Performance Opportunity of Workforce Agility in Collaborative and Noncollaborative Work Systems

Mark P. Van Oyen, Esma G. Senturk-Gel, and Wallace J. Hopp

Department of Industrial Engineering and Management Sciences
Northwestern University
Evanston, IL 60208-3119

Corresponding Author: Mark P. Van Oyen
Ph: 847-491-7008, FAX: 847-491-8005
E-mail: vanoyen@iems.nwu.edu
URL: http://www.iems.nwu.edu/~vanoyen/

Running Headline: Opportunity of Workforce Agility

August 1, 1998

Performance Opportunity of Workforce Agility in Collaborative and Noncollaborative Work Systems [1]

Mark P. Van Oyen, Esma G. Senturk-Gel, and Wallace J. Hopp

Department of Industrial Engineering and Management Sciences

Northwestern University, Evanston, IL 60208-3119

Abstract

To gain insight into the potential logistical benefits of worker cross-training and agile workforce policies, we study simple models of flexible workers in serial production systems. The primary control issue is how to assign workers to jobs/stations over time. Under assumptions of complete worker flexibility and collaborative work, we prove that a simple *expedite policy* minimizes along each sample path the cycle time (delay) for each job. Therefore, the expedite policy also minimizes work in process and maximizes throughput along every sample path for any time t. We also consider the optimal static allocation of effort (workers) and use it to compute the performance improvement opportunity achievable using flexible workers. This enables us to examine the factors that make workforce agility a potentially attractive strategy. We extend our analysis to the noncollaborative work environment by presenting evidence for the effectiveness of a common policy we call the *pick-and-run* policy, but demonstrate by counterexample that it is not always optimal. Finally, we extend some of our insights from a push to a pull environment operating under a constant-WIP (CONWIP) protocol.

1 Introduction

Most production modeling research in the literature has focused on rigid systems in which labor is implicitly viewed as tied to specific tasks or workstations. This is not surprising, since virtually all manufacturing systems since the industrial revolution have been based on work standardization and division of labor. Furthermore, mathematical and conceptual models of production are relatively complex, even for the case of rigid production systems. For systems with agile workers that can be dynamically allocated to different tasks or stations, models quickly become far more complex (see Farrar [6] and Ahn, Duenyas and Zhang [1]).

In spite of the literature, however, intense global competition has wrought major changes in the workplace over the past two decades. In particular, the just-in-time movement has turned the focus away from individual workstation efficiency and toward speed and smoothness of work flow. To support these flow-oriented systems, a number of new organizational schemes have arisen in which workers are not tied to individual stations, but are cross-trained and empowered to move between tasks to follow the workload. In such "agile worksystems", the value of a worker is more dependent on how many skills he/she has than on how fast he/she can perform a specific task.

There are many ways to build workforce agility into a production system. Some examples we have observed in industry include the following. Elgin Digital Colorgraphics (EDCG), a division

[1] This material is based upon work supported by the National Science Foundation under Grants No. DMI-9322930, DMI-9522795, and DMI-9732868 to Northwestern University.

of R.R. Donnelley and Sons Co., allowed workers to follow jobs through almost an entire pre-media printing process, changing stations and performing each of the required operations. John Deere, in an agricultural equipment assembly plant, permitted workers to shift between fabrication, subassembly, and final assembly within their production cells as workloads dictate. American Steel Foundries, in a finishing operation for castings, enabled workers to dynamically shift tasks from one station to another to promote flow. IBM, in a printed circuit board plant, authorized one group of workers to switch jobs to replace another group during lunches to increase production at a bottleneck. These are but a few examples of what is becoming common practice in industry.

Modeling research can support agile workforce practices in two ways. First, descriptive models can improve our understanding of how such systems behave and what factors drive performance. Second, prescriptive models can help us determine what types of policies are effective in different production environments. Either type of model requires a basic framework for representing how workflow depends on both equipment and labor.

How an agile worksystem is modeled depends on the nature of the production environment. One important distinction is whether the work is collaborative (i.e., tasks that permit more than one worker to work on a job simultaneously) or noncollaborative (i.e., tasks restricted to one worker at a time). If work is collaborative, then an approach to modeling agile workers that shift between stations in a system is to focus on the problem of service rate control. Related work with controlled service rates, but with different objectives can be found in the work of Veatch and Wein [18] and Weber and Stidham [19].

In her dissertation on flexible work systems, Kim [9] studies optimal control policies in various queueing networks. Under certain conditions, she argues that all workers should collaborate on the same job. Kim's policy is essentially the same as the "pooling policy" studied by Mandelbaum and Reiman [12], who compare the the steady-state mean sojourn times of the single-server Jackson network and the M/PH/1 queue. They assume exponential interarrival and service times and examine the conditions under which pooling might be advantageous using an efficiency index. No claims are made for the optimality of any type of pooling policy.

The literature on noncollaborative work systems has been more extensive than that on collaborative work systems. There have been a number of studies on machine-operator interference in the context of queueing for multiple levels of resources: machines and operators such as repairmen that service multiple stations (see Suri, Sanders and Kamath [17]). Downey and Leonard [5] study an assembly line with flexible workers that move from a station that has become idle to an unoccupied station that has work available. They employ a heuristic rule of server movement and study 3, 6, and 10 station cases using simulation. They find the optimal buffer sizes and the optimal number of servers that minimize the cost per unit time, subject to certain system parameters.

There have been numerous studies of specific work-sharing schemes, in particular, the Toyota

Sewn Products Management System (TSS), or *bucket brigade* and its derivatives. These policies are defined by rules that tell each worker what to do next as a function of the system state. TSS has been employed by sewn products manufacturers in modules that are used in the finishing and assembly of cut parts into a subassembly or finished garment [3]. In its basic form, under the bucket brigade policy, each worker picks up a job and processes it at each station until he gets bumped by a downstream worker. Only a single worker is allowed at each station and it is usually assumed in the literature that the ordering of the workers is preserved, to create work zones in which workers spend the majority of their time. Bischak [3] uses simulation to compare the throughput of U-shaped flexible worker manufacturing modules with that of serial lines with one stationary worker per machine, with and without the use of buffers in the systems. In contrast to the other literature on TSS lines, Bartholdi and Eisenstein [2] consider deterministic processing times and heterogeneous workers, each with a certain speed at a given task. They show that although movement of the workers can be chaotic under some circumstances, a stable partition of work will eventually emerge if the workers are sequenced from slowest to fastest, independent of the stations at which they begin. Their main result states that if the worker velocities are constant and if workers are never blocked, then the system converges to a fixed point (each worker repeatedly executes the same interval of work content), and the production rate is the largest possible. Others who have examined bucket-brigade-like systems include Zavadlav, McClain and Thomas [21], McClain, Thomas and Schultz [14] and Iravani, Posner and Buzacott [8].

In this paper, we go beyond the analysis of specific policies and general queueing insights of previous research and develop explicit descriptive and prescriptive models of serial production systems with collaborative and noncollaborative tasks in make-to-stock and make-to-order environments. Our main focus from a descriptive standpoint is on characterizing the factors that affect the "opportunity" for workforce agility. We define opportunity as the difference in a performance measure (typically cycle time) under an optimal agile workforce policy and under an optimal fixed workforce policy. By determining which aspects of the system lead to large opportunity, we provide insight into what types of systems are attractive candidates for agile workforce systems. We also take a prescriptive modeling approach to analyzing the collaborative, make-to-order, serial production system, and demonstrate the optimality of a specific policy, which we term the *expedite policy*. This policy is applicable to some environments; in other environments, as we demonstrate, it can be used effectively in partial form. The reasons for the optimality of the expedite policy suggest related policies for other environments; we examine one such policy, the *pick-and-run* policy for the noncollaborative, make-to-order serial system. Finally, we analyze a CONstant Work In Process (CONWIP), make-to-stock environment with exponential processing times.

