2000

Determining inventory levels in a CONWIP controlled job shop

Sarah M. Ryan
Iowa State University, smryan@iastate.edu

Bruno Baynat
Universite Pierre ef Marie Curie

F. Fred Choobineh
University of Nebraska - Lincoln

Determining Inventory Levels in a CONWIP Controlled Job Shop

Sarah M. Ryan*
Senior Member, IIE
Department of Industrial and Management Systems Engineering
University of Nebraska-Lincoln
Lincoln, NE 68588-0518

Bruno Baynat
Laboratoire d'Informatique de Paris 6
Universite Pierre et Marie Curie
4, place Jussieu
75252 PARIS CEDEX 05

F. Fred Choobineh
Senior Member, IIE
Department of Industrial and Management Systems Engineering
University of Nebraska-Lincoln
Lincoln, NE 68588-0518

November, 1997
Revised July, 1998
Revised December, 1998

For publication in *IIE Transactions* Special Issue on Manufacturing Logistics

This is an Accepted Manuscript of an article published by Taylor & Francis in IIE Transactionson in 2000, available online: http://dx.doi.org/10.1080/07408170008963883

* Corresponding author: smryan@unlinfo.unl.edu
Phone: 402-472-1384
Fax: 402-472-2410

Abstract

We extend the concept of CONWIP control to a job shop setting, in which multiple products with distinct routings compete for the same set of resources. The problem is to determine the fixed overall WIP level and its allocation to product types (WIP mix) to meet a uniformly high customer service requirement for each product type. We formulate an optimization problem for an open queuing network model in which customer orders pull completed products from the system. Then, assuming heavy demand, we derive a throughput target for each product type in a closed queuing network and provide a simple heuristic to find a minimum total WIP and WIP mix that will achieve an operating throughput close to this target. In numerical examples, the WIP mix suggested by this approach achieves the customer service requirement with a relatively low total WIP.

1. Introduction

The development of CONstant Work In Process (CONWIP) control [1,2] has highlighted the benefits of control policies that pull work into the facility in response to demand while limiting inventory. The theoretical justification for this approach has been provided primarily for serial and assembly systems with simple, standardized routings. However, the benefits of capping inventory can be expected to accrue in job shops as well. Limiting the amount of WIP reduces storage, finance and record keeping costs, allows the quick identification of quality problems, and permits a rapid response to machine breakdowns, material shortages or worker unavailability [3, pp. 318-323]. Pull policies simplify scheduling and allow customer or downstream internal demands to dictate directly what is produced and when.

One question that has not been answered adequately is how to determine the amount of inventory that is allowed. Only simple heuristics and empirical approaches have been suggested [3]. In a job shop that produces many different products, this is an even more difficult question, as the allowable WIP must be divided among the various product types. A rule of thumb developed for a single item produced in a serial line is not likely to be applicable to each of the products in a job shop since it ignores the interactions among products. Empirical approaches are unworkable when the number of decision variables is large.

We assume that the goals of the job shop's managers are to meet demands for a variety of products equitably; that is, so that orders for some products are not favored at the expense of others and all orders are satisfied as quickly as possible. Confronted with sporadic demand, one extreme approach would be to produce according to a demand forecast and hold finished goods

inventory (FGI) in order to satisfy orders immediately. The other extreme would be an absolute make to order system in which a new job is released to the system in response to each arriving order. We propose a compromise between these approaches in which the total inventory (WIP plus FGI) of each product is held constant. This approach avoids the very high FGI costs of the make to stock system while reducing the long lead times that are possible in the make to order system. In our control system, the total inventory of each product is held constant and new jobs are pulled into the system as old jobs are completed and orders arrive. The pull policy ensures that production rates for individual products will follow the orders received. A drop in demand for one particular product effectively halts production for that product, while more frequent orders for another product are met by more frequent releases into the job shop.

Spearman [4] examined the customer service provided by a single product serial CONWIP line, as measured by the mean waiting times of orders, and demonstrated its superiority over a kanban system with equivalent inventory. He noted that customer service improves with increasing WIP, but did not address how to find an appropriate WIP level to meet a desired customer service criterion. Recently, Herer and Masin [5] have addressed the allocation of WIP to different products in a serial CONWIP system. They proposed a nonlinear integer program to determine the order in which products are released to the line in order to minimize the total inventory, backorder and overtime costs. The deterministic processing times and sequence dependent setups were approximated by a single exponential distribution for the processing time of all jobs on a machine. Choobineh and Sowrirajan [6] proposed an extension of CONWIP to a job shop with a limit on total inventory but a control policy that allowed a completed part of one type to be replaced by a raw part of a different type. For each product type, an upper limit on

the number of jobs in process was determined according to work content and the throughput was estimated by simulation.

In this paper, we focus on the proportion of orders that wait to be fulfilled and observe the orders' mean waiting time as a secondary measure of performance. The problem is to determine a card count for each product type so that the probability that an order waits to be filled is uniformly low over product types. Each product type may have a distinct routing through the processing stations and its own processing time distribution on each station it visits. Section 2 details the models, notation and the formulation of a customer service optimization problem. An assumption of heavy demand leads to the formulation of a combinatorial problem to set inventory levels according to a throughput criterion. In Section 3 we present a two stage procedure to solve the throughput problem and then apply its solution to the customer service problem. In Section 4 we provide the throughput and customer service results for numerical examples, and we conclude with Section 5.

2. Problem Definition

Consider a job shop in which R types of products are processed at M stations. A product type is defined by a set of alternative routings through the shop and a processing time distribution at each station it visits. Orders for each product type arrive according to some externally determined process. Our objective is to design a pull control system that will meet these demands with a high service level, that is, a uniformly low proportion of customer orders for each product type that must wait for fulfillment.

The CONWIP control mechanism can be modeled as a queuing network with two types of synchronization stations. Each product type has an input synchronization station for raw parts and an output synchronization station for finished goods. At the input station, a queue of raw parts is synchronized with a queue of authorization cards for that product type. When both a raw part and an authorization card are available, the card is attached to the part and the part is released to the manufacturing system. The output station synchronizes a customer order queue with a queue for finished products, so that as soon as one of each is available, the product is released to fill the order. Upon a finished product's release, its authorization card is detached and immediately returned to that product type's input station. Figure 1 shows an open queuing network with four synchronization stations for a CONWIP controlled job shop with two product types.

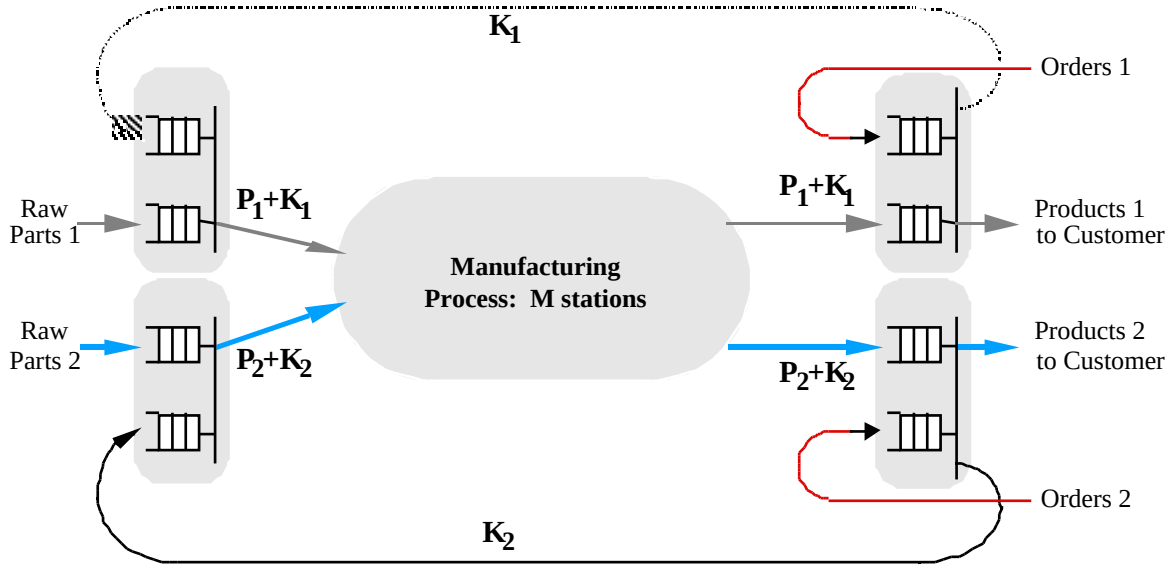

Figure 1. Hybrid open/closed queuing network model of a two product CONWIP system. At input synchronization stations (left), work authorization cards (K_r) are attached to raw parts (P_r) before they are released to the manufacturing process. Output synchronization stations at right model the way customer orders pull completed products from the system and release their work

authorizations.

