_j(M/M/1) = \frac{1}{N} \sum_{j=1}^{n} N_j^i(M/M/1)$$

where in the last expression we have also used the obvious notation for the number in system of class i in center j, when it is treated as an open M/M/1 queue. Therefore, multiplying by N, the last and first terms above give:

$$n^i = \sum_{j=1}^{n} N_j^i\,(M/M/1)$$

revealing that the asymptotic approximation is equivalent to *considering each center as an M/M/1 open queue, and requiring the total population of each class across all centers to be equal to the given population vector.* Thus, the asymptotic formulation models exactly the problem of finding rates x's, such that, in steady state, an open network of M/M/1 queues with several classes has a stated population vector **n.** The above provides an interpretation of P as the *total sojourn time* in the system of each class, per customer in the network.

The above implies the following facts which we record here for future reference:

1) the total number of items in center j is

$$N_j = \frac{(1 - z_j)}{z_j}$$

and thus the total number in the network, N, is the sum of the N_j.

2) The total production time per customer, sojourn time of class i per customer is:

$$P^i = \sum_{j=1}^{n} a_{ij} \, z_j^{-1}$$

As the population grows, ε decreases, some of the centers will become saturated, its capacity will be completely used, and thus they will become bottlenecks. In this case, most of the items will queue up behind bottlenecks. Actually, queues in non-bottlenecks will account for a negligible amount of the population. Thus for large populations, we can ignore non-bottlenecks. This remark allows an easy interpretation of the conditions (13) for ε = 0. The third conditions of (13) states that only bottlenecks can hold non-zero limiting proportions of the population. In addition, since bottlenecks have a utilization of 1, waiting time in a bottleneck is just the product of the service time times the number of items waiting in it.

3. Existence and Uniqueness of Solutions to the Asymptotic Conditions

We now study the properties of the solutions of (13). A central theoretical result in this respect is *that the conditions (13) and the natural non-negativity of the rates and times are the Karush - Kuhn -Tucker Conditions for the optimization in (x,z) of the problem:*

$$\begin{aligned} &\text{Max } \{ \sum_{i=1}^{i=m} b^i \ln(x_i) + \varepsilon \sum_{j=1}^{j=n} \ln(z_j) \} \\ &A\,x + I\,z = 1 \\ &x \geq 0 \quad z \geq 0 \end{aligned} \tag{14}$$

the y's being the multipliers of the NLP problem.

This shows that given b and ε > 0, a solution to (13) in (x,z,y,P) exists and is unique, having all x's and z's greater than zero. The result is easily checked by writing down the KTL conditions and observing that since all elements of A are positive, and the objective is strictly concave, a (unique) optimal solution exists that has all components $x^i > 0$ and $z_j > 0$. Then the two last conditions in (13) can be used to determine y and P uniquely. When ε = 0, the same argument shows that for every b, (13) has a unique solution in the variables x, z, P. However the y´s are not necessarily unique.

Conversely, given a vector x with $A^T x < 1$, unique values for ε, b (with B = 1), z, y, and P exist, solving (13). This is checked by simple algebra as follows: From x we can uniquely compute z. Using ε as a parameter, solve for y in the third condition and substitute in the second, getting P as a function of ε. Finally plug this P into the fourth condition and require the b's to sum to 1, getting from this requirement the unique value of ε.

A x solving $A^T x \leq 1$ gives rise to a limiting solution if *some slack is zero*. In this case, a constraint is saturated and the wait in this center is unbounded. Conversely, limiting solutions correspond to solutions of the above constraints with some slacks equal to zero. Otherwise by the third condition in (13) all y's should be zero, thus, by the second condition P = 0, contradicting the fourth condition.

Remember that the classical stationary capacity analysis problem would read in our case:

$$\begin{aligned} &\text{Max } f(x_1, \ldots, x_m) \\ &A^T x \leq 1 \\ &x \in X \end{aligned} \tag{15}$$

where X usually contains the set $x_i \geq 0$, plus some other constraints that help the decision maker define an acceptable solution. Thus, the constraints in (14) are (a subset of) the classical capacity analysis constraints, defining a limit to the production of each class at each production center.

A version of the NLP (14) appears at the core of some new algorithms for the polynomial solution of LP problems. As a function of ε it defines a family of weighted logarithmic barrier problems (Monteiro and Adler, 1989; Monteiro et al., 1990) for an LP with constraints $Ax + I z = 1$ and objective function identically zero, i.e. for a pure feasibility problem for the constraints. Compare, for instance, with the following *weighted logarithmic barrier prob*lem for the LP capacity problem with objective function cx:

$$\begin{aligned} &\text{Max } \left\{ c x + \mu \sum_i d_i \ln(x_i) + \mu \sum_j h_j \ln(z_j) \right\} \\ &A^T x + I z = 1 \\ &x \geq 0 \ , \quad z \geq 0 \end{aligned} \tag{16}$$

where d_i, $h_j \geq 0$ are the weights and $\mu \geq 0$ is a parameter. Remembering that the (weighted) central path of a LP is the locus as a function of μ of the solutions of (16), a one parameter family of approximating penalty problems, we see that conditions (13) are the defining conditions of *a point in the central path* of the pure feasibility problem, a point characterized by $\mu = \varepsilon$ and weights $(1, b/\varepsilon)$.

The role played by the central path in the new LP algorithms is, however, substantially different than the one played here. There, the calculation of the central path is just a way of approaching in polynomial time, and as μ goes to zero, the solution of the original LP. Here we are never approaching the solution of the LP problem, since, although we are letting μ go to zero, one of the weights is adjusted increasingly so that the penalty term to whom it applies never vanishes. Moreover, we are interested not so much in the solution for $\mu = 0$ as in points of the path for different weighting vectors, since each point provides a point of a load curve.