The remainder of the paper is organized as follows: Section 2 provides a general formulation of our problem. Sections 3 and 4 demonstrate the optimality and quantify benefits of the expedite

policy under specific conditions. Section 5 presents various arguments to support the efficiency of the pick-and-run policy, as well as a counterexample that shows its slight suboptimality. Section 6 characterizes optimal policies for a CONWIP production environment and shows that both expedite and pick-and-run policies can be highly effective in such systems.

2 Formulation of the Collaborative Case with Full Cross-training

We consider a tandem production system that can be modeled as a series network of queues; that is, jobs served at queue n proceed directly to queue $n+1$ and from station N they exit the system. Jobs arrive to the system according to a general (possibly time-varying and correlated) process under the following assumption: the cumulative number of arrivals up to and including time t, denoted by $A(t)$, is such that $A(t) \in \mathbb{N}$ for any $t \in \mathbb{R}^+$, where $\mathbb{N}(\mathbb{R}^+)$ denotes the naturals (nonnegative reals). This guarantees a finite system queue length for any finite time. Let $a(i)$ denote the random time of the arrival of the ith job, where we require $E\{a(i)\} < \infty$ for all $i \in \mathbb{N}$. Batch arrivals are permitted, but we assume that they all have distinct labels that dictate the arrival sequence. This is important, because we assume the existence of a set of job sequences that requires FCFS job completions at the last station.

We model the collaborative work environment with the following assumptions. (1) All workers are identical and can collaborate on the same job without interfering with each other. Furthermore, there exist ample machinery, tooling, etc. for all workers to be working at the same station simultaneously. At any time t, the total work effort applied in the system cannot exceed the maximum amount of labor units available: M. (2) The workforce is divisible among the N stations in arbitrary increments; therefore, we allow $M \in \mathbb{R}^+$. (3) Workers can move from one station to another with zero cost and zero time (as was done in the analysis of [2], [12], and [9]). (4) At station n the service requirements for successive jobs $i = 1, 2, 3, \ldots$ are given by an i.i.d. sequence of random variables $\{S_n(i) : i \in \mathbb{N}\}$ with the unit being man-hours. The cumulative distribution function (c.d.f.) of $S_n(\cdot)$, denoted by F_n, is a general strictly positive distribution with finite first and second moments. We let $\mu_n = (ES_n(i))^{-1}$ denote the mean rate of the underlying service requirement at station n. (5) Work effort at any station or job can be controlled continuously over time. (6) The time it takes to complete a job at any station is inversely proportional to the amount of effort allocated (i.e., workers can collaborate without loss of efficiency). Note that these assumptions, particularly (2), (3) and (6) are "best case" assumptions regarding workforce agility. Hence the performance of this idealized system represents an upper bound, because work may not be infinitely divisible or perfectly additive in realistic systems. We will relax the assumption of additive work effort and examine systems in which only one worker at a time can work on a job, in Section 5.

Although these assumptions may be idealistic for many real life production environments, the pre-media printing process in Elgin Digital Colorgraphics (EDCG) is an example of a system that

comes close to satisfying them. EDCG has about 100 workers organized in three teams, whose primary function is to receive photographs/images and text material, combine these elements into a publication such as a mail order catalog with proper color and layout, and then generate the computer files that enable the printing presses to actually manufacture the catalogs. Although the traditional approach in the printing sector is strict specialization of labor, EDCG has invested in ample computer workstations so that a worker is able to take ownership of a particular piece of work and bring it through most stages of processing without undue reliance on the productivity of other workers. At a macro level in which a job is defined as a whole mail order catalog, having each worker perform most of the operations on a subset of pages makes it possible for workers to efficiently collaborate on the same job.

Due to the generality of the arrival process and the service time distribution, an optimal policy must be justified within the context of G, the class of all nonanticipative policies based on the history of observed states and control actions through time t. This requires a detailed state space. The system state is denoted by the matrix $(q(t), p(t))^T$, where T indicates transpose. Let $q(t) = (q_1(t), q_2(t), q_3(t), \ldots)$ where $q_i(t) \in \{0, 1, 2, \ldots, N, N+1\}$ denotes the queue job i is in at time t, 0 signals that the job has not yet arrived, and $N+1$ signals job completion. Let $p(t) = (p_1(t), p_2(t), p_3(t), \ldots)$ where $p_i(t) \in \mathbb{R}^+$ denotes the amount of the underlying service requirement at station $q_i(t)$ that has been met on $[0, t]$ (i.e., the age of job i). From this state, one can generate $x(t) \stackrel{\Delta}{=} (x_1(t), x_2(t), \ldots x_N(t))$, the state vector of queue lengths (including any job in service) at time t via $x_n(t) = \sum_{i=1}^{\infty} \mathbb{1}\{q_i(t) = n\}$ with $\mathbb{1}\{\cdot\}$ denoting the indicator function.

A Markov control policy g maps $(q(t), p(t))^T$, to an effort allocation matrix $u(t) \stackrel{\Delta}{=} [u_{n,i}(t)] : n = 1, 2, \ldots, N$ and $i \in \mathbb{N}$. We require $u(t) \in \mathcal{C}$, the space of effort matrices with $\sum_{n=1}^{N} \sum_{i=1}^{\infty} u_{n,i}(t) = M, u_{n,i}(t) \geq 0 \ \forall i$. Note that our space of admissible policies G, allows a policy to be *preemptive*; that is, at any instant, some or all of the labor applied to job j prior to time t can be transferred to one or more other jobs, even though job j is not yet completed at time t. If at time t, $q_i(t) = n$ and $p_i(t) = 0$, then job i will be completed at station n at the random time $C_n^g(i) = \inf\{T \geq 0 : \int_t^T u_{n,i}^g(t)dt = S_n(i)\}$. If in addition we assume that $u_{n,i}^g(t) = \underline{u} \in \mathbb{R}^+$ for a time interval of at least $S_n(i)/\underline{u}$ along each sample path, the resulting mean of the *service time*, $\tilde{S}_n^g(i)$, at station n is

$$E\{\tilde{S}_n^g(i)\} = E\{S_n(i)\}/\underline{u} = \frac{1}{\underline{u}\mu_n}. \tag{2.1}$$

An outcome of this model is that the coefficient of variation (cv) for the service time with a fixed effort level $\underline{u}$ at station n is $(\sqrt{\text{var}(S_n)}/\underline{u})/(E\{S_n\}/\underline{u})$. This implies that the cv is unaffected by the effort allocation, which is intuitive. In the sequel we use the random variable $C_N^g(i)$ to denote the completion time of the ith job at the last station.

3 Optimality of the Expedite Policy

Having defined the problem, we now demonstrate that a simple policy is optimal with respect to job completion times along every sample path. To do so, we require the following intuitive result

that a nonidling policy is optimal in this pathwise sense.