The queuing network model shown in Figure 1 can be viewed as a hybrid open/closed queuing network. For each product type three kinds of entities (products, authorization cards, and orders) move through the network. With respect to products, the network is open since parts arrive at the input stations and finished products depart from the output stations. Likewise, the network is open with respect to orders since these entities arrive and depart from the output stations. However, the network is closed with respect to the authorization cards since a fixed number of cards for each type circulate among input, processing and output stations. The number of authorization cards limits the number of products of each type that can be in process.

We assume throughout this paper that there is ample raw material available for each product type. It follows that, since an authorization card will never wait for raw material at the input synchronization station, the input synchronization stations can be eliminated from the model. Figure 2 shows the model of Figure 1 under the assumption of infinite raw material. For this model we wish to determine the total number of authorization cards and its allocation to product types in order to achieve a uniformly high service level as measured by the proportion of the arriving orders that are filled immediately.

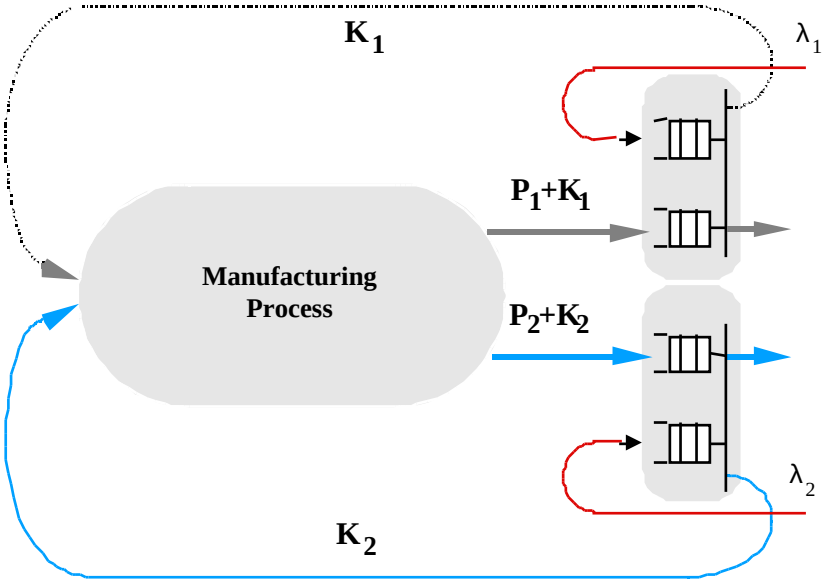

Figure 2. The two product CONWIP system of Figure 1 with infinite raw parts available. A recycled work authorization is attached to a raw part without delay.

Define the following notation for the system parameters:

$r, s = 1, \ldots, R$: Indices for product types.

$i, j = 1, \ldots, M$: Indices for stations.

λ_r : Arrival rate of orders for type r products.

$\alpha_r = \lambda_r / \sum_{s=1}^{R} \lambda_s$: Proportion of all orders over the planning horizon that are for type r.

$\alpha = (\alpha_1, \ldots \alpha_R)$: Product mix vector.

$R(i)$: Set of product types that visit station i.

T_i^r : Mean service time for type r products at station i (for $r \quad R(i)$).

p_{ij}^r : Probability that a type r product leaving station i goes to station j.

V_i^r : Mean number of times a type r product visits station i (for $r \quad R(i)$).

$\tau_r = \sum_{i \mid r \in R(i)} V_i^r T_i^r$: Work content of a type *r* product.

Note that, as the routing of products of each type *r* is assumed to be probabilistic, the mean visit rates, V_i^r, can be derived as one solution of the following classical system of equations [7, 8]:

$$V_i^r = \sum_{j=1}^{M} V_j^r p_{ji}^r \quad \text{for } i = 1,\ldots, M .$$

Clearly, $V_i^r = 0$ for any $r \quad R(i)$.

We wish to determine:

K_r: Number of type *r* authorization cards.

$N = \sum_{r=1}^{R} K_r$: Total number of authorization cards.

$\gamma_r = K_r / N$: Proportion of authorization cards dedicated to type *r*.

$\gamma = (\gamma_1,\ldots,\gamma_R)$: WIP mix vector.

The performance of the system is constrained by a specified service level. Define:

Ω : Specified service level, i.e., the proportion of orders to be satisfied at once.

$\omega_r(K_1,\ldots,K_R,\lambda_1,\ldots,\lambda_R)$: Proportion of orders for product type *r* that are not immediately satisfied.

$W_r(K_1,\ldots,K_R,\lambda_1,\ldots,\lambda_R)$: Mean waiting time for an order for product type *r*.

The performance measures ω_r and W_r, *r* = 1, …, *R*, are observed at the output synchronization stations, where customer orders interface with the closed network of authorization cards. Therefore, these quantities depend not only on the design parameters $K_1,\ldots,K_R$, but also on the exogenous parameters $\lambda_1,\ldots,\lambda_R$. For a given product set with known manufacturing requirements and a given service level the problem of determining the optimal number of

authorization cards can be stated as problem (**P**).

$$
(\mathbf{P}) \quad \begin{array}{lll} \text{Minimize} & N & \\ \text{s.t.} & K_1 + K_2 + \ldots + K_R & = N \\ & \omega_r(K_1,\ldots,K_R,\lambda_1,\ldots,\lambda_R) & \leq 1-\Omega, \quad r \\ & K_1,\ldots,K_R \geq 0, \text{ integer.} & \end{array}
$$

This formulation, which constrains the card counts to provide a *uniformly* high level of service, is designed to satisfy orders for different product types equitably. For a given set of parameters $(K_1,\ldots,K_R,\lambda_1,\ldots,\lambda_R)$, the approximation technique described in [7,8] and briefly highlighted in the Appendix can be used to obtain the values of ω_r, r = 1, …, R. It also predicts the mean waiting times, W_r, r = 1, …, R, which will be observed as a secondary measure of performance.

Since the impact of a control policy on the system's productivity becomes more pronounced under high demand, when system resources are very highly utilized, consideration of this condition is very important when designing a job shop control policy. If the demand rate for each product type is very high, then the probability of observing an empty order queue at an output synchronization station is very low. Therefore, under this condition we can assume that the authorization card attached to a part is recycled immediately after the part completes processing. This assumption eliminates the output synchronization stations from the queuing network model. The resulting multiproduct closed queuing network is depicted in Figure 3.

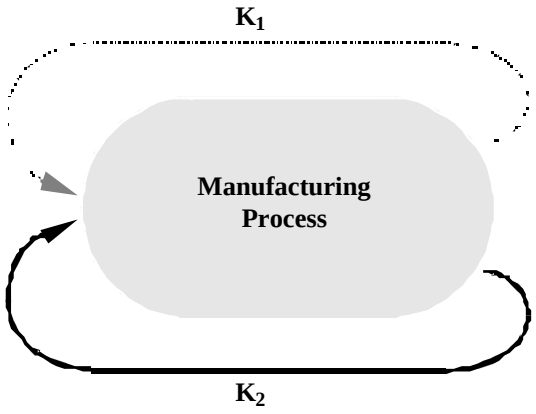

Figure 3. The two product CONWIP system of Figure 2 under heavy demand. A completed product's work authorization is recycled immediately upon completion of processing. The work authorization cards become the customers in a closed queuing network.

For several reasons, the model in Figure 3 is better suited for optimizing card counts than that in Figure 2. First, its performance depends solely on the card counts $K_1,\ldots,K_R$ (plus of course on all the internal system parameters), but not on external orders. In Figure 2, products are synchronized with external orders that arrive according to rates $\lambda_1,\ldots,\lambda_R$. As a result of the synchronization mechanisms, the throughputs of each product type will exactly equal the demand rates $\lambda_1,\ldots,\lambda_R$. In contrast, since the model in Figure 3 does not have any synchronization mechanisms, the demand rates do not control the throughput.

Secondly, the model in Figure 3, a pure closed queuing network, can be analyzed more easily by existing techniques. The cards circulating through the system can be seen as the customers in the closed queuing network, who are not allowed to change class. Note, however, that it need not have a product-form solution, as we have made no assumptions on the service time distributions of each type of product at each station, nor on the queue disciplines. The same approximate technique used to analyze the model in Figure 2 and highlighted in the Appendix will

also be used to analyze the model in Figure 3 for given values of $K_1,\ldots,K_R$ and obtain its performance measures of interest:

$\theta_r(K_1,...,K_R)$: System throughput of type *r* products.

$\Theta(K_1,...,K_R) = \sum_{r=1}^{R} \theta_r(K_1,...,K_R)$: Total system throughput.

The third and most important advantage of the closed model is that we can derive a fixed throughput target to facilitate the design of a card allocation heuristic. Keeping in mind the equitable service constraint in problem (**P**), it seems intuitively clear that balancing the system throughputs θ_r according to the product mix, so that $\theta_r \quad \alpha_r\Theta, r = 1,\ldots,R$, should satisfy this constraint with a minimum of inventory. An upper bound on the throughput constrained to follow the product mix under heavy traffic can be computed as follows. (We emphasize that this is only a theoretical value, as an infinite number of cards would be required to reach it exactly. In the derivation of this bound, we suppress the dependence on the parameters $K_1,\ldots,K_R$ in the notation.)