Actually conditions (13), in terms of the parameters ε and b, define a *central surface* on which we can define many paths. All functions [x(μ), z(μ), y(μ), P(μ), b(μ), e(μ)] of a single scalar parameter μ over the algebraic variety defined by (13) are paths of possible interest, depending on the operations management significance of the invariants defining them. The engineering solution, to be presented later, hinges on exploiting the operational properties of some such paths.

4. Solving the Approximating Problems Given ε and b

In this section we consider numerical procedures to solve both the asymptotic problem and the single class approximation when ε and the population vector b are given.

a) The asymptotic problem

In principle, to solve (13) we could solve the related NLP problem (14). Although (14) is a simple NLPs, it belongs to the family of penalty function problems known to be badly conditioned. An alternative approach would be to use Newton's Method in a way reminiscent of the interior point LP methods to find a point in the central path. However, and according to the collected computational evidence, iterative procedures do the job much faster and are easier to implement and understand. From this point of view (14) is more interesting than practical.

To solve (12) for a given b and ε we write it as

$$x^i = \frac{b^i}{\sum_{j=1}^{n} a_{ij}\, y_j} \qquad (17)$$

$$y_j = \varepsilon + y_j \sum_{i=1}^{m} a_{ij}\, x^i$$

and use this form to compute recursively values of y_j. It is important to start with an initial $y > 0$ with the sum of its components equal to B+ n ε. We use this initial set of y's to calculate the x's in the first set of equations (17), for i=1,..., m, and substitute the result in the right hand side of the second set of equations, obtaining a new value for the y's. The procedure is repeated until a selected tolerance is satisfied between two successive values of the y's.

By decreasing the value of ε in steps dε, from a given initial value ε_0, and proceeding towards zero, and by restarting each time the procedure from the solution to the previous value of ε, the above procedure can be used to trace a path of asymptotic solutions, leading to a limiting solution with ε = 0.

b) The solution of the single class problem (9)

The exact solution of (9) can be obtained for N integer by applying the full MVA analysis to it. However, for small ε the full MVA procedure requires us to perform 1/ε iterations. Once again, this fact makes the use of an approximation convenient. The Schweitzer (Lazowska et al., 1984) approximation is usually quite accurate and we recommend it for this case. The Appendix shows that the x is the positive root of the equation:

$$1 = \sum_{j=1}^{n} \frac{a_j\, \varepsilon\, x}{1 - (1 - \varepsilon)\, a_j\, x} \qquad (18)$$

Given x, a bit of algebra shows that the y's can be computed as:

$$y_j = \frac{a_j\, \varepsilon\, x}{1 - (1 - \varepsilon)\, a_j\, x} + \varepsilon$$

5. An "Engineering" Procedure

A crude summary of the foregoing results could claim that what we have accomplished is a kind of integration-separation property. In the asymptotic problem we have simultaneously *integrated the relationship of constrained production flows with the waiting times resulting from queuing* and shown the *separability of the constrained-capacity and queuing-time reasoning.*

The x's determine the production characteristics of the system. They are responsible for the direct profits of the operation. The slacks, the z's, are responsible for the waiting properties. They originate the delivery time and associated inventories. You can compute the production characteristics via a straight forward flow analysis with constraints. Then use elementary queuing theory to deduce the waits from the slacks in the capacities.

Since in the following we will have $\varepsilon > 0$ most of the time, we find it easier to use a version of (13) where all conditions, except for the first one, have been divided on both sides by ε, and a change of variables has been made, redefining P as P/ε and y as y/ε. This removes the character of "proportions per customer" of the problem variables, which become the full values of the associated properties. We then get the equivalent system:

$$
\begin{aligned}
& A^T x + z = 1 \\
& A\, y - P = 0 \\
& y_j\, z_j = 1 \qquad j = 1, \ldots, n \\
& P^i\, x^i = \mathbf{n}^i \qquad i = 1, \ldots, m
\end{aligned}
\tag{13)'}
$$

We will refer to the variables in (13)' as the **unnormalized variables** (total values) and the ones in (13) as the **normalized variables** (values per customer in the system). Notice that x's and z's are the same in both cases.

A practical procedure to design an effective control rule of the type being analyzed has to concentrate on choosing an operating point **n**. This kind of specification is desirable from the control point of view. Making sure that the control rule implemented is the one desired is just a problem of input output monitoring. Starting from the right **n**, it is enough to replace every finished order by another for the same product to make sure the rule is implemented.

On the other hand, it is not at all obvious how to pick a good **n**. This is not a primary design parameter for the operations manager, one that relates with his or her objectives or performance measures. As we have said, **n** is more in the nature of a control parameter, one that allows easy control of the system operation. The x's are more primary. It is easier to define a criterion of goodness for a set of x since they are part of the mental framework of the operations manager being the production flow rates.

If a cost function could be specified for each of the sojourn times, a nonlinear problem would probably be a good approach to picking **n** by combining the contribution margin of the production with the penalty for long sojourns. Indeed, if we assume that g(P) is such a function, the *design problem* could be formulated as:

$$\text{Max } f(x) + g(P)$$

$$A^T x + z = 1$$

$$P = A\,y$$

$$y_j\, z_j = 1$$

$$x \geq 0,\ z \geq 0$$

From a solution of the design problem, the control rule could be computed as $\mathbf{n} = P\ x$. (the condition of (13) missing among the design problem constraints). Thus N would be equal to the sum of the components of **n**, and b would be the **n** normalized. If we do not bother about numerical problems, this NLP can be even reduced to the following unconstrained NLP

$$\text{Max } f(x) + g(A\,[1 - A^T x]^{-1})$$
$$x \geq 0$$

where the inverse of a vector is the vector whose components are the inverses of the original components. The reason for the reduction lies on the fact that the quantity in square brackets is a penalty function for the constraints.

However, we feel that from the operations manager perspective, the choice is better approached as the selection of a point in a "good set" of low dimension, goodness being characterized by some a priori desired property instead of a full approach to equations (13)'. Thus, from this point of view, we are led to the load curve as a tool for policy selection. To do this requires first a selection of the kind of load curve, equivalent to the selection of a one parameter "good set," and then picking the parameter for the selected point in the curve.