Definition 1: *Policy $g \in G$ is a nonidling policy if $\sum_{i=1}^{\infty} u_{n,i}^g(t) = 0$ whenever $x_n(t) = 0$ and if $x_i(t) > 0$ for some node i and time t, then all M workers must be allocated to the nonempty stations.*

Lemma 1: *Within the class of preemptive policies, policies that idle at least one server for a positive amount of time (in expectation) are strictly suboptimal.*

Proof: We condition on the event that at time τ, an idling policy $g \in G$ idles a positive amount of worker effort for a random length of time, ψ such that $E\{\psi\} > 0$, even though job j is available for processing at time τ. We take ψ to be a random variable to allow for randomization in g as well as in the arrival and service events. It suffices to show that g can be improved by a policy $\tilde{g}$ that does not idle until some time strictly greater than τ. Fix the sample path realization as $\omega \in \Omega$. Let the function $I(t,\omega)$, $t \geq \tau$ describe the worker effort (as a function of time and sample path) that is idled by policy g on $[\tau, \infty]$ (and is available for application to job j). Recall that a policy must work on job j from time τ on so as to accumulate an amount of effort equal to the remaining (residual) life of job j, which is calculated as $R_j(t) \stackrel{\Delta}{=} -p_j(t) + \sum_{n=q_j(t)}^{N} S_n(j)$.

We construct a policy, $\tilde{g}$, to mimic g except that $\tilde{g}$ applies all of the idled worker effort to job j beginning at time τ and concluding when job j is completed. Effort allocations for all jobs other than j are identical under g and $\tilde{g}$. Once $\tilde{g}$ completes job j, it must idle workers whenever g previously either served job j or idled workers. This causes the two processes to be coupled at time $C_N^g(j,\omega)$. With respect to g, policy $\tilde{g}$ completes job j earlier than $C_N^g(j,\omega)$ by a strictly positive length of time, denoted Δ. Along ω, we can compute $C_N^g(j,\omega) - \Delta$ (and hence Δ) as the time at which the additional effort recovered from idling equals the remaining effort previously expended by policy g:

$$\int_{\tau}^{C_N^g(j,\omega)-\Delta} I(t,\omega)dt = \int_{C_N^g(j,\omega)-\Delta}^{C_N^g(j,\omega)} \Big[\sum_{n=1}^{N} u_{n,(j)}^g(t) \Big] \, dt \ . \tag{3.1}$$

By iteratively applying the interchange argument to each instance of idling, idling can be eliminated and performance can be improved. □

Clearly, to minimize the completion times, all effort should be allocated to job i so as to bring it from queue 1 through its completion at queue N without interruption. We call the nonidling policy that applies all system effort successively to jobs $1, 2, 3, \ldots$ the *expedite policy*, denote it by XP, and define it formally as $u_{n,i}(t) = M$ if-f $q_i(t) = n$ and $q_{i-1}(t) = N + 1$. Note that since XP assigns all workers to one job at a time, there is no need to make use of the idealistic assumption that workers are "infinitely divisible".

Theorem 1: *Assuming that job completions must obey FCFS discipline at station N, the expedite policy achieves minimum completion times for all jobs along every sample path:*

$$C_N^{XP}(i) \leq C_N^g(i) \quad a.s. \quad \forall i \in I\!N \ . \tag{3.2}$$

Proof: We use sample path coupling to show that for any realization of the random variables, ω, $C_N^{XP}(i,\omega) \leq C_N^g(i,\omega)$, for all $g \in G$, and $i \in I\!N$. By Lemma 1, we will only show the result for "nonidling policies". Assume the first job arrives at time 0. Under a realization $\omega \in \Omega$ expressed in terms of arrival and service requirement times, we have $a(i,\omega)$ denoting the arrival time of the ith job, $S_n(i,\omega)$ denoting the service requirement time of the ith job at station n, and let $C_N^g(k,\omega)$ be the time that the kth job exits the system. Under policy XP, the completion time of the kth job will be

$$C_N^{XP}(k,\omega) = \max\{C_N^{XP}(k-1,\omega), a(k,\omega)\} + \sum_{n=1}^{N} \frac{S_n(k,\omega)}{M} \ . \tag{3.3}$$

For job 1, the claim is obvious. Since XP policy allocates all of the effort towards the processing of job 1, it achieves the minimum achievable completion time for the job,

$$C_N^{XP}(1,\omega) = \sum_{n=1}^{N} \frac{S_n(1,\omega)}{M} \ . \tag{3.4}$$

As the induction hypothesis, assume that XP achieves the minimum achievable completion times for jobs $1, 2, \ldots, k-1$. If job k arrives to an empty system, the argument for job 1 applies to job k. If job k arrives during a busy period to a system with l jobs, $l \in \{1, 2, \ldots, k-1\}$ ahead of it, then due to the FCFS discipline assumption, we know that all l jobs must be completed before job k. Let $g \in G$ be any nonidling policy with FCFS discipline at station N, which allocates some effort to job k prior to the completion of job $k-r$, where $r \in \{1, 2, \ldots, l\}$. To construct an interchange argument, we can exchange equal amounts of work between jobs $k-r$ and k as follows. Let $C_N^g(k-r,\omega)$ be the completion time of job $k-r$ under realization $\omega \in \Omega$ on probability space $(\Omega, \mathcal{S}, P)$. Then, there must be at least an $\epsilon_1 > 0$ amount of effort allocated to job $k-r$ over an interval of some length $\epsilon_2 > 0$ prior to its completion at $C_N^g(k-r,\omega)$. Similarly, by assumption, g must devote at least an $\epsilon_3 > 0$ amount of effort to job k over some time interval of length $\epsilon_4 > 0$ prior to $C_N^g(k-r,\omega)$, which we will designate $(t, t+\epsilon_4)$. Let $\epsilon = \min\{\epsilon_1, \epsilon_2, \epsilon_3, \epsilon_4\}$. To construct a modified policy g', we take ϵ amount of effort from job k during the interval $(t, t+\epsilon)$ and allocate this ϵ^2 man-hrs towards the processing of job $k-r$ during the same time period. Similarly, we take ϵ amount of effort from job $k-r$ and allocate it to job k during the interval, $(C_N^g(k-r,\omega) - \epsilon, C_N^g(k-r,\omega))$. Due to the assumption of FCFS completions at station N, job k must be available for processing during $(C_N^g(k-r,\omega) - \epsilon, C_N^g(k-r,\omega))$. Observe that the completion time of job k does not change under policy g', since the interchange applied only "shifted" the work done on job k from $(t, t+\epsilon)$ to $(C_N^g(k-r,\omega) - \epsilon, C_N^g(k-r,\omega))$, which did not change the amount of processing done on job k prior to $C_N^g(k-r,\omega)$.

It is easily seen that under g', the completion time of job $k-r$ will decrease by Δ, where Δ can be computed from the equation

$$\epsilon^2 = \int_{C_N^g(k-r,\omega)-\Delta}^{C_N^g(k-r,\omega)} \Big[\sum_{n=1}^{N} u_{n,(k-r)}^g(t) \Big] \, dt \ . \tag{3.5}$$

If $\epsilon < \epsilon_1$, continue to iterate this interchange and comparison of $k-r$ and k until $\Delta > 0$. Thus, we conclude that allocating effort to job k prior to the completion of the l jobs ahead of it only results in delaying any or all of the jobs $\{k-1, k-2, \ldots, k-l\}$ and does not change the completion time of job k.

Repeating the interchanges iteratively for each job, we reach the XP policy. XP achieves the minimum completion times for jobs $\{1, 2, \ldots, k-1, k\}$, and thus, the claim is proven inductively along every sample path in probability space $(\Omega, \mathcal{S}, P)$. □

Theorem 1 has implications for the throughput, work-in-process, and cycle time performances of the system.