Assume $\theta_r = \alpha_r\Theta$ for $r = 1,\ldots,R$. For each station *i*, the throughput of type *r* products for the station is given by $\theta_i^r = V_i^r\theta_r = \alpha_r V_i^r\Theta$. Let $q_i^r \equiv \theta_i^r / \sum_{s\ R(i)} \theta_i^s = \alpha_r V_i^r / \sum_{s\ R(i)} \alpha_s V_i^s$ represent the proportion of all products processed at station *i* that are of type *r*. We can compute the maximum throughput of station *i* by assuming it is always busy, so that as soon as a product completes processing at the station, a new product (of type *r* with probability q_i^r) begins processing. Then the mean time between successive service completions at station *i* is given by $T_i = \sum_{r\ R(i)} q_i^r T_i^r$. The quantity $\theta_i^{\max} \equiv 1/T_i$ represents the maximum total throughput

at station *i* under the product mix constraint. Now the maximum system throughput can be obtained by adjusting $\theta_i^{\max}$ for the proportion of all products that visit station *i* as $\min_{i=1,\ldots,M}\left\{\theta_i^{\max} / \sum_{r\ R(i)} \alpha_r V_i^r\right\}$. Then, substituting for q_i^r in $\theta_i^{\max}$ and simplifying, we obtain the expression for the maximum throughput subject to the exact product mix:

$$\Theta_{\max} = \min_{i=1,\ldots,M} \frac{1}{\sum_{r\ R(i)} \alpha_r V_i^r T_i^r}.$$

Now we can state the problem to be solved for the closed queuing network model in Figure 3. Given a proportion $\beta < 1$ of the theoretical throughput $\Theta_{\max}$, let $\theta_r^* = \beta\alpha_r\Theta_{\max}$ be the operating throughput level for type *r*. Our goal is to determine card counts that enable the throughputs θ_r of type *r* products in the closed network to achieve the targets θ_r^*:

$$(\mathbf{Q}) \quad \begin{array}{lll} \text{Minimize} & N & \\ \text{s.t.} & K_1 + K_2 + \ldots + K_R & = N \\ & \theta_r(K_1,\ldots,K_R) & \geq \theta_r^*, \quad r \\ & K_1,\ldots,K_R \geq 0, \text{ integer.} & \end{array}$$

The solution to (**Q**) should provide comparatively good results in problem (**P**) for any set of demand rates following the proportions $\alpha_1,\ldots,\alpha_R$. We formalize a procedure to use a solution to (**Q**) for solving (**P**) in Section 3 and illustrate the accuracy of this procedure by numerical examples in Section 4.

Before attacking either problem, we can point out that the maximum throughput defines the feasible values of the demand rates. For problem (**P**) to have a solution, the demand rate λ_r must not exceed $\alpha_r\Theta_{\max}$ for any type *r*.

3. Solution Procedure

The proposed solution procedure for (**P**) consists of two stages. The first stage is a heuristic for solving problem (**Q**). In the second stage, the solution for (**Q**) is utilized for solving (**P**).

The goal in problem (**Q**) is to find a minimum WIP level (N) and WIP mix to attain the balanced throughput threshold. For a fixed N, the allocation of cards to product types is similar to a nonlinear resource allocation with integer variables. Since the throughput is not separable in the decision variables $K_1,\ldots,K_R$, there are no efficient methods for finding suitable values for them. As there are $\binom{N-1}{R-1}$ allocations of N cards to R types allotting each type at least one card, it is impractical to enumerate the possible solutions, especially for systems with many product types. Indeed, in view of the complexity of evaluating the queuing network's performance, we wish to examine as few solutions as possible. The card dealing heuristic is designed to efficiently search for the smallest N with an effective allocation.

The maximum balanced throughput, $\Theta_{\max}$, represents the target for the card allocation. The idea of the card dealing heuristic is, in an analogy with dealing out playing cards, to allocate work authorization cards one at a time. First the operating proportion, β, of the maximum throughput is specified. Starting from an initial allocation of a small number of cards for each product type, each successive card is given on a trial basis to each of the product types. The type that makes the best use of the additional card, by moving the *total* system throughput closest to the target (according to Euclidean distance), is allowed to keep the card. The process

continues until each type's throughput attains or exceeds its requirement, $\theta_r^* = \beta\alpha_r\Theta_{\max}$. The formal statement of the algorithm is:

Card Dealing Heuristic

Step 1. Given $\beta < 1$, compute θ_r^*. Initialize K_r for $r = 1,\ldots,R$ to small values.

Evaluate $\theta_r(K_1,\ldots,K_R)$ for $r = 1,\ldots,R$.

Step 2. While $\theta_r(K_1,\ldots,K_R) < \theta_r^*$ for some r do

For r=1,…,R do

For s=1,…,R do

Evaluate $\theta_s^{r+} = \theta_s(K_1,\ldots,K_r+1,\ldots,K_R)$

Choose type r to $\min_r \sqrt{\sum_s \left(\theta_s^{r+} - \alpha_s\Theta_{\max}\right)^2}$

Set $K_r \leftarrow K_r + 1$. Return to **Step 2**.

The throughputs are evaluated by the approximate performance evaluation routine highlighted in the Appendix.

Given a service level, Ω, it remains to determine how to choose an appropriate value for the operating throughput proportion, β, and then apply the solution for (**Q**) to solving (**P**). That is, we wish to achieve acceptable service with a minimum total inventory in the model with output synchronization stations. Through experience with numerical examples, we have found that the card dealing heuristic with a value of β near 0.95 finds an effective WIP mix. The second stage of our procedure for solving problem (**P**) sets K_r in proportion to the card dealing WIP mix and searches for the smallest N for which the customer service constraint is satisfied. The

combined procedure for solving (**P**) consists of the following steps.

1. Given order arrival rates, $\lambda_1,\ldots,\lambda_R$, calculate the product mix vector, $\alpha = (\alpha_1,\ldots,\alpha_R)$. Compute $\Theta_{\max}$ and verify that $\lambda_r \le \alpha_r \Theta_{\max}$, r.

2. Set β=0.95. Use the **Card Dealing Heuristic** to find $K_1',\ldots,K_R'$ and N'. For each product type r, calculate $\gamma_r' = K_r' / N'$.

3. If $\omega_r(K_1',\ldots,K_R',\lambda_1,\ldots,\lambda_R) \le 1-\Omega$, r, then search by trial and error for the smallest $N \le N'$ such that $N = \sum_{r=1}^{R} [\gamma_r' N]$ and $\omega_r([\gamma_1' N],\ldots,[\gamma_R' N],\lambda_1,\ldots,\lambda_R) \le 1-\Omega$, where $[\,]$ denotes rounding to the nearest integer. Similarly, if $\omega_r(K_1',\ldots,K_R',\lambda_1,\ldots,\lambda_R) > 1-\Omega$ for some r, then search for the smallest $N \ge N'$ as above.

The waiting probabilities are evaluated by the approximate performance evaluation routine highlighted in the Appendix.

4. Numerical Examples

We demonstrate the performance of our solution procedure through two examples. The first example represents a system with two product types. This system allows a comparison of the card dealing allocation's performance with any other alternative in both problems (**P**) and (**Q**). Tests with this system confirm the accuracy of the card dealing heuristic for (**Q**) and support our procedure for applying its WIP mix to solve (**P**) for some specified value of Ω. The

use of this combined procedure is then tested in the second example, which represents a larger and more complex system.

To ensure feasibility in each example, we initially specify the product mix vector, α, compute $\Theta_{\max}$, and then set the demand rates $\lambda_r < \alpha_r \Theta_{\max}$. In practice, of course, the demand rates would be observed initially and the product mix vector derived from them.

Example 1. Consider the system shown in Figure 4, with two product types (R=2) and six stations (M=6). Service times are exponentially distributed with means given in the boxes that represent the stations and routing probabilities (where less than one) shown on the arrows. The allocations found by the card dealing heuristic for different product mix vectors (α) and operating throughputs (β) are given in Table 1. The achieved throughput, Θ_a, is the system throughput attained when the card dealing algorithm stops. The WIP mix proportions, $\gamma_r = K_r / N$, are also shown.