Many families of load curves could be employed. For instance, the load curve for a constant proportion among classes, or the curve for a constant b, and parameter ε, could be calculated by the methods in section 4a. In the following we put forth a procedure based on the calculation of the load curve for some selected a priori path of the x's, x(μ), selected because of production considerations.

Thus, from the operations manager point of view, we approach the computation of the operating policy as a *three phase* procedure:

a) First you decide on your *production strategy* taking in account the competitive behavior of the firm. This means **choosing a set of x's allowing the company to serve its customers needs** in a satisfactory way.

b) Second, you *plan* the production system, meaning you **chose a desired load curve, a one parameter locus for the x's** on the central surface defined by (13).

c) Third, you examine the *operation* of the design, describing the variation of its throughput and its production time as the load in the system (the path parameter) changes along the selected curve. Then, you select **one value of the load curve parameter as the operating point** in the central surface.

Obviously, in practice the procedure would be iterative since the operation of the system may mean a reconsideration of its design.

a) *Solve a classical capacity analysis problem* (16)

The initial phase concentrates *in the pure capacity problem* (16). This is because (16) is closer to the operations manager perception of the global structure of the situation, obscuring boring details. Given the technology structure of the factory, the matrix A, the decision maker just has to pick a set of x's the production rates at each center, appropriate to the way of competing of the firm, i.e. of its approach to the market. You would probably like to introduce some side constraints to adjust the problem to your view of the world. We see that classical capacity analysis fits in nicely with this approach. Capacity analysis is the limiting technique as it belongs to the domain of the limit problem. It concentrates on picking a set of x's, without regard to timing or congestion considerations.

The result of this step will give a **desired set of x's**, in principle chosen without concern for the time performance of the production system. Also, the result of this exercise will be the identification of bottlenecks, that is, production centers whose corresponding constraint holds as an equality for the chosen solution. Since this may be a limiting solution of (13), because of the existence of bottlenecks with z=0, the value of the sojourn times may be infinity.

b) *Decide on the operating path*

There are, at least, three natural approaches to selecting an operating curve:

a) *Proportional production.* Select the path $x = \mu v$ for a fixed vector v, probably given by the x's in the previous solution to the limit problem. i.e. v is such that $v \geq 0$, $ATv \leq 1$, $v \oplus \varepsilon X$, and is optimal in some sense. This is the well known practice of scaling down the production flows to reduce the load on the centers. We have seen this strategy applied under many forms. For instance, it may mean keeping reserve capacity, performing less than full loading, etc.

b) *Parametric paths.* Paths obtained by applying parametric programming over the RHS to the pure capacity problem. This is a generalization of a) above. In this case, the x's form a piecewise linear path. Along this path the optimal value of the objective function in (15) decreases monotonically.

c) *Proportional sojourn times.* A particular case of parametric path is the path of proportional y's, or proportional sojourn times. We select a vector of y's so that the P's are in some desired proportions. Call this vector y0. For instance it could be obtained by a program of the form:

$$\text{Max} \quad g(Ay)$$

$$\sum_{j=1}^{n} y_j = 1 + n$$

$$y \geq 1$$

where the lower bound on y comes from the upper bound on z through the condition $y_j z_j = 1$. A path of the form $y = \mu y_0$ is then equivalent to the path $z = 1/(\mu y_0)$, which can be written, by redefining the parameter, as $z = \mu 1/y_0$. This is a parametric change in the RHS of the capacity problem, reducing the case to b).

c) *Pick a point in the loading curve*

Now we have to compute the performance characteristics of the points along the loading curve. We will refine the degree of approximation provided by H1, especially for small populations, by adjusting to H0.

Keep in mind that the asymptotic approximation treats the system as a network of M/M/1 queues with occupations 1-z. Then, given a path $x(\mu)$, and letting $r(\mu) = A^T x(\mu)$, the system properties of the asymptotic approximation (H1) are given by:

$$\begin{aligned} z(\mu) &= 1 - r(\mu) \\ y_j(\mu) &= \frac{1}{z_j(\mu)} \\ P^i(\mu) &= \sum_{j=1}^{n} a_{ij}\, y_j(\mu) \\ \mathbf{n}^i(\mu) &= P^i(\mu)\, x^i(\mu) \end{aligned} \qquad (19)$$

H1 suffers, as we have said, from serious drawbacks whenever the population is small. Therefore we propose to use here the single class approximation, exploiting the aggregation result explained in section 3. To this end, we combine H1 with the single class analysis, in an unified approximation scheme that refines the load curve under H0. By the remarks following (10), if we knew the right $x(\mu)$'s we could solve the H0 approximation to (7)-(8) as a single class problem. The idea is to use, at every point μ, the solution of the asymptotic problem as an estimate for the x, calculate the loadings for a related single class problem and solve it. With the solution (ξ,ψ) for the single class problem, we once again slightly revise the b's to get the desired H0 solution of (7)-(8) for each point in the curve. Formally:

Approximate solution of (7)-(8) under H0. *Given b and ε, let* (X, Y) *be a solution of (12) and* (ξ, ψ) *a solution of (9) with*

$$a_j = \sum_{i=1}^{m} a_{ij}\, X^i \qquad \textit{and} \qquad B = \sum_{i=1}^{m} b^i$$

If we compute

$$\begin{aligned} x^i &= X^i\, \xi \\ y_j^i &= \frac{a_{ij}\, X^i}{a_j} (\psi(B) - \varepsilon) + \frac{\varepsilon}{m} \\ \beta^i &= \sum_j y_j^i \end{aligned}$$

Then (x,y) solve (7)-(8) under H0 with the second member of (8) replaced with β. Also

$$\sum_i \beta^i = B \quad \textit{and} \quad \beta = b + o(\varepsilon^2)$$

The only a priori approximation is in the definition of x. The rest is exact under H0, coming from a rewriting of (8) using H0, substitution of $\psi_j(B-e)$, computed via the single class equations as

$$\psi_j (B - \varepsilon) = \frac{\psi_j(B) - \varepsilon}{\xi a_j}$$

and solving for the desired variables. Notice that the final ν's are just a proportional split of the ν in the single class problem with coefficients $a_{ij}X^i = \rho_{ij}$, i.e., the traffic inten sities of each class at the center.