Corollary 1: *In the almost sure sense, the XP policy maximizes the throughput* $\Theta^g(t) \stackrel{\Delta}{=} max\{k \in I\!N : C_N^g(k) \leq t\}$, *minimizes the WIP* $L^g(t) \stackrel{\Delta}{=} |x^g(t)| = \sum_{n=1}^{N} x_n^g(t)$ *for* $t > 0$, *and minimizes the cycle times* $W^g(i) \stackrel{\Delta}{=} C_N^g(i) - a(i)$, $i \in I\!N$.

Proof: First we observe that since $C_N^{XP}(i) \leq C_N^g(i)$ a.s., it is immediate that

$$\max\{i \in I\!N : C_N^{XP}(i) \leq T\} \quad \geq \quad \max\{i \in I\!N : C_N^g(i) \leq T\} \ \text{ a.s.} \tag{3.6}$$

Next, defining the WIP level as $\sum_{n=1}^{N} x_n^g(t) = A(t) - \Theta^g(t)$, from the preceding case we see that the WIP level is also minimized.

Because $W^g(i) = C_N^g(i) - a(i)$ and the arrival time, $a(i)$, is uncontrollable for all jobs $i \in I\!N$, the result for cycle times of jobs is immediate from Theorem 1. □

Independently, Kim [9], addressed a variant of the problem described above and argued a similar result in her thesis, which was never published. Assuming that there is no limit on the number of servers allowed at each station and that the service rate at each station is proportional to the number of servers there (i.e., work is collaborative), she states that in a tandem network of queues allocating all servers to the most downstream station with positive unfinished work is optimal with respect to minimizing sojourn times. Kim's model has the restrictive assumption of a FCFS ordering on the initiation of job service *at every station* and her result is only for minimizing sojourn times. In addition to the result for service rates that are linear in the number of servers, Kim makes the important point that if the service rate of m workers at station n is convex in m, then the XP policy, in our terminology, remains optimal. Unfortunately, as one would expect, this is not the case with service rates that are concave in the number of workers.

It is interesting to observe that the XP policy remains optimal under the assumption of finite buffers between stations, provided that jobs are not rejected at station 1. The XP policy maintains jobs in queue (and not in service) only at station 1, and at any instant keeps only one job in service. By Corollary 1, XP minimizes the total WIP, $L(t)$, (queue plus service) pathwise. Thus, the XP policy is optimally efficient from the standpoint of buffer design. It requires a buffer only at station 1 and it uses less total buffer space across stations than any other policy. Thus we have the following property:

Corollary 2: *For a given problem instance with flexible workers and without the occurrence of job rejections pathwise, the policy in G that minimizes (pathwise) the total buffer space required is XP; moreover, all of the buffer space required by XP can be pooled at station 1.*

In many applications, it is realistic to assume that the worker staffing level will vary over time due to absenteeism, vacations, maternity/parenting leaves, up/down-sizing, changes in demand, and so forth. So it is useful to extend the model from a constant number of workers M, to a total staff level $M(t)$ that varies as a function of time. We allow the staff level process $\{M(t) : t \in \mathbb{R}^+\}$ to be either deterministic or stochastic, provided that the sample paths are continuous almost everywhere. For the sake of generality, we also allow service times (underlying service requirements) to depend on the job index i, station number n, and past service completions of other jobs at other stations. However we continue to assume that the effect of multiple workers on the combined service rate be linear.

Under this increased generality, the proof of the optimality of XP remains valid as given above. The XP policy processes the farthest downstream job with maximum possible service rate, therefore maintaining the following result:

Corollary 3: *The optimality claims made in Corollary 1 for XP remain true with a time-varying staff level and general service processes.*

Finally, we note that the XP policy is an easily implemented self-organizing policy in the sense that it does not require any computation for allocation of servers to stations and jobs. Management can adjust the production rate by changing the staffing levels without disturbing the system's operation. In addition it results in an equal utilization of all workers and gives a nonidling policy. Unfortunately, many systems do not lend themselves to collaborative worksharing and hence cannot take advantage of this policy. Section 5 gives some insight into noncollaborative systems.

4 Quantification of Benefits

In this section we address the question of how large a performance increase is possible through use of agile workforce policies. We first examine the stability benefits, which result from dynamic

line balancing. Then we investigate the opportunity for performance improvements by comparing the performance of the expedite policy (in terms of cycle time) with that of an optimal static allocation policy. Because of the idealistic assumptions underlying our models, these calculations represent an upper bound on the opportunities available in realistic systems. Using exact results and approximations, we give some insights into the factors affecting the magnitude of performance improvement opportunity in a given production environment.

4.1 Dynamic Line Balancing Benefits

Dynamic line balancing through the use of flexible workers has been studied for various scenarios (see Ostolaza, McClain and Thomas [15], McClain, Thomas and Sox [13] and Zavadlav et al. [21]). Altering work assignments on-the-fly enables a production line to balance itself by shifting the workloads continuously and automatically in response to changes in the state of the system (adapted from Zavadlav et al. [21]).

To demonstrate how the expedite policy achieves dynamic line balancing, we consider the case with general, i.i.d. service times at each station and independent and identically distributed interarrival times with mean $1/\lambda < \infty$ with finite second moments for both interarrival and service times. In the *static* system we divide the workforce equally across the stations (typically with one worker per station), resulting in a mean service time of $S_n N/M$ at station n. Although it may be possible to statically balance the line and thereby achieve the maximum stability region, we are interested in realistic *static* problems for which there is at least one station which serves as a bottleneck. Let b denote the bottleneck: such that, $(N/M)\mu_b^{-1} \geq (N/M)E[S_n]$ for all n and strict inequality holds for $n = 1, 2, \ldots, b-1$.

We see that if $\lambda > \mu_b M/N$, then the effective arrival rate at station b is strictly greater than $\mu_b M/N$ (because all stations upstream of b have service rates in excess of station b). Thus a $\lambda > \mu_b M/N$ results in instability of the static system.

The XP policy, on the other hand, achieves dynamic line balancing and a greater throughput. To see that, the effective aggregate service time of the XP system has a finite second moment and mean $E[T_{XP}] \triangleq M^{-1} \sum_{i=1}^{N} \mu_n^{-1} < N/M\mu_b^{-1}$. Thus, XP increases the maximum arrival rate (equivalently, throughput) by $E[T_{XP}]^{-1} - M/N\mu_b$.

4.2 Cycle Time Opportunity

If we assume arrivals are Poisson(λ) and service times are exponential(μ_i) for all stations $i = 1, 2, \ldots, N$ we can derive an exact measure of the cycle time performance improvement achievable through use of fully cross-trained workers in a collaborative environment. As before, the total effort allocation capacity is M. Under the exponential processing time assumption, the network is a special case of a Jackson network. In steady state, it behaves as N tandem independent M/M/1 queues (see Wolff [20]), so the mean cycle time for a static allocation $u \in \mathbb{R}^N$, $u_n \geq 0$ is given by

$$W^{St}(u) = \sum_{n=1}^{N} (u_n \mu_n - \lambda)^{-1}. \tag{4.1}$$

The effective service time of the ith job under XP is given by the random variable $T_{XP} = \sum_{n=1}^{N} S_n(i)/M$. The mean cycle time of the XP policy can be computed as in Wolff [20] using the Pollaczek-Khintchine formula:

$$W^{XP} = \frac{\lambda[\mathrm{var}(T_{XP}) + (E[T_{XP}])^2]}{2(1 - \lambda E[T_{XP}])} + E[T_{XP}]. \tag{4.2}$$

We can make this dramatic simplification because the system boils down to an M/G/1 queue with mean service time, $E[T_{XP}] = \sum_{i=1}^{N} (M\mu_i)^{-1}$ and variance, $\mathrm{var}(T_{XP}) = \sum_{i=1}^{N} (M\mu_i)^{-2}$.