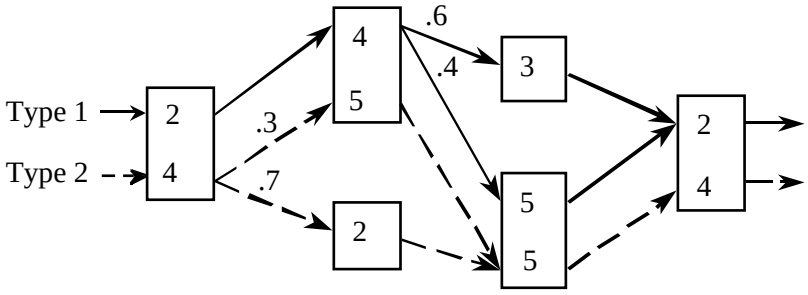

Figure 4. Product type flow schematic for Example 1. A box represents a processing station. Numbers in the boxes represent mean processing times of product types. Numbers on arrows represent routing probabilities, which equal 1 if not specified.

Table 1. Allocation results for Example 1 obtained from the card dealing heuristic. The last line of the table shows that for $\alpha = (0.7, 0.3)$, the allocation that satisfies the stopping rule for β=0.990 also causes the algorithm to stop if β=0.995.

α	Θ_{max}	β	Θ_a	N	K_1	K_2	γ_1	γ_2
(0.4, 0.6)	0.263	.950	0.253	15	5	10	.33	.67
		.990	0.261	23	7	16	.30	.70
		.995	0.263	27	8	19	.30	.70
(0.5, 0.5)	0.286	.950	0.277	18	8	10	.44	.56
		.990	0.284	30	12	18	.40	.60
		.995	0.285	33	13	20	.39	.61
(0.7, 0.3)	0.308	.950	0.299	18	13	5	.72	.28
		.990	0.305	25	19	6	.76	.24
		.995	0.305	25	19	6	.76	.24

For just two product types, it is possible to compare the algorithm's performance with an enumeration of the feasible allocations of cards. Figure 5 shows, for each of the three product mix vectors, the distance from Θ_{max} obtained by each feasible allocation of 18 cards to the first product type (with the remainder given to the second product type). The allocation of cards with the shortest distance from Θ_{max} is termed optimal. For example, for $\alpha = (0.5, 0.5)$, the optimal allocation of N=18 cards is K_1=8, K_2=10. In this example, for each product mix the card dealing heuristic does find the optimal allocation of N=18 cards between the two product types. This can be verified in Table 1 for $\alpha = (0.5, 0.5)$ and $\alpha = (0.7, 0.3)$ since the heuristic stops on N=18. In fact, an examination of the allocation trajectory for each product mix vector (shown in Figure 6 for $\alpha = (0.5, 0.5)$) reveals that the same statement holds for any $N \le 18$.

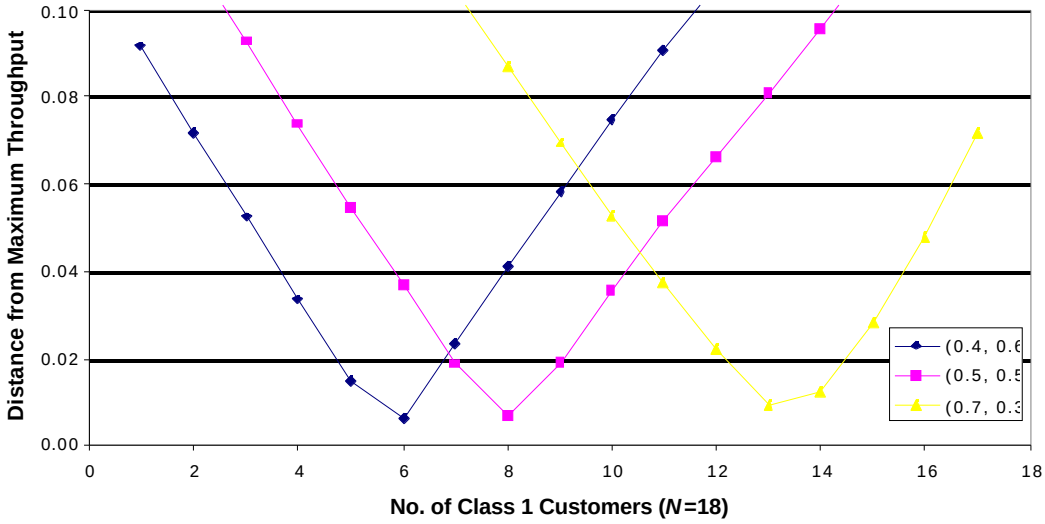

Figure 5. The distance from the maximum throughput obtained by each allocation of 18 cards to types 1 and 2 in Example 1. For each product mix vector, the minimum distance is obtained by the card dealing allocation.

The card dealing heuristic's trajectories from various starting points are shown in Figure 6 for α = (0.5, 0.5). The path from each initial allocation eventually joins the same path in the (K_1, K_2) plane. The trajectory plotted in the (θ_1, θ_2) plane shows how the product mix constrains the throughput. The total throughput could be increased substantially by deviating from the product mix vector. Also, each additional card provides a diminishing increase in the throughput. In Figure 7, the trajectories are shown in terms of the proportion of cards allocated to type 1 products (γ_1) as *N* increases. In this example, the allocations maintain an approximately stable WIP mix vector, which differs from the product mix vector.

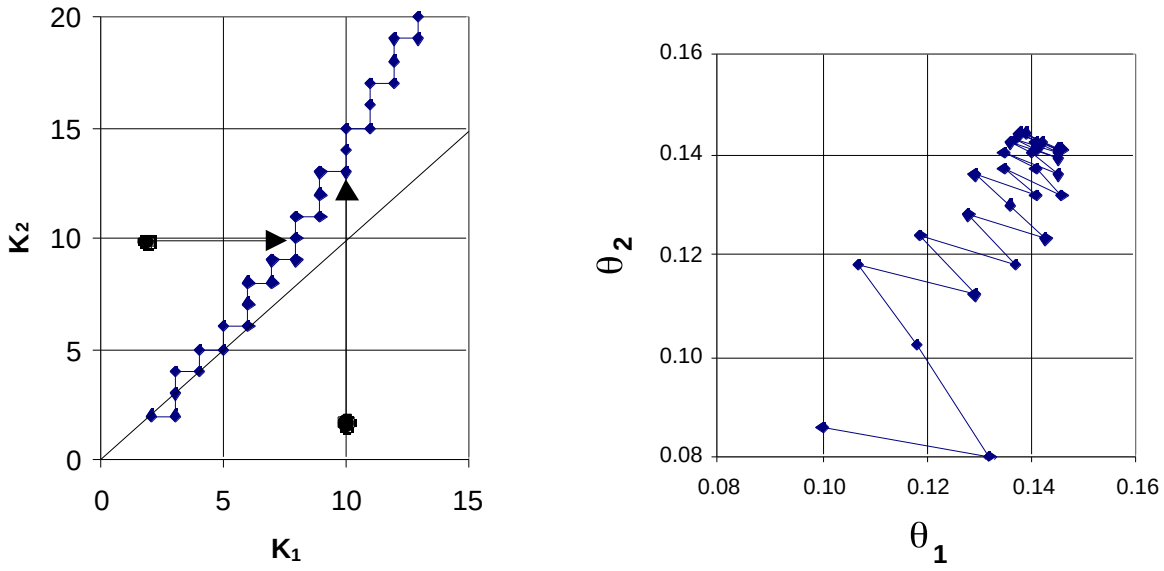

Figure 6. At left, the allocation trajectories from three different initial allocations followed by the card dealing heuristic for Example 1 with α=(0.5, 0.5). At right, the throughput trajectory from the initial allocation of (2, 2).

With two product types, it is also easy to compare the customer service performance of an allocation of a fixed N with neighboring allocations. Table 2 compares the customer service performance of the card dealing allocations with alternative allocations. Throughout, the demand rate for each product type was set as $\lambda_r = 0.85\alpha_r\Theta_{\max}$. For each product mix, the card dealing allocations that achieved 95% and 99% of $\Theta_{\max}$ (shown in boldface type) are compared with alternative allocations of the same *N*. For β=0.95, the most equitable customer service performance is achieved by the card dealing allocation since the values of ω_r and W_r are approximately equal across product types. The card dealing allocations found with β=0.99 also provide nearly the most equitable service. However, more equitable service with the larger values of *N* can be obtained by allocating cards according to the card dealing WIP mix, (γ_1', γ_2'),

obtained with β=0.95. The shaded allocations shown for β=0.99 are obtained under this strategy by setting $K_r = \lfloor \gamma_r' N \rfloor$. For example, for α = (0.4, 0.6), K_1=8 is obtained by rounding the product of $\gamma_1' = 5/15$ and N=23.