6. Example

We assume 3 classes, 4 centers and the visit ratios and service times of the table below.

	Visit Ratios			Service
Center	Class 1	Class 2	Class 3	Time
1	1	1	1	5
2	0	1	0	10
3	0	0	1	14
4	1	1	1	12

the matrix A is then

5	0	0	12
5	10	0	12
5	0	14	12

Margins per unit of production are 1, 2 and 3 monetary units respectively.

a) *Solving the capacity planning problem*

Assume we solve the capacity planning model to obtain x = (0, 0.0119, 0.00714) with bottlenecks in centers 3 and 4, and thus z = (0.5833, 0.8810, 0, 0). Assume this solution is not adequate since we have commitments on product 1, derived from our approach to competing in products 2 and 3. By playing with constraints and from our knowledge of the company we find that an acceptable solution is x = (0.01, 0.014, 0.05), slacks z = (0.63, 0.86, 0.3, 0.112), y = (1.5873, 1.1628, 3.3333, 8.9286), P = (115.08, 126.71, 161.75) and resulting in parameter values of b = (0.1045, 0.1611, 0.7344) and e = 0.09081, equivalent to **n** = (1.1508, 1.7739, 8.0873) and N = 11.012. The production flows and delays are referred to the unit of time used in specifying S_j. If we assume that in this case the S's were given in hours, the production flows are in units per hour and the waits in hours.

b) *Deciding on the operating path*

Now we decide on the load curve of the system. Assume that a desirable curve is proportional production. This may be because we want to reduce the activity, but keeping the market proportions of the products in the proportion present in the currently selected x.

We initially use the formulas (19). We have x(μ) = μ (0.01, 0.014, 0.05). The value of r(μ) = μ (0,37, 0,14, 0,7,0,888). Thus the slacks are given by z(μ) = 1 - μ (0,37, 0,14, 0,7,0,888). Since slacks are constrained to be positive, m can take any value in the range [0, 1 / 0.888] = [0, 1,126]. Figures 1-7 show the performance characteristics of the system. We display all quantities as functions of the number of jobs in system, N.

Figure 1

Value of the Flows x, with the Resulting Load

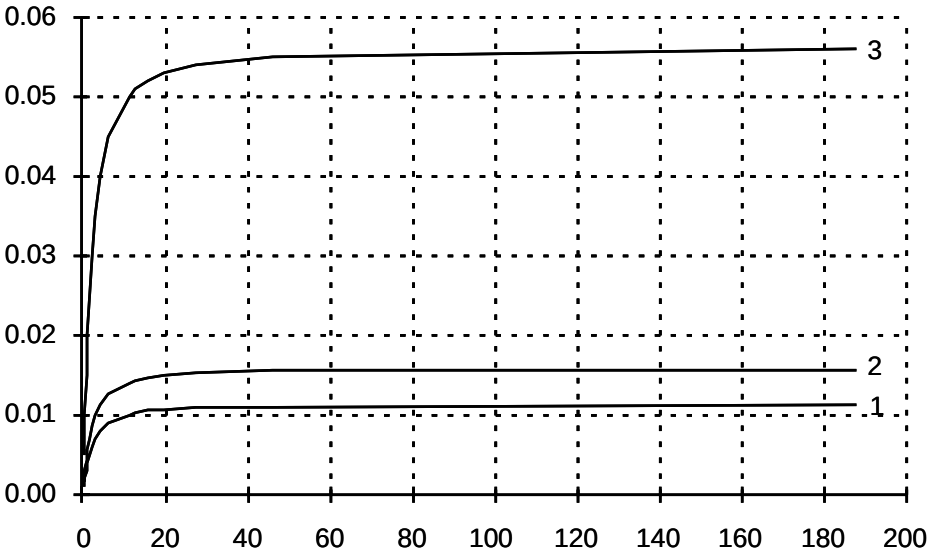
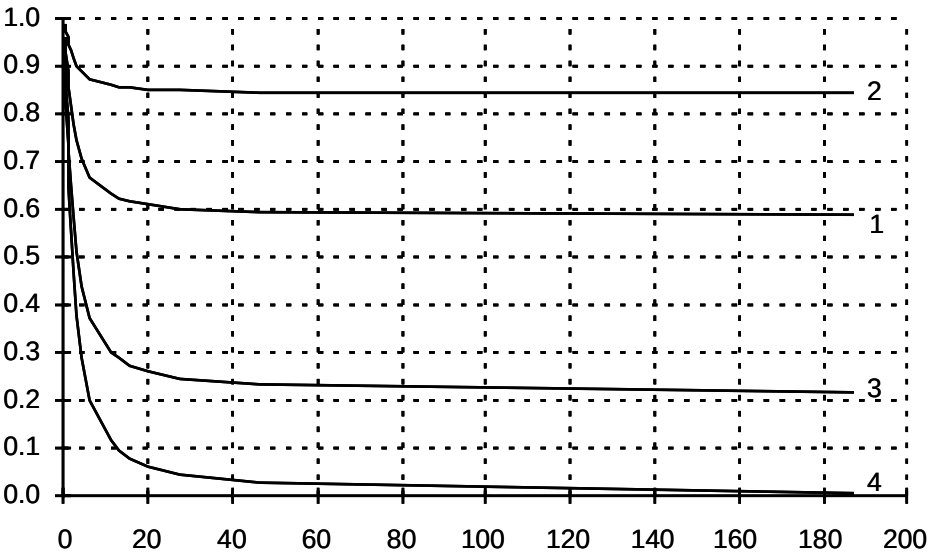


Figure 1 is just a restatement of Figure 7, since in the actual calculations x has been taken as the independent variable, and the total number of jobs in system has been calculated as the sum of the components of **n.**

Figure 2

Values of the Slacks, z, at Each Center



The slacks in Figure 2 show center 4 as the bottleneck of the operation. Here we see how fast the values of the slacks stabilize to non-zero values, meaning that the queuing in front of centers 1-3 builds up quickly to their open queue values. From then on, as we will see, all queuing takes place in front of center 4.