Although a direct comparison can be made for any static allocation, to provide a fair comparison we optimize the static allocation with respect to cycle time under the constraint that $\sum_{i=1}^{N} u_i = M$. The same result can be found using Kleinrock's square-root channel capacity assignment formula (see section 5.7 of Kleinrock [10]), as has been noted in Mandelbaum and Reiman [12].

Theorem 2: *Assume that the rate of the Poisson arrival process, $\lambda \in \mathbb{R}^+$ is such that $1 - \lambda \sum_{i=1}^{N} (M\mu_i)^{-1} > 0$. Then, the optimal static allocation of workers that minimizes the average total cycle time is given as*

$$u_n^* = \frac{M - \lambda \sum_{i=1}^{N} \mu_i{}^{-1}}{\sqrt{\mu_n} \sum_{i=1}^{N} \mu_i{}^{-1/2}} + \frac{\lambda}{\mu_n}, \qquad n = 1, 2, 3, \ldots, N. \tag{4.3}$$

The best mean cycle time that a fixed allocation can achieve is calculated as

$$W^{St*} = \frac{(\sum_{i=1}^{N} (\mu_i)^{-1/2})^2}{M(1 - \lambda E[T_{XP}])} \tag{4.4}$$

where $E[T_{XP}] = \sum_{i=1}^{N} (M\mu_i)^{-1}$ by definition.

The stability boundary, ($\rho = 1$) is captured for both the expedite policy and the optimized static allocation by the condition $\lambda < E[T_{XP}]$. For a given vector of mean underlying service requirement rates, μ, the mean cycle time performance of both flexible and static allocation systems do not change as the order of the μ_i coefficients are permuted. *Opportunity* is defined formally as the difference between the optimal static allocation cycle time (W^{St*}) and the cycle time of the XP policy (W^{XP}), given by:

$$W^{St*} - W^{XP} = \sum_{1 \le i < j \le N} \frac{2(\mu_i \mu_j)^{-1/2} + \lambda (M \mu_i \mu_j)^{-1}}{M(1 - \lambda E[T_{XP}])}\ . \tag{4.5}$$

Nevertheless, stability guarantees that the denominator is positive, so opportunity thus, the difference is always positive, which verifies that a flexible worker system outperforms a rigid one under the conditions assumed for XP.

4.3 Factors Affecting Opportunity

From a managerial perspective, it is useful to identify conditions that cause the opportunity for agile workers to be large. This provides practitioners with intuition into where cross-training, tooling and other flexibility investments are apt to be cost effective. Given the system configuration in terms of the number of tandem stations and the number of workers, we determine three factors that affect the system performance improvement achievable by full cross-training, in the case of collaborative tasks: utilization of the system, balance and variability.

The effect of utilization is simple, given a fixed number of workers. As the system gets more and more congested, the opportunity increases because the ability to shift workers to wherever they are needed most becomes increasingly crucial for system performance. In the above case where all distributions are exponential, from (4.5) we see that $\partial(W^{St^*} - W^{XP})/\partial\lambda > 0$, which implies that as the arrival rate λ increases, the performance improvement opportunity increases. This qualitative behavior carries over to the non-exponential case as can be shown using approximations like those in the next subsection.

We note that opportunity could be defined as a decrease in the number of workers required to achieve a given cycle time. This is an intuitive and useful approach, however, it is not analytically tractable. It is clear that there may be benefits to workforce agility even in lightly loaded systems. By permitting workers to staff more than one station, such a policy can achieve the same cycle time with a lower headcount.

Although opportunity is always computed with respect to optimized worker allocations, systems in which the underlying service requirements are unbalanced will provide less opportunity as the following result shows. We again consider the exponential tandem system for which the exact value of opportunity was derived above and state the following result, omitting the proof.

Theorem 3: *For an N-station tandem queueing system with exponentially distributed interarrival and processing times, a balanced system (i.e., $\mu_i = \mu_j$ for all $i, j = 1, 2, \ldots, N$) maximizes the difference between the average total cycle time of an optimal static allocation and that of a flexible system under the expedite policy.*

This is intuitive, since in the presence of clear bottlenecks, a significant fraction of worker effort should remain at the bottleneck and does not need to roam the network in search of jobs. As an example of this result, we considered a problem with 5 stations and 6 servers and compared a "balanced" system with an "unbalanced" system. Although there is no precise measure of degree to which a system is *unbalanced*, computational experience suggests that the following index B is roughly effective in capturing the main effect of balance on opportunity:

$$B = \frac{\sum_{n=1}^{N} \mu_n - N \min_n\{\mu_n\}}{\sum_{n=1}^{N} \mu_n} \ . \tag{4.1}$$

The index B is zero for a perfectly balanced system, and approaches one as $\min_n\{\mu_n\}$ goes to zero. For the balanced case of our example, we selected service rate requirements $\mu_n = 8.0$ for all n (which specifies complete balance). With six servers and five stations, the system will be stable provided that the arrival rate is such that $\lambda \leq 48/5$. To provide a sense of the range of possibilities, we also present an extreme case of unbalance, but one that would provide stability over the same region ($\lambda \leq 48/5$). To do this, we selected $\mu_1 = 2650$, $\mu_2 = 1.6$, $\mu_3 = 5320$, $\mu_4 = 3580$, $\mu_5 = 3580$, which is clearly highly unbalanced. For each case, we computed (exactly) the difference $W^{St^*} - W^{XP}$ and we present the results in Figure 1. This graph illustrates the fact that opportunity increases with system utilization and also as the service requirements become increasingly balanced.

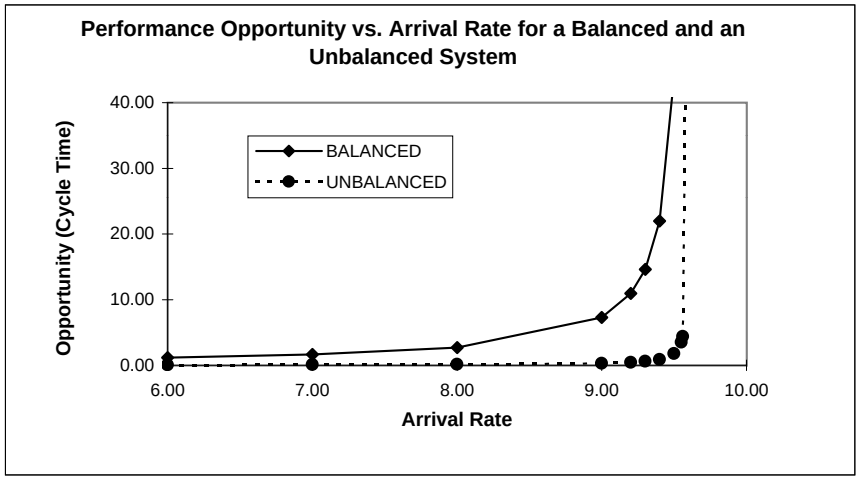

Figure 1: Opportunity, $W^{St^*} - W^{XP}$, versus λ for a balanced and an unbalanced system.

The factor of variability includes both arrival and service process variability. Since our aim is to provide insights on how variability affects the efficacy of flexible labor schemes and the expedite policy in particular, we use queueing approximations to compute average cycle times for non-exponential systems.