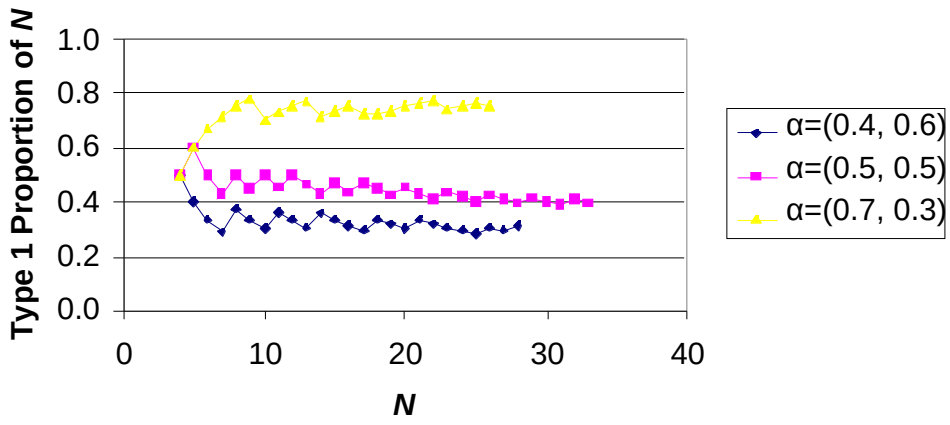

Figure 7. Proportions of cards dedicated to type 1 along allocation trajectories for Example 1.

The equitable waiting times achieved by applying (γ_1', γ_2') allow the service constraint in problem (**P**) to be satisfied with nearly minimal values of N. For Ω=0.9, total enumeration finds the optimal values of N in (**P**) for the three α vectors to be 25, 27, and 28, respectively. Applying our procedure, we obtain the respective values for N of 26, 27, and 31.

Table 2. Customer service comparisons for Example 1 for values of N corresponding to β=0.95 and 0.99.

β	α	N	K_1	K_2	ω_1	ω_2	W_1	W_2
0.95	(0.4, 0.6)	15	4	11	.649	.247	45.06	8.97
			5	**10**	**.406**	**.330**	**15.68**	**13.80**
			6	9	.241	.424	6.70	20.76

β	α	N	K_1	K_2	ω_1	ω_2	W_1	W_2
	(0.5, 0.5)	18	7	11	.444	.220	19.32	8.12
			8	**10**	**.306**	**.292**	**10.09**	**12.06**
			9	9	.203	.378	5.46	17.84
	(0.7, 0.3)	18	11	7	.436	.155	19.50	4.23
			12	6	.349	.252	13.43	8.08
			13	**5**	**.274**	**.397**	**9.31**	**16.42**
			14	4	.209	.611	6.33	40.40
0.99	(0.4, 0.6)	23	6	17	.292	.066	9.75	1.99
			7	**16**	**.183**	**.088**	**5.01**	**2.80**
			8	15	.112	.113	2.64	3.78
			9	14	.066	.144	1.38	5.00
	(0.5, 0.5)	30	11	19	.126	.034	3.27	1.05
			12	**18**	**.087**	**.045**	**2.08**	**1.42**
			13	17	.060	.058	1.32	1.89
			14	16	.040	.075	0.83	2.49
	(0.7, 0.3)	25	16	9	.170	.068	5.37	1.65
			17	8	.138	.109	4.18	2.87
			18	7	.111	.174	3.23	5.07
			19	**6**	**.088**	**.271**	**2.45**	**9.28**
			20	5	.067	.417	1.79	18.37

Example 2. The second example, shown in Figure 8, represents a much more complex system, modified from [6]. Five product types (R=5) compete for ten single server stations (M=10) with complex routings, including some possible rework for type 2 customers. Processing times are exponentially distributed, with total processing times in minutes for types one through five of (58.0, 66.8, 53.7, 84.4, 53.6), respectively. Assuming an equal product mix of α = (0.2, 0.2, 0.2, 0.2, 0.2), Θ_{max} is calculated to equal 0.07396 products per minute. Table 3 shows the allocations found by the card dealing heuristic for increasing values of β. Because the

total throughput does not exactly match the product mix, the achieved throughput, Θ_a, actually exceeds Θ_{max}.

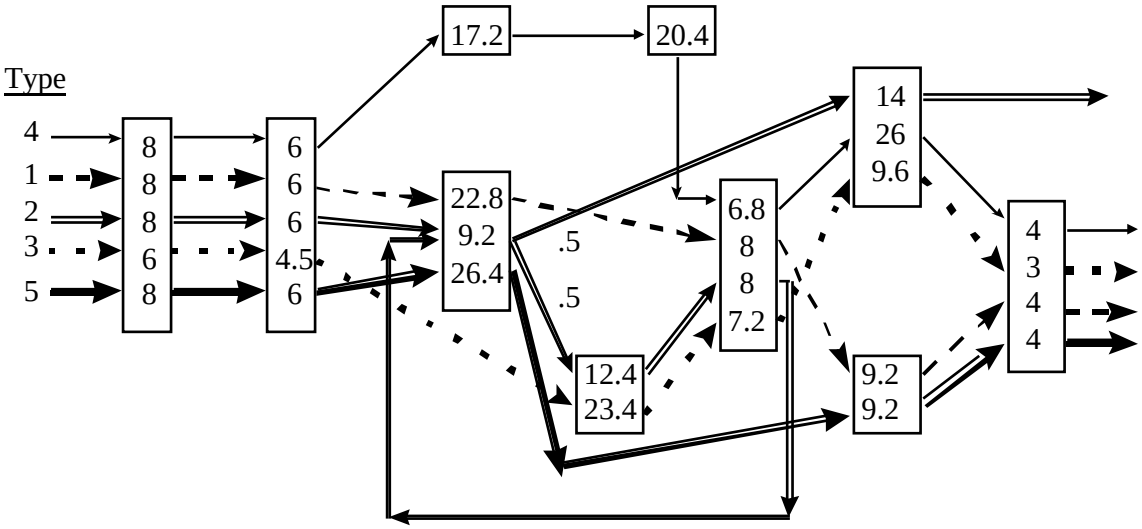

Figure 8. Product flow schematic for Example 2.

Table 3. Allocation results for Example 2 with equal product mix.

β	Θ_a	N	K_1	K_2	K_3	K_4	K_5	γ_1	γ_2	γ_3	γ_4	γ_5
.950	.07836	19	3	7	3	3	3	.158	.368	.158	.158	.158
.970	.07825	23	4	9	3	3	4	.174	.391	.130	.130	.174
.980	.07806	31	6	13	3	3	6	.194	.419	.097	.097	.194
.985	.07795	39	8	17	3	3	8	.205	.436	.077	.077	.205

The allocation trajectory is shown in Figure 9 in terms of the WIP mix. In contrast to Example 1, the mix does not stabilize. For $\beta > 0.95$, most of the algorithm's effort is spent trying to raise the throughput for type 2 above its threshold value, while types 3 and 4 rest well above theirs. For this reason, as in Example 1, using values of β very close to 1 results in allocations that provide less than optimal customer service results. Table 4 shows the customer service performance of the stopping card dealing allocations for increasing values of β, tested

with $\lambda_r = 0.9\alpha_r\Theta_{\max}$. The waiting times for orders for product types 3 and 4 actually degrade as $\beta \to 1$ since the system becomes congested and successive cards are allocated to the other product types.

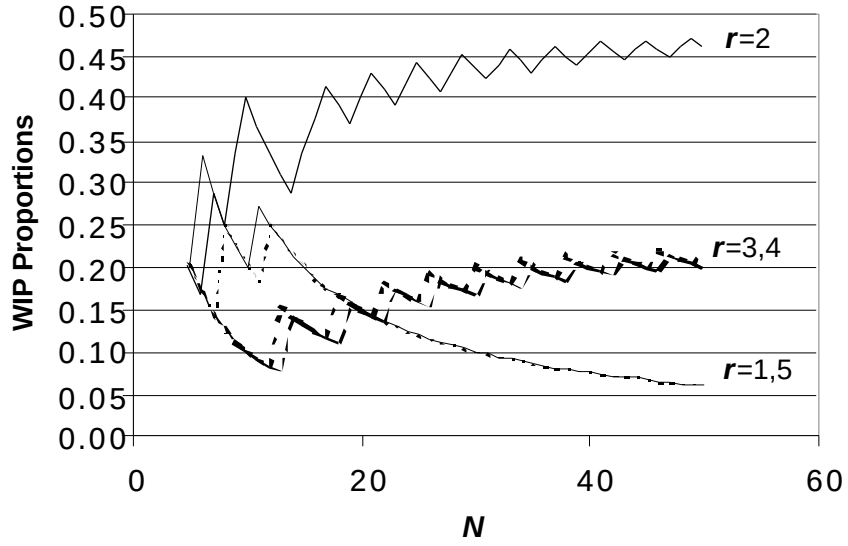

Figure 9. Proportion of cards for each type (WIP mix) in the allocation trajectory for Example 2 with an equal product mix.

Table 4. Customer service results provided by allocations where the card dealing heuristic stops. Mean waiting times, W_r, are given in minutes.