Figure 3

Normalized Values for the y's as a Function of the Load

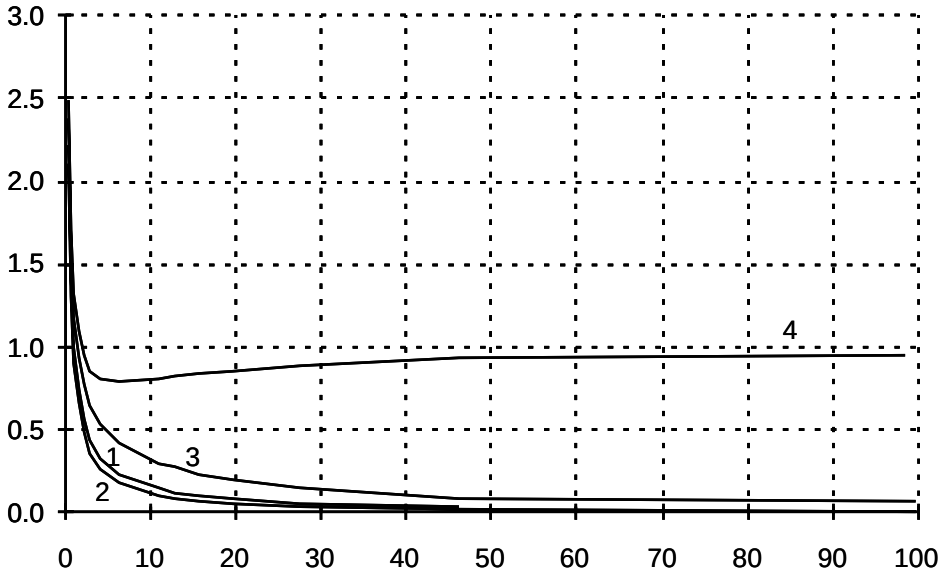
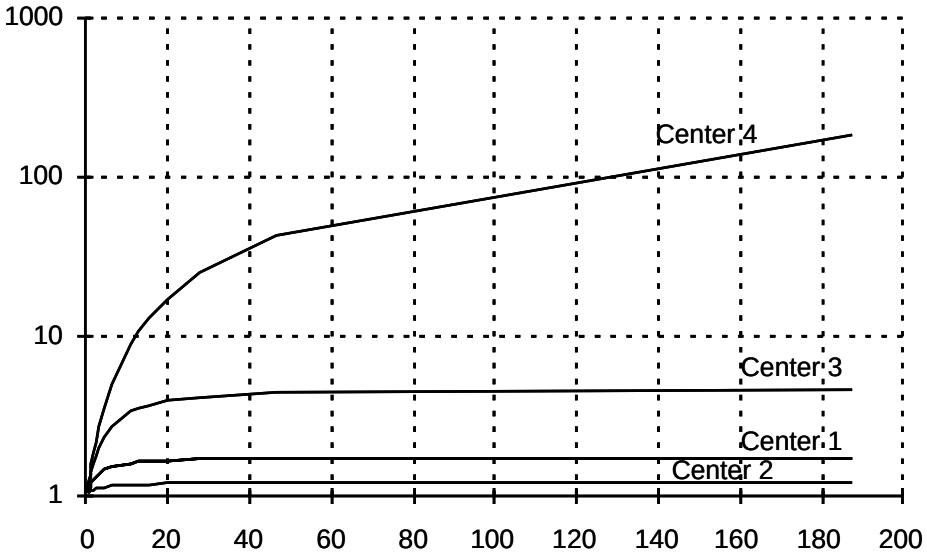


Figure 3 shows the y's, related to the proportion of the total population in each center. The result is as expected. The funny shape of y_4 is probably due to the changes in the population composition at low values of μ.

Figure 4

Unnormalized Values of y. Notice the Logarithmic Scale on the Ordinates



The unnormalized values of y in Figure 4 dramatically show the accumulation of items in center 4 as the load increases. The following figure, Figure 5, shows the behavior of the total sojourn time. It explodes, converging to a constant slope for all centers. We will see in Figure 6 that the common slope is responsibility of the service time of center 4, which is the slope is 12 (whose slope is 12??); since all waiting takes place in center 4, adding a unit to the system just increases the delay of everybody else by the amount of service.