We consider a line of N identical station in series, with $M = N$ workers, so that the static allocation has one worker at each station. The interarrival and service times are of general distribution, with means $1/r_a$ and $1/\mu_i$ respectively. Let c_a^2 denote the squared coefficient of variation (SCV) of the arrival process and c_i^2 denote the SCV of the process time at the ith station. Then, the mean cycle time of the static system is approximated by the following (see Hopp and Spearman [7]):

$$W^{St} = \sum_{i=1}^{N} \left\{ \frac{1}{\mu_i} \left[\left(\frac{c_{d_{i-1}}^2 + c_i^2}{2} \right) \left(\frac{\rho_i}{1-\rho_i} \right) + 1 \right] \right\} \tag{4.2}$$

where $\rho_i = r_a/\mu_i$ for all $i = 1, 2, \ldots, N$ and $c_{d_i}^2 = \rho_i^2 c_i^2 + (1-\rho_i^2) c_{d_{i-1}}^2$ with $c_{d_0}^2 = c_a^2$ for all $i = 1, 2, 3, \ldots, N$. Using a standard $G/G/1$ approximation [7], the mean cycle time of the expedite

policy is approximated by

$$W^{XP} = E[T_{XP}]\left[\left(\frac{c_a^2 + c_{XP}^2}{2}\right)\left(\frac{\rho}{1-\rho}\right) + 1\right] \tag{4.3}$$

where $E[T_{XP}] = \sum_{i=1}^{N}(1/N\mu_i)$, $c_{XP}^2 = \mathrm{var}(T_{XP})/E[T_{XP}]^2 = \sum_{i=1}^{N}(c_i^2/(N^2{\mu_i}^2))/(\sum_{i=1}^{N}(N\mu_i)^{-1})^2$ and $\rho = r_a E[T_{XP}]$. Then, assuming mean interarrival and service times are fixed, we can quantify the sensitivity of opportunity to the SCV of the arrival process. For simplicity, we have taken $1/\mu_i = \tau$ for all $i = 1, 2, \ldots, N$ (and thus, $\rho = r_a\tau$) to get

$$\frac{\partial(W^{St} - W^{XP})}{\partial c_a^2} = \frac{\tau}{2}\left(\frac{\rho}{1-\rho}\right)\left[\sum_{i=2}^{N}(1-\rho^2)^{i-1}\right] > 0\ , \tag{4.4}$$

$$\frac{\partial(W^{St} - W^{XP})}{\partial c_n^2} = \frac{\tau}{2}\left(\frac{\rho}{1-\rho}\right)\left[1 + \rho^2\sum_{i=1}^{N-n}(1-\rho^2)^{i-1} - \frac{1}{N^2}\right] > 0\ . \tag{4.5}$$

From (4.4) and (4.5) we see that the opportunity $W^{St} - W^{XP}$ is linearly increasing in the SCV of the interarrival and processing times. That is, for a system with N identical stations, the opportunity will increase linearly as the SCV of the processing times increases. These approximations can be used to obtain other insights. For example, it is straightforward to show that the opportunity in a system with a highly variable machine upstream is higher than the opportunity if the highly variable machine is moved further downstream (ceteris paribus).

In summary, our analysis shows that for collaborative work environments, the opportunity of performance improvement for agile workforce policies is always positive and that well balanced systems with high utilization and highly variable demand and service processes can benefit greatly from implementing such policies.

4.4 Benefits of Partial Expedite Policy

For many real life systems, it may be infeasible or undesirable to have full cross-training because training may not be cost effective, or there may be tasks that need certain skills or a high level of specialization. Such is the case at Elgin Digital Colorgraphics. Several key operations like project management and customer interface, image scanning, color approval and final media output are still best handled by specialists. In this section, we demonstrate the benefits of implementing a partial expedite policy for systems where only partial cross-training is available.

Specifically, we consider a line with three distinct tasks, the first two of which can be implemented with an XP policy after the appropriate cross-training. We model this as three stations in tandem with a Poisson arrival process with rate $\lambda = .9$. Let S_i, which is $\exp(\mu_i = 1)$, denote the service time required for task $i = 1, 2, 3$. There are three workers in the system with infinite buffer space at each station. In the *static system,* worker i works at station i for $i = 1, 2, 3$. The model is simply a tandem Jackson network, so the cycle time is $W^{St} = \sum_{i=1}^{3}(\mu_i - \lambda)^{-1} = 30$.

In the agile system, we cross-train workers 1 and 2 to be able to process at both tasks 1 and 2 using the XP Policy. That is, workers 1 and 2 process each job at both stations 1 and 2 in collaboration without interruption. The resulting processing time is described by $T = (S_1 + S_2)/2$, which has mean 1 and a squared coefficient of variation of $c_T^2 = 1/2$. Thus the effective utilization for the resulting combination of stations 1 and 2 is $\tilde{\rho} = 0.9$; while worker 3 still works alone at station 3 as before. The total expected waiting time of this agile system for tasks 1 and 2 combined is exactly specified as

$$W_{1,2}^{XP} = \left[\left(\frac{1+c_T^2}{2}\right)\left(\frac{\tilde{\rho}}{1-\tilde{\rho}}\right)+1\right]E[T] = 7.75\ . \tag{4.1}$$

Observe that if the resulting distribution at stage 1 were approximated as exponential(1), the waiting time would be 10, since queueing is eliminated at station 1; however, the variance reduction in processing time yields an additional significant performance improvement. This reduction in processing time has yet another benefit in that it reduces the amount of variability that is propagated to station 3. Approximating the variability of the arrival process to station 3 as $c_{g,3}^2 = \tilde{\rho}^2 c_T^2 + (1-\tilde{\rho}^2)1$, we get the expected cycle time at station 3 as

$$W_3^{XP} = \left[\left(\frac{c_{g,3}^2+1}{2}\right)\left(\frac{\rho_3}{1-\rho_3}\right)+1\right]\mu_3^{-1} = 8.178, \tag{4.2}$$

which gives the agile system a resulting cycle time of approximately 15.93 (a 47% reduction).

Although we expect that it is often a simple matter to improve system performance using pooled servers some caution is advisable. Bramson [4] has constructed a tandem Jackson network with three groups of pooled servers that produced an unstable system, even though the static (unpooled) system was stable.

5 The Noncollaborative Case with Full Cross-training and Ample Equipment

A critical assumption leading to the optimality of the expedite policy is the ability of multiple workers to collaborate on the same job with no loss of total efficiency. But in many systems, collaboration is either limited to a certain number of workers or a single worker, because it is infeasible, dangerous, or simply undesirable. Theorem 1 suggests in principle that it is optimal to process the furthest downstream job with the highest service rate possible. In the case of limited collaboration constraints, an implication of this result is to create teams of workers and have workers in a team collaborate on a single job. For this reason, we restrict attention to systems in which only one worker (or team) can serve the same job.

We assume there is full cross-training and sufficient equipment at each station to ensure that workers are never blocked due to lack of equipment. For such systems we propose the nearest feasible policy to the expedite policy, in which workers work sequentially on the job furthest downstream

subject to the one-worker-per-job constraint. That is, in this policy which we call the *Pick-and-Run* (PR) policy, workers independently expedite the jobs in FCFS order through the line without interruption. For concreteness, we assume that an arriving job is paired with the server that has been idle the longest. Notice that queueing only occurs at station 1, as with XP. This implies that PR reduces the problem to a GI/G/M multiserver queue with the general service distribution given by the convolution of the station processing times.