β	N	Max ω	ω_1	ω_2	ω_3	ω_4	ω_5	W_1	W_2	W_3	W_4	W_5
.95	19	.573	.399	.269	.573	.490	.356	63.9	66.4	128.8	90.0	50.5
.97	23	.577	.252	.155	.577	.492	.228	34.8	36.7	131.2	90.7	29.2
.98	31	.578	.112	.060	.578	.493	.102	14.9	14.6	132.2	91.1	13.0

However, proportionally scaling the WIP mix found using a moderate value of β yields good results in (**P**). For this example, it is not feasible to compare the card dealing allocations with all neighboring alternatives with the same N. Instead, we compare the customer service performance of the card dealing WIP mix with simple heuristics for allocating WIP. Table 5

shows the customer service results for randomly arriving orders using the same values for λ_r as in Table 4. The card counts according to the rule labeled CRD are obtained by rounding $\gamma_r' N, r = 1,\ldots,5$, where γ_r' is the WIP mix proportion found with β=0.95, to integer values. The work content (WRK) allocation sets the card count for type *r* in proportion to $\alpha_r \tau_r / \sum_{s=1}^{R} \alpha_s \tau_s$. It considers relative processing times, but no information about routing or the competition for processing stations. The simplest method (DEM) matches card counts to the product mix proportions, so that $K_r = \alpha_r N$.

Results are shown for all three allocation schemes using *N*=25 and *N*=40. From these common values of *N*, it is clear that the naïve allocations cause very long and frequent waits for orders for product 2 compared with the other products, but the card dealing allocations are much more equitable. (These two *N* values were chosen so that the DEM and WRK allocations rounded easily to integer values. For *N*=40, applying the card dealing WIP mix obtained with β=0.95 and rounding yields (6,15,6,6,6) for a total of 39. By trial and error, the most equitable customer service is obtained by allocating this "extra" card to type 1).

Finally, for each allocation rule Table 5 includes the smallest value of *N* that achieves a customer service requirement of Ω=0.8. By providing a more uniform level of service across product types, the card dealing WIP mix achieves this customer service goal with much less inventory (*N*=31 vs. *N*=50 or *N*=55). These results highlight the importance of taking into account queuing interactions, in addition to the product mix and mean processing times, when setting WIP levels.

Table 5. Customer service results obtained by alternative WIP mix heuristics with $\lambda_r \quad 0.9\alpha_r\Theta_{\max}$.

Rule	N	Max ω	ω_1	ω_2	ω_3	ω_4	ω_5	W_1	W_2	W_3	W_4	W_5
CRD	25	.279	.252	.161	.279	.245	.228	34.8	38.5	32.5	27.0	29.2
	31	.166	.166	.100	.132	.118	.151	22.1	24.2	11.8	10.1	19.0
	40	.113	.073	.035	.062	.055	.113	9.6	8.2	4.8	4.1	11.4
WRK	25	.680	.103	.680	.280	.023	.187	9.6	448.3	32.9	1.5	20.1
	40	.334	.045	.334	.029	.001	.041	4.4	139.1	2.0	0.1	4.0
	50	.168	.019	.168	.006	.000	.033	2.0	63.9	0.4	0.0	3.6
DEM	25	.699	.108	.699	.127	.115	.098	10.4	499.8	11.0	9.8	9.0
	40	.349	.024	.349	.012	.011	.023	2.3	155.2	0.8	0.7	2.1
	55	.190	.006	.190	.001	.001	.006	0.6	82.8	0.1	0.1	0.5

5. Conclusions

Our objective in this paper has been to extend CONWIP control to a job shop setting with an emphasis on customer service. Given this goal, inventory levels must be set for the whole system and each individual product to satisfy demands fairly. We addressed a secondary objective of minimizing inventory costs by designing our procedure to find the smallest effective inventory level. Formulating a solvable problem meant shifting the focus to throughput, but the allocation found by the throughput driven heuristic can be utilized to provide good customer service results in the examples tested. In both of these examples, a higher total throughput could have been achieved without the product mix constraint, but the resulting system design would greatly favor orders for some products at the expense of others.

We derived a throughput target that balances the shop's production according to a specified product mix. The card dealing heuristic allocates cards to product types on the basis of Euclidean distance from this target. The stopping criterion, however, simply requires that each

product type's throughput exceed some specified proportion of its target throughput. Our computational experience suggests that attempting to come extremely close to the throughput target is counterproductive from a customer service standpoint. Instead, equitable customer service can be provided by finding an allocation that achieves a throughput reasonably close to the target. From this allocation we derive a desirable WIP mix. A specific customer service goal can be achieved without excessive inventory by maintaining this WIP mix while varying the total WIP until a sufficient amount of inventory is identified.

Acknowledgements

This work was supported in part by the National Science Foundation through grants DMI-9701403 (Ryan) and DDM-9400146 (Choobineh). Additional support was provided by the Layman Fund at the University of Nebraska-Lincoln.

References

[1] Spearman, M., Woodruff, D. and Hopp, W. (1990) CONWIP: a pull alternative to kanban. *International Journal of Production Research* **28**, 879-894.

[2] Spearman, M. and Zazanis, M. (1992) Push and pull production systems: issues and comparisons. *Operations Research* **40**, 521-532.

[3] Hopp, W. and Spearman, M. (1996) *Factory Physics*, Irwin, Chicago.

[4] Spearman, M. (1992) Customer service in pull production systems. *Operations Research* **40**, 948-958.

[5] Herer, Y. and Masin, M. (1997) Mathematical programming formulation of CONWIP based production lines; and relationships to MRP. *International Journal of Production Research* **35**, 1067-1076.

[6] Choobineh, F. and Sowrirajan, S. (1996) Capacitated – constant work in process (C-CONWIP): a job shop control system. Technical Report, University of Nebraska-Lincoln, Lincoln, NE 68588-0518.

[7] Baynat, B. and Dallery, Y. (1996) A product-form approximation method for general closed queueing networks with several classes of customers. *Performance Evaluation* **24**, 165-188.

[8] Baynat, B. and Dallery, Y. (1993) A unified view of product-form approximation techniques for general closed queueing network. *Performance Evaluation* **18**, 205-224.

[9] Baynat, B. and Dallery, Y. (1993) Approximate techniques for general closed queueing networks with subnetworks having population constraints. *European Journal of*

Operational Research **69**, 250-264.

Appendix

The method developed in [7, 8] approximates the performance of each class in a multi-class queuing network by the performance of a single-class product-form network. Let K_r be the population of class r customers and $S(r)$ be the set of indexes of the stations that they visit, $r = 1,...,R$. A single-class network for class r will have $|S(r)|$ load-dependent exponential service stations. Let $\mu_{ri}(n_{ri})$, $n_{ri} = 1,...,K_r$, denote the load-dependent service rates of station i, $i \in S(r)$, for the r^{th} single-class network. The visit rates in the r^{th} single-class network, V_{ri}, $i \in S(r)$, are identical to those of class r customers in the original network. Let $\mathbf{n}_r$ be the state-vector pertaining to the r^{th} single-class network, i.e., $\mathbf{n}_r = [n_{ri}]$, $_{i \in S(r)}$. Since each single-class network is a Gordon-Newell network, the steady-state probabilities $P(\mathbf{n}_r)$ for the r^{th} network, $r = 1,...,R$, have the following product-form solution:

$$P(\mathbf{n}_r) = \frac{1}{G_r(K_r)} \prod_{i \in S(r)} \left[\prod_{n=1}^{n_{ri}} \frac{V_{ri}}{\mu_{ri}(n)} \right],$$

where $G_r(K_r)$ is the normalization constant associated with the class r network.

Ideally, $\mu_{ri}(n_{ri})$ for the class r product-form network should equal the average flow of class r customers out of station i in the original multi-class network, given that n_{ri} class r customers are present at the station (regardless of the number of customers of the other classes simultaneously present at the station). However, the method approximates $\mu_{ri}(n_{ri})$ by analyzing each service station in isolation when its queue is fed by $|S(r)|$ external state-dependent processes

[7, 9]. An iterative fixed point procedure solves a set of equations for the $\prod_{r=1}^{R} |S(r)|^{K_r}$ unknown values of $\mu_{ri}(n_{ri})$. The number of iterations to achieve convergence is usually very reasonable.

Biographical Sketches

Sarah M. Ryan is an assistant professor of Industrial and Management Systems Engineering at the University of Nebraska-Lincoln, having taught previously in the Department of Industrial Engineering at the University of Pittsburgh. She holds a B.S. in systems engineering from The University of Virginia and M.S.E. and Ph.D. in industrial and operations engineering from The University of Michigan. She is a senior member of IIE and a member of INFORMS and has published in several journals including *IIE Transactions* and *Operations Research.*

Bruno Baynat is *Maître de Conférence* (Associate Professor) at the *University Pierre et Marie Curie.* He received the M.S. degree from the *Institut National Polytechnique de Grenoble* in 1988 and the Ph.D. degree from the *University Pierre et Marie Curie* in 1991. His research interests are in the development of approximation methods for single-class and multi-class closed queuing networks with applications to computer-communication networks and manufacturing systems.