Figure 5

Unnormalized Values of P, i.e., Total Sojourn Times

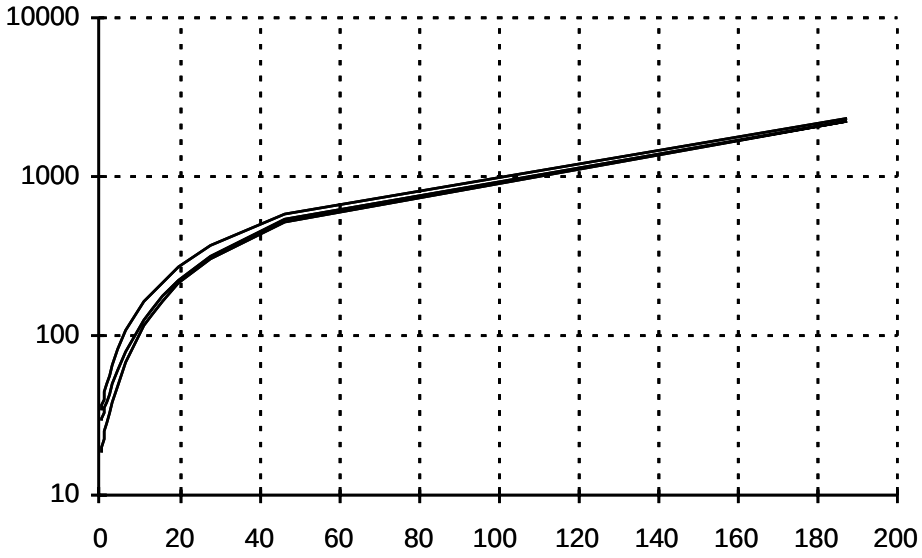
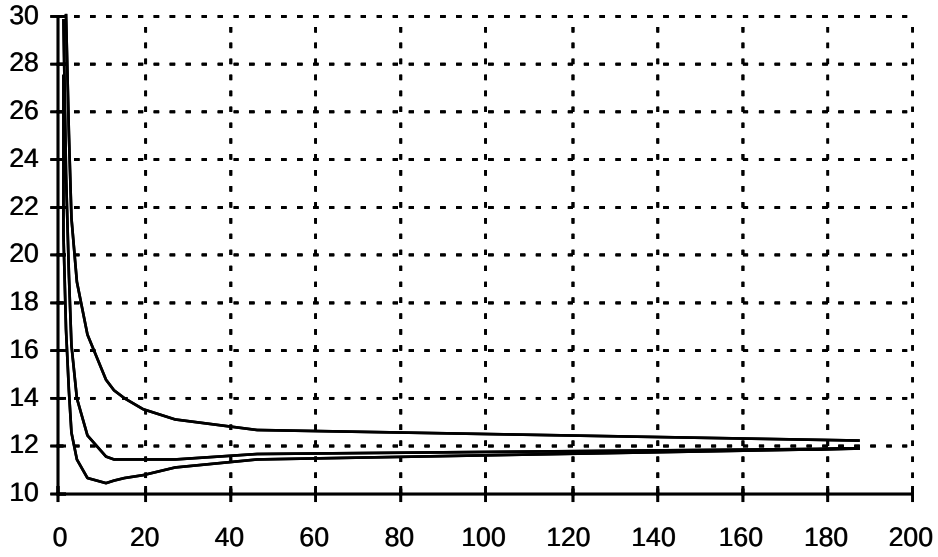


Figure 6

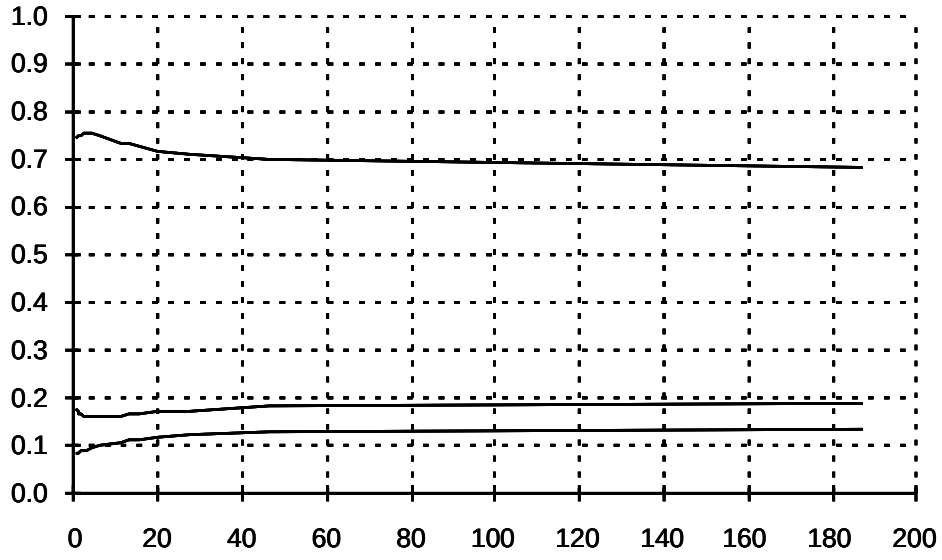
Normalized Sojourn Times



Finally, Figure 7 shows the control rule. We see that for values of the total load larger than 40 jobs, the proportion of jobs in each class remains roughly the same. This behavior shows that the analysis has also produced an approximate loading curve for a constant proportion of jobs.

Figure 7

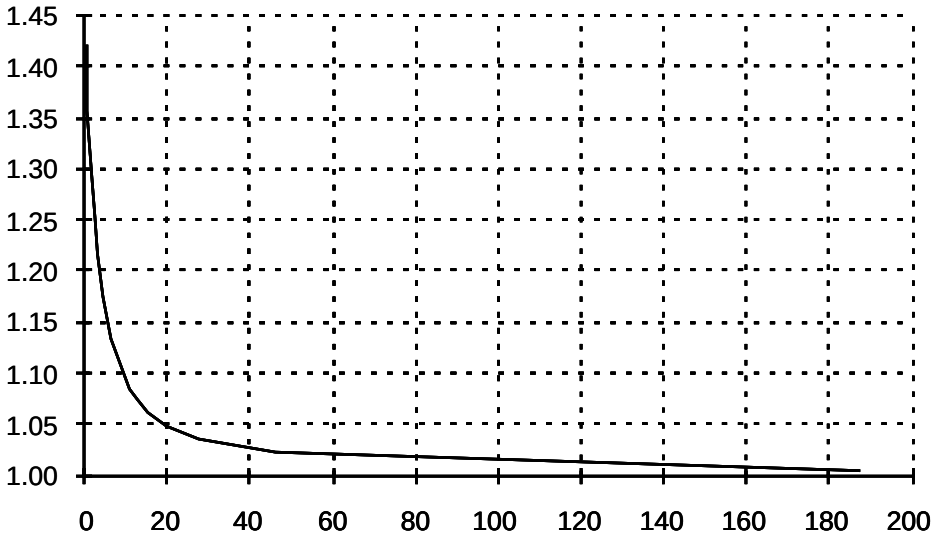
Normalized Values of the n's. (b's)



Finally, we show in Figure 8 the solution ξ of the one class problem that gives an indication of how close the asymptotic analysis is to the true solution of the problem. It may be seen that for values of the total population of 10 or more, the correction to the asymptotic x's, introduced by H0 is less than 10%. Thus H1, with its simple implications, is a true engineering tool for production systems design.