We pause to note that although we make the assumption of ample equipment, this is less essential for the PR policy than for the XP policy. Under XP all workers are always at the same station, whereas under PR this event would be rare for most systems. Therefore, even systems with less than ample equipment where workers may be blocked and switch to other stations, the performance of an PR-like agile worksystem may approach that of the PR in the ample capacity case. PR-like policies are used in industry (e.g., Volvo assigned teams to assemble entire automobiles instead of dividing work into station specific tasks), because they are extremely simple to implement, provided full cross-training exists. Indeed, in the EDCG system cited earlier, most stations had the necessary equipment to prevent blocking and they did have workers follow jobs through most of the system.

PR, like the expedite policy, also achieves dynamic line balancing. In fact, the stability region for XP is no larger than that for PR. To see this, note that the aggregate line processing time under PR is equal to $(\sum_{i=1}^{N} \mu_i^{-1})$. Now first consider the case where $M = 1$, so that PR is identical to XP and the service rate of a single PR worker is $(\sum_{i=1}^{N} \mu_i^{-1})^{-1}$. For $M > 1$, since the workers act in parallel the maximum worker throughput capacity for PR is $M(\sum_{i=1}^{N} \mu_i^{-1})^{-1} = E[T_{XP}]^{-1}$. Hence, the PR policy achieves the same increase in throughput capacity with respect to a static policy as we saw for XP and we have the following theorem.

Theorem 4: *For a renewal arrival process with rate λ, both the XP policy and the PR policies induce stable queue length processes if $M(\sum_{i=1}^{N} \mu_i^{-1}) < \lambda$.*

It is interesting to observe that because we proved XP is optimal in minimizing the queue length process pathwise, it therefore maximizes the stability region provided we restrict attention to service policies that guarantee that job completion times from station N are ordered according to FCFS. Although PR initiates job services in the FCFS order, under non-deterministic processing times it will not yield a FCFS job departure ordering along all sample paths. Because XP was proved to be optimal in a class of policies that does not include PR, we cannot directly conclude that XP is always better than PR. However, we do not expect PR to perform as well as XP, which suggests that worker flexibility results in better opportunity for performance improvement in collaborative systems. We offer the following example as evidence. It is clear that the ideal performance of a PR system is obtained in a system in which there is a batch arrival process with

batch size M (one job per worker) with *deterministic* batch interarrival times and *deterministic* processing times at each station. For our example, the PR system is stable provided the batch interarrival times are not greater than $\sum_{i=1}^{N} \mu_i^{-1}$. Thus, the expected wait per job for this example under PR is $W^{PR}(M) = \sum_{i=1}^{N} \mu_i^{-1}$. Even so, the long-run average waiting time per job under XP is given by $W^{XP}(M) = M^{-1} \sum_{i=1}^{M} iW^{PR}(M)/M = (W^{PR}(M)/2)(1 + M^{-1})$. The performances are equal, of course, for $M = 1$, since we know PR is optimal for such a system under the assumption of only one server per job, because there is no wait in queue. But for $M > 1$, the ratio of XP to PR performance, $W^{XP}(M)/W^{PR}(M) = (M-1)/(2M)$, approaches 50% as $M \to \infty$.

We expect XP to dominate PR to an even greater extent as stochastic (non-batch) arrival processes and service times are considered. There are two key reasons for this. First, we note that the collaboration of XP induces a variance reduction of M^{-2} in the mean aggregate processing time. Second, XP delays the start of service for job i as long as possible (until jobs $1, 2, \ldots, i-1$ have been completed), thereby maximizing server utilization through any point in time. We suggest that XP or a variation of it be used in place of PR whenever possible.

Although, as we will show below, PR is not optimal, we believe that it is highly effective in problems that do not allow preemption before the completion of service at the current station. For one thing, PR keeps all workers busy whenever there is work in the system. We define a non-idling policy formally as follows.

Definition 2: *Under the assumption of one worker per job, a policy g is said to be* nonidling *if when the system queue length is $L^g(t)$ at time t, then the number of active servers is* $\min(L^g(t), M)$.

Even with a restriction on the number of jobs allowed per worker, with a slight modification the proof of Lemma 1 of Section 3 holds true pathwise. If we restrict attention to a renewal arrival process and the minimization of average cycle time per job, we this implies the following result.

Theorem 5: *Within the class of preemptive policies which allow only one worker per job, policies that idle at least one server for a positive amount of time (in expectation) for a busy cycle are strictly suboptimal.*

PR possesses the nonidling property, but we have not been able to prove conditions under which an optimal policy will, at each decision epoch, ensure that no jobs are left waiting in queue with an age (i.e., processing time already invested) that is greater than the age of any job that receives service. This property, together with the nonidling property would be sufficient to imply the PR policy. However, our research has led us to conclude that PR is not optimal, even under mild assumptions. Below we give an example in which PR is strictly suboptimal along a sample path and then extend this insight to construct an example in which PR provides a strictly suboptimal steady state expected cycle time and, hence, expected WIP level.

5.1 Example of PR Suboptimality Along a Particular Sample Path

Consider a deterministic system in which jobs arrive with a periodic structure to a three-station system with two workers. Let the first busy cycle begin with an arrival at time $t = 0$, followed by arrivals at times 0.5, 1, 4.5, and 5.0 minutes, which comprise the cycle. The next arrival does not arrive until time 9.5 to start the next busy cycle. Service times are deterministic and equal to 1.0 at each of the three stations.

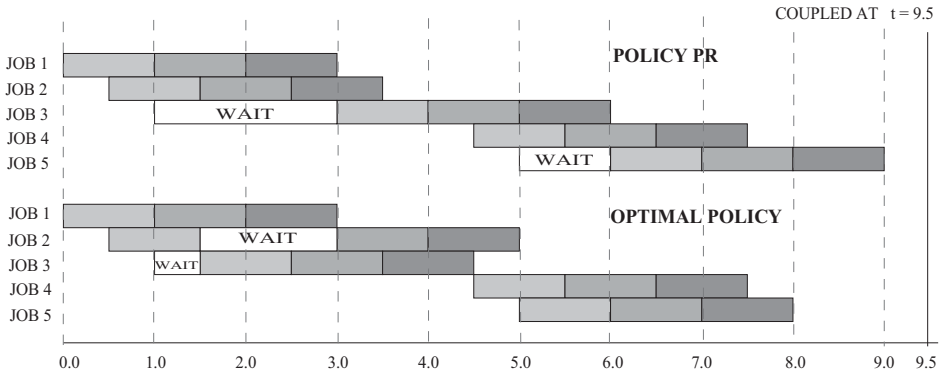

Figure 2: The suboptimality example for Pick and Run

The example is best presented graphically, as in Figure 2, which provides a Gantt chart depicting waiting times and service times on the lattice of half-minute intervals. We construct a policy called *Drop-and-Pack* (DP), which is similar to PR, with only one exception. It preempts at time 1.5 the second arrival, which has just completed service at station 1. The busy cycle has a length of 9.5 minutes. The cycle time for policy PR is 18, while it is 17 for the optimal policy. Moreover, the busy period of PR ends at time 9, while the optimal policy ends at time 8 (an improvement in the sense of makespan on every cycle).

In the next subsection we extend this sample path argument to show that PR is not optimal for systems with a general renewal arrival process, a result which we find surprising. We present a model with deterministic processing times, three stations, two workers, and a renewal arrival process for which we can explicitly define a policy that has a strictly lesser steady state expected cycle time than PR.