F. Fred Choobineh is a professor of Industrial and Management Systems Engineering at the University of Nebraska-Lincoln. He received B.S.E.E., M.S.I.E. and Ph.D. degrees from Iowa State University. He is a licensed Professional Engineer in the State of Nebraska and serves on the State's Board of Engineers and Architects. His research interests are related to the design and control of manufacturing systems and use of approximate reasoning techniques such as Rough Sets and Evidence Theory in decision making. He is a member of IIE, IEEE, ASEE and INFORMS.

Figure 1.

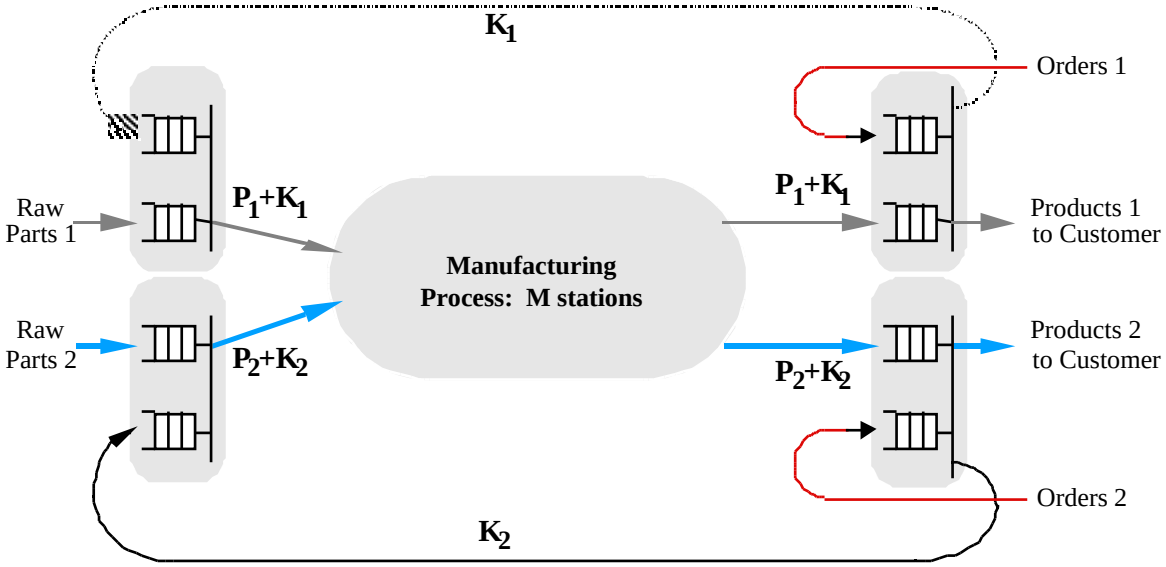
K1
Orders 1
Raw
Parts 1
P1+K1
P1+K1
Products 1
to Customer
Manufacturing
Process: M stations
Raw
Parts 2
P2+K2
P2+K2
Products 2
to Customer
Orders 2
K2

Figure 2.

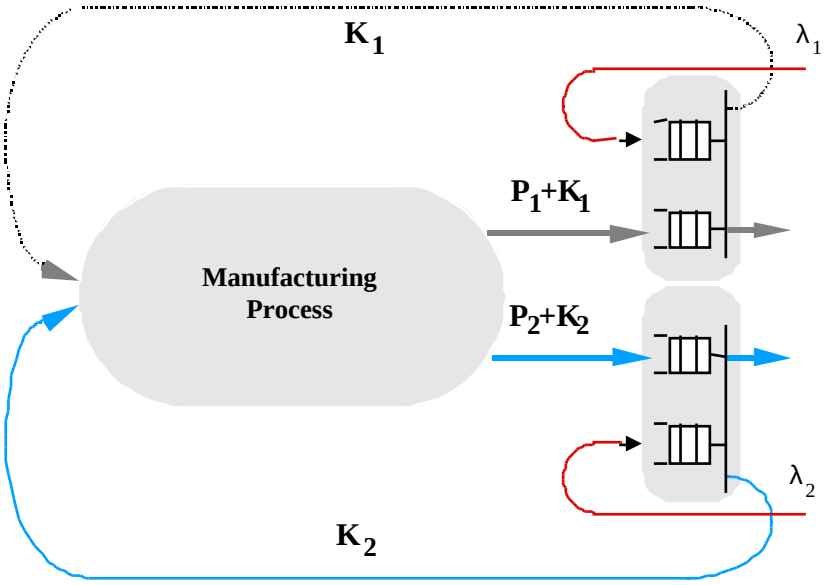

Figure 3.

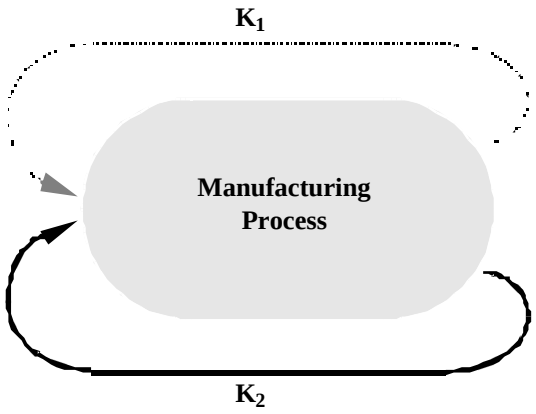

Figure 4.

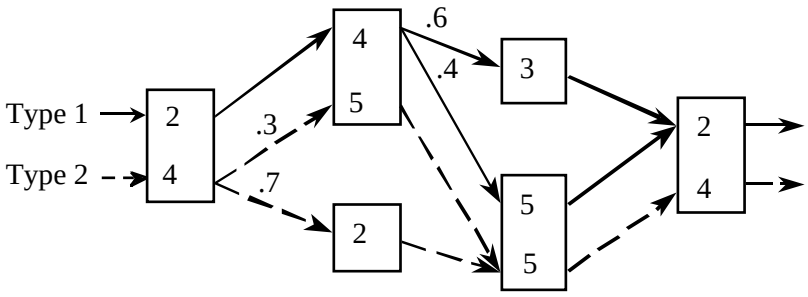

Figure 5.

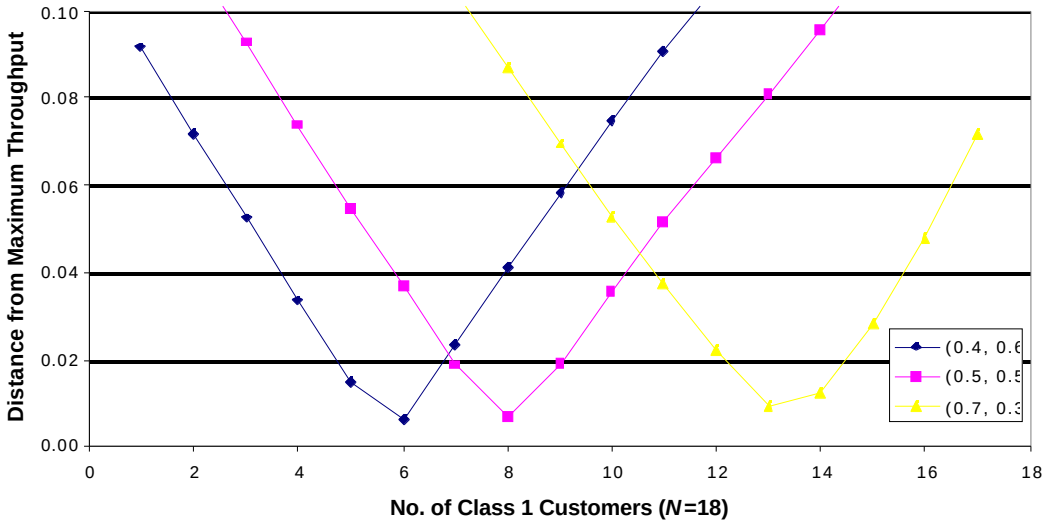
Distance from Maximum Throughput
0.10
0.08
0.06
0.04
0.02
0.00
0
2
4
6
8
10
12
14
16
18
No. of Class 1 Customers (N=18)
(0.4, 0.6
(0.5, 0.5
(0.7, 0.3

Figure 6.