Figure 8

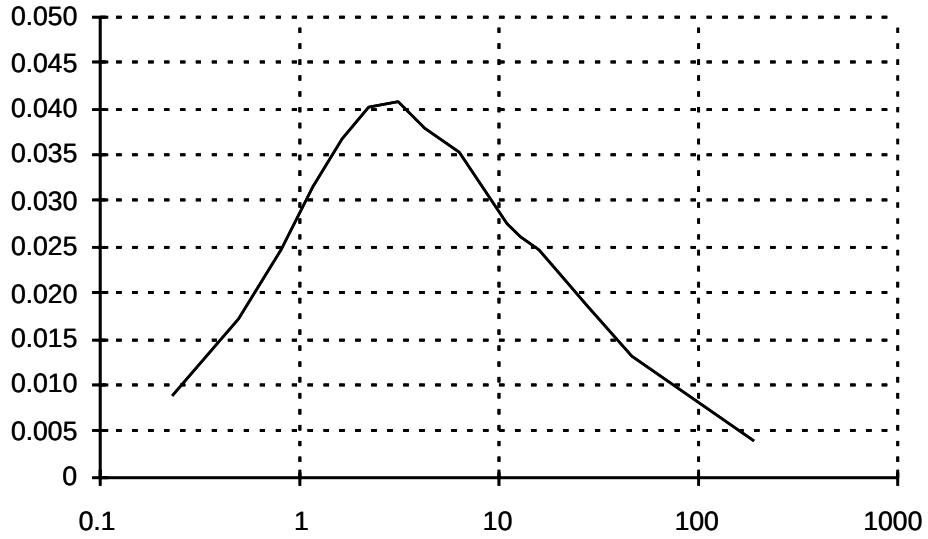
Single Class ξ Solution (Correction Multiplier)



Although we do not have full computational evidence of the degree of approximation to the true solution, we have applied the full Schweitzer approximation to our example and we record in Figure 9 the maximum percentage of difference between both approximations of the total number in system at each center.

Figure 9

Percentage of Difference with Respect to Schweitzer Method

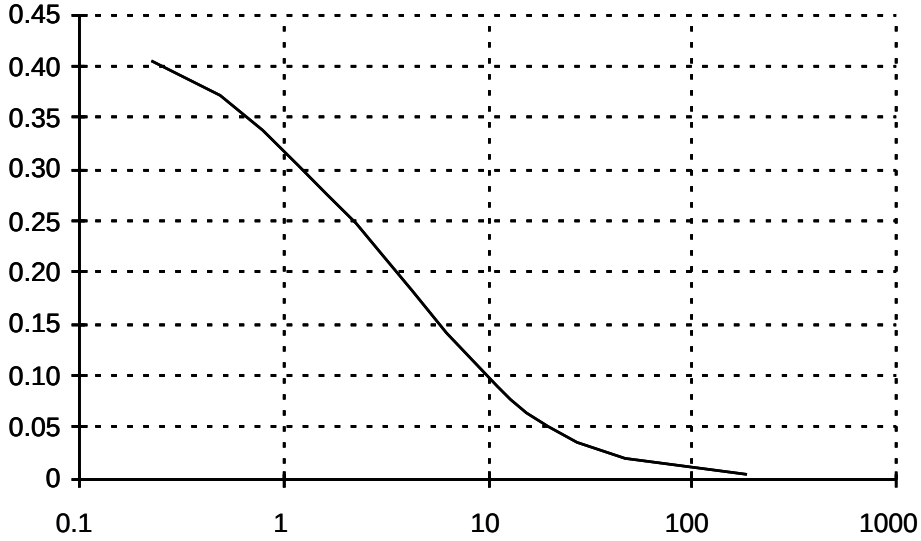


The maximum error is well below 5%, an adequate value for all practical purposes. The precision gained by computing the single class approximation exactly, i.e. using the MVA exact formulas is small due to the truncation errors in the imbedding.

Finally Figure 9 shows the relative differences in the x's. Although much greater, observe that for values of the total population above 10, the percentage of difference remains below 10%, again consistent with the approach.

Figure 10

Relative Differences Among Between the x's



Appendix I

A lemma

Lemma. For all **n** and j , $A^T x \leq 1$, and therefore $\rho_j \leq 1$.

Proof.

a) For all n, i and j it is true that:

$$1 + \sum_{k=1}^{m} N_j^k (\mathbf{n} - e_i) \geq \sum_{k=1}^{m} N_j^k (\mathbf{n})$$

Indeed, the left hand side would be the number of customers of the class in center j if the additional one in the class were assigned totally to center j. The right side is the actual number of customers that should be less than the above.

b)

$$\sum_{i=1}^{m} \lambda_j^i (\mathbf{n}) = \sum_{i=1}^{m} \frac{N_j^i (\mathbf{n})}{S_j \left(1 + \sum_{k=1}^{m} N_j^k (\mathbf{n} - e_i) \right)} \leq \frac{1}{S_j} \sum_{i=1}^{m} \frac{N_j^i (\mathbf{n})}{\sum_{k=1}^{m} N_j^k (\mathbf{n})} \leq \frac{1}{S_j}$$

The first equality follows from Little's Law and the recursion on waiting time. Since $\lambda(\mathbf{n}) = x(\mathbf{n}\ \varepsilon)$, the lemma follows by multiplying both sides times S_j.

Appendix II

Solving (18) by Newton's Method

In our notation the approximation assumes, in the formulas (7)-(8), that

$$\nu_j^i (b - \varepsilon e_k) = \begin{cases} \nu_j^i (b) & \text{if} \quad k \neq i \\ (1 - \varepsilon) \nu_j^i (b) & \text{if} \quad k = i \end{cases}$$

For the single class case, with n centers, the approximation specializes to:

$$\nu_j (B - \varepsilon) = (1 - \varepsilon) \nu_j (B) \qquad \text{or} \qquad y_j (B - \varepsilon) = (1 - \varepsilon) \nu_j (B) + \varepsilon$$

Substituting in (9), solving for x in the second equation, replacing it in the first equation and remembering that because of normalization B = 1, we get:

$$\nu_j = \frac{\varepsilon a_j + (1 - \varepsilon) a_j \nu_j}{\varepsilon \sum_j a_j + (1 - \varepsilon) \sum_j a_j \nu_j}$$

Therefore, to solve the approximate MVA equations we have to find a fixed point ν_j of a nonlinear mapping. It is possible to either carry out a recursive application of (18) starting from a n_j, obtained from the y's of some previous approximation by subtracting ε from each one of them, or to find a value of x very easily by finding a root of a polynomial. Indeed, we can reduce the problem to that of finding an eigenvalue by making a projective transformation. Define:

$$z = \varepsilon \sum_{j=1}^{n} a_j + (1 - \varepsilon) \sum_{j=1}^{n} a_j \nu_j \tag{20}$$

and $v_i = \nu_i / z$. Then the fixed point problem is equivalent to finding $\alpha > 0$, $v \geq 0$, $z > 0$, solving the eigenvalue problem

$$\begin{aligned} \alpha v_j &= \varepsilon a_j z + (1 - \varepsilon) a_j v_j \\ \alpha z &= z \, \varepsilon \sum_j a_j + (1 - \varepsilon) \sum_j a_j v_j \end{aligned} \tag{21}$$

Appendix II (continued)

The characteristic polynomial is obtained by solving for y_j in the first set of equations of (21) and sticking the result in the last set. From (20), we see that $\alpha = 1/x$. Thus, after some manipulations we find that x should be a solution of f = 0, where:

$$f = \sum_{j=1}^{n} \frac{a_j \varepsilon x}{1 - (1 - \varepsilon) a_j x} - 1$$

whose derivative is:

$$f' = \sum_{j=1}^{n} \frac{a_j \varepsilon}{[1 - (1 - \varepsilon) a_j x]^2}$$

The Newton iteration is therefore:

$x^+ = x^c - f / f'$

A good starting value is x = 1. No more than 2-5 iterations are needed to converge to very strict tolerances, since the function is very "steep" in the neighborhood of the solution.

References

(1) Bruell, S.C. and G. Balbo (1980), "Computational Algorithms for Closed Queuing Networks," New York: Elsevier Noth Holland, p. 190.

(2) Hopp, W.J. and M.L. Spearman (1989), "Throughput of a Constant WIP Manufacturing Line Subject to Failures," Department of Industrial Engineering and Management Sciences, Northwestern University, Evanston. IL.

(3) Knessl, C. and C. Tier (1990), "Asymptotic Expansions for Large Closed Queuing Networks," *Journal of ACM*, **37**, pp. 144-174.

(4) Lazowska, E.D., J. Zahorjan, G.S. Graham, and K.C. Sevcik (1984), "Quantitative Systems Performance: Computer System Analysis Using Queuing Network Models," Englewood Cliffs (New Jersey), Prentice Hall.

(5) MacKenna, J. and D. Mitra (1984), "Asymptotic Expansions and Integral Representations of Queue Lengths in Closed Markovian Networks," *Journal of ACM*, **31**, pp. 346-360.

(6) MacKenna, J. and D. Mitra (1986), "Asymptotic Expansions for Closed Markovian Networks with State-Dependent Service Rates," *Journal of ACM,* **33**, pp. 586-592.

(7) Monteiro, R. and I. Adler (1989), "Interior Path Following Primal-Dual Algorithms. Part I: Linear Programming," *Math. Programming*, **44**, pp. 27-42.

(8) Monteiro, R., I. Adler, and M. Resende (1990), "A Polynomial-Time Primal-Dual Affine Scaling Algorithm for Linear and Convex Programming and Its Power Series Extension," *Math. of O.R.*, **15**, pp. 191-214.

(9) Reiser, M. and S.S. Lavenberg (1980), "Mean Value Analysis of Closed Multi-chain Networks," *Journal of ACM,* **27** (2), pp. 313-332.

(10) Sauer, C.H. and K.M. Chandy (1981), "Computer Systems Performance Modeling," Englewood Cliffs (New Jersey), Prentice Hall.

(11) Shantikumar, J. (1990), "The Load Curve and its use in Production Planning," *Decision Sciences.*

(12) Shantikumar, J.G. and M. Gocmen (1983), "Heuristic Analysis of Closed Queuing Networks," *Int. J. Prod. Res.*, **21** (5), pp. 675-690.

(13) Spearman, M.L. (1990), "Customer Service in Kanban and CONWIP Production Systems," *Department of Industrial Engineering and Management Sciences*, Northwestern University, Evanston. IL.

(14) Spearman, M.L. and M.A. Zazanis (1990), "Push and Pull Production Systems: Issues and Comparisons," *Department of Industrial Engineering and Management Sciences*, Northwestern University, Evanston. IL.

(15) Walrand, J. (1988), "An Introduction to Queuing Networks," Englewoods Cliffs, Prentice Hall International.

(16) Whitt, W. (1979), "Open and Closed Models for Networks of Queues," *AT&T Bell Labs Tech. J.*, **63**, pp. 1911-1979.