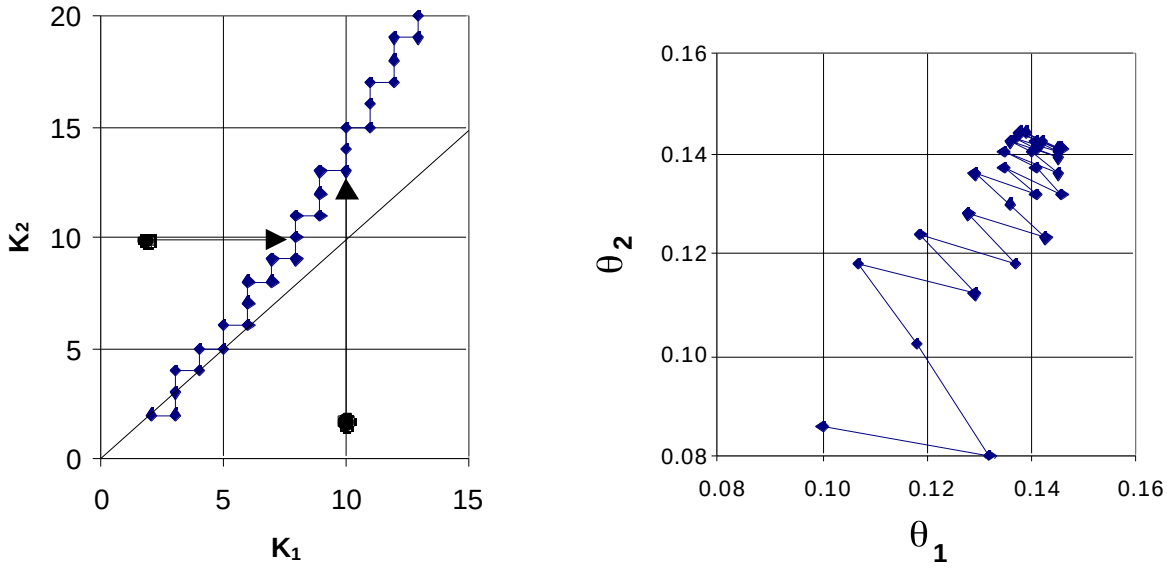
20
15
10
5
0
0
5
10
15
K2
K1
0.16
0.14
0.12
0.10
0.08
0.08
0.10
0.12
0.14
0.16
θ2
θ1

Figure 7.

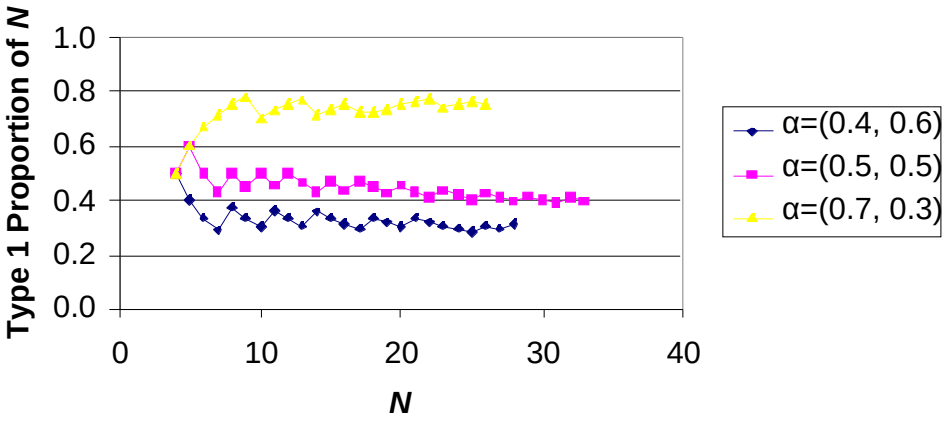
Type 1 Proportion of N
1.0
0.8
0.6
0.4
0.2
0.0
0
10
20
30
40
N
α=(0.4, 0.6)
α=(0.5, 0.5)
α=(0.7, 0.3)

Figure 8.

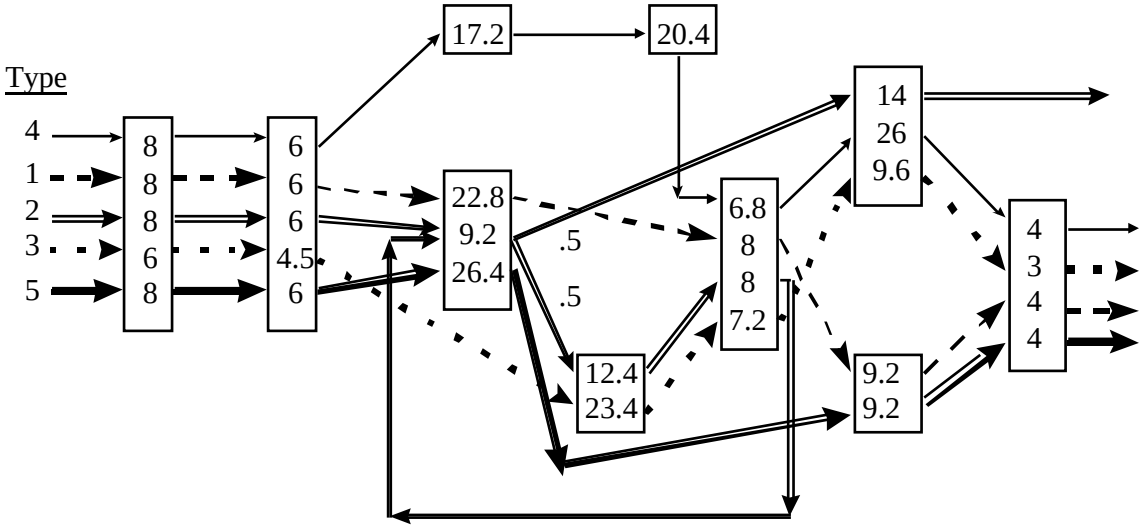
Type
4
1
2
3
5
8
8
8
6
8
6
6
6
4.5
6
17.2
20.4
22.8
9.2
26.4
.5
.5
12.4
23.4
6.8
8
8
7.2
14
26
9.6
9.2
9.2
4
3
4
4

Figure 9.

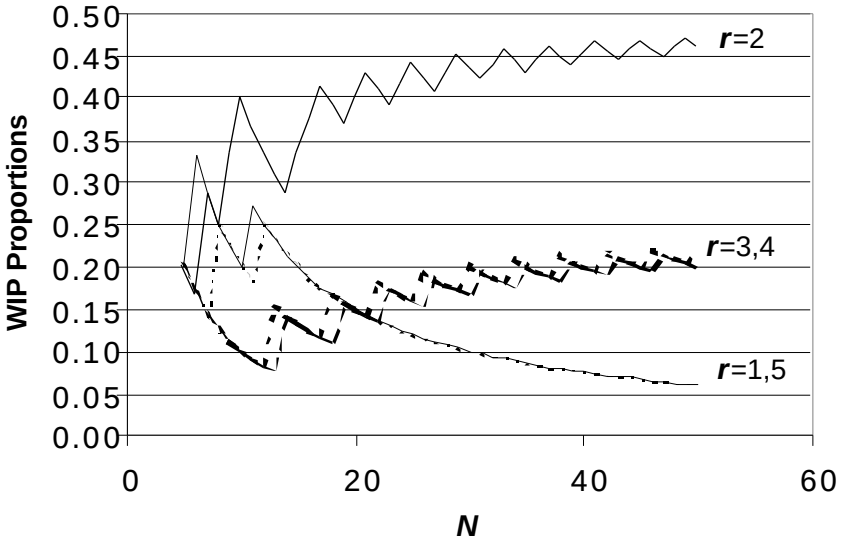
0.50
0.45
0.40
0.35
0.30
0.25
0.20
0.15
0.10
0.05
0.00
WIP Proportions
0
20
40
60
N
r=2
r=3,4
r=1,5

Figure Captions

Figure 1. Hybrid open/closed queuing network model of a two product CONWIP system. At input synchronization stations (left), work authorization cards (K_r) are attached to raw parts (P_r) before they are released to the manufacturing process. Output synchronization stations at right model the way customer orders pull completed products from the system and release their work authorizations.

Figure 2. The two product CONWIP system of Figure 1 with infinite raw parts available. A recycled work authorization is attached to a raw part without delay.

Figure 3. The two product CONWIP system of Figure 2 under heavy demand. A completed product's work authorization is recycled immediately upon completion of processing. The work authorization cards become the customers in a closed queuing network.

Figure 4. Product type flow schematic for Example 1. A box represents a processing station. Numbers in the boxes represent mean processing times of product types. Numbers on arrows represent routing probabilities, which equal 1 if not specified.

Figure 5. The distance from the maximum throughput obtained by each allocation of 18 cards to types 1 and 2 in Example 1. For each product mix vector, the minimum distance is obtained by the card dealing allocation.

Figure 6. At left, the allocation trajectories from three different initial allocations followed by the card dealing heuristic for Example 1 with α=(0.5, 0.5). At right, the throughput trajectory from the initial allocation of (2, 2).

Figure 7. Proportions of cards dedicated to type 1 along allocation trajectories for Example 1.

Figure 8. Product flow schematic for Example 2.

Figure 9. Proportion of cards for each type (WIP mix) in the allocation trajectory for Example 2 with an equal product mix.