5.2 Example of PR Suboptimality in Mean Cycle Time

As before, consider a three-station system with two workers and deterministic processing times of length one at each station. Jobs arrive according to a renewal process. The time between arrivals is 0.5 minutes with probability $p = 0.5$ and 3.5 minutes with probability $q = 0.5$ (which yields an effective system utilization of 0.75). Observe that this model can generate the sample path realization analyzed in the previous example. As in that example, we construct a Drop-and-Pack (DP) policy. For most busy cycles, DP is identical to PR. If, however, the busy cycle begins with

an arrival at some time $t = 0$ (for convenience) followed by successive 0.5 minute interarrival times, then DP preempts at time 1.5 the second arrival, which has just completed service at station 1. The busy cycle of Figure 2 illustrates such a busy cycle. Other than this event, DP is identical to PR. With $p = 0.5$, we simulated 40 replications of both the DP and PR policies for five million job system-departures. Common random numbers were used to couple the arrival processes and thereby achieve variance reduction. Analyzing the long-run average cycle time per job advantage of policy DP with respect to PR, we found that the 500-sigma confidence interval does not include zero. The mean cycle-time advantage of DP over PR is 0.025 minutes, and the estimator had a standard error of 4×10^{-5}. The above results were verified through manual simulation traces

Policy	Mean Cycle Time	6σ Conf. Interval
Pick-and-Run (PR)	4.306	($\pm$ 0.0008)
Drop-and-Pack (DP)	4.281	($\pm$ 0.0008)
Savings of DP	.025	($\pm$ 0.00025)

Table 1: Simulation Comparison of Cycle Time Performance.

and the construction of an approximate analytical model, which provided close agreement with the simulation results. Hence, they conclusively demonstrate that the PR policy is not optimal.

The preceding example exhibits improved performance under DP under both light traffic and very heavy traffic (as p varies from 0 to 0.662, which gives a utilization as high as 0.99). It is easy to see from the above example that the service times were carefully matched to a bursty arrival process to achieve the counterexample. Even so, the performance suboptimality of PR is very slight. For this reason, we are confident that the PR policy, while not optimal, is indeed effective.

The same example also enables us to make a firm statement about the suboptimality of the performance of bucket-brigade policies when ample equipment is available to allow all workers to perform the same task (on distinct jobs) simultaneously. The standard bucket-brigade policy of Bartholdi and Eisenstein [2] assumes that a worker becomes blocked and idles, upon catching up to the station at which the next downstream worker is working. First of all, we know that the bucket brigade cannot be optimal because an idling policy cannot be optimal. Secondly, since our system allows multiple workers to be at the same station provided there is only one job per worker, the standard one-worker-per-station bucket brigade policy may not be as good as policies that take advantage of allowing multiple workers per station. However, it is straightforward to modify the bucket brigade policy to allow multiple workers at the same station. Suppose worker $m + 1$ is intended to be downstream of worker m. If worker m happens to complete job j at station n before worker $m + 1$ completes job j' at station n, then workers m and $m + 1$ can instantaneously swap jobs so that worker $m + 1$ picks up job j at station $n + 1$ and thereby remains at or downstream of the station of worker m. The worker ordering is preserved. More importantly, unnecessary idling is avoided. Observe that the performance of this modified bucket brigade policy is pathwise identical

to that of PR under assumption of identical workers. Thus, our counterexample proves that even modified bucket brigade policies cannot be optimal with respect to cycle time.

6 Collaborative and Noncollaborative Systems with CONWIP Production Control

The previous models considered make-to-order systems in which arrivals correspond to job releases. By modeling these as open queueing networks we implicitly assumed a "push" protocol. We now reconsider the issue of workforce agility in the context of a make-to-stock system operating in a "pull" environment. Specifically, we assume that there is unlimited availability of raw materials and an admission controller that releases a new job to the system whenever a job completes its processing, thereby maintaining a CONstant Work In Process (CONWIP) level. We can model this system as a closed queueing network (CQN) model with K jobs in the network. As before, there are N identical stations in series, each having an exponentially distributed underlying service requirement with rate 1 (for simplicity of presentation). The staff is of constant size, M. The workers are fully flexible (can work at any station) and identical (all have the same at all stations).

Again, we contrast agile workforce policies with a static policy, which we denote as SCW to distinguish it from the notation, St of Section 4. If M is not a multiple of N, the static system is assumed to be able to split workers (a generous assumption) so that the number of workers assigned to any station is exactly M/N. We assume for both policies that if m workers are present at a station, the resulting mean service time is $1/m$. The throughput rate of a policy is taken to be the long run rate of job completions at station N.

The analysis of the static CONWIP system is straightforward for $K = 1$, since there is no queueing of jobs. On average, the job is advanced one station every unit of time. Thus, a job is completed every N/M units of time at each station, resulting in a cycle time of N^2/M. It is important to observe that the throughput rate under the static policy is sensitive to the number of jobs, K. Mean value analysis allows us to calculate the waiting time at station i (see [16]). Under the assumption of identical workstations, a job is equally likely to be at any station in the network. Thus, the arrival theorem for CQN's indicates that the expected number of jobs at any station i seen by an arriving customer is $(K-1)/N$. In addition to the arriving job's own service time of N/M, the job will on average wait $(K-1)/N)(N/M)$ units in queue i. Summing up the waiting times at each of the N queues, we obtain the cycle time of the static system as

$$W^{SCW*} = \frac{N(N+K-1)}{M} \ . \tag{6.1}$$

The throughput follows by Little's Law, and we have:

$$\Theta^{SCW*} = \frac{KM}{N(N+K-1)} \ . \tag{6.2}$$

In the case where $K = \infty$, we see that every buffer of the static system will always have jobs present. Therefore, every $(M/N)^{-1}$ time units, a job exits station N, on average and the throughput is M/N. With $K < \infty$, worker starvation will occur with empty queues and M/N can serve as an upper bound on throughput rate.

The analysis of the XP policy is also straightforward, since there is no worker starvation. Every $1/M$ units of time on average, one job (the one farthest downstream) is advanced one station. Thus, one job is completed every N/M units of time. It is important to observe that throughput of XP is insensitive to the number of jobs, K: $\Theta^{XP}(K) = \Theta^{XP}(1) = M/N$ for all $K \in I\!N$. The reason, of course, is that XP only requires one job in the system at a time since workers collaborate fully.

From this we see that the XP system with $K = 1$ achieves a throughput rate that is achievable in the static system only with $K = \infty$. On the other hand, as system congestion increases (heavy traffic), the XP policy's throughput does not improve and is approached by the performance of static systems.

As in the make-to-order environment, the collaboration assumption may be inapplicable to many make-to-stock environments. So we now consider the noncollaborative case, but still under the assumption that machines (tools, jigs, floor space, etc.) are in sufficient supply so that all workers can work simultaneously on distinct jobs at the same station (and thus, there is no blocking of one worker by another). Again, the PR policy would appear to be an attractive one. When the number of jobs is at least as large as the number of workers ($K \geq M$), it is interesting that the PR policy introduced in the previous section can achieve the same performance in a CQN.

Furthermore, we see that the PR policy achieves 100% utilization of every server. The throughput rate of any given server is $1/N$. With K workers active, we get $\Theta^{PR} = K/N$. If $K \geq M$, all M workers serve in parallel without blocking, and the throughput rate is $\Theta^{PR} = M/N = \Theta^{XP}$. Hence, the cycle time is KN/M by Little's Law. On the other hand, if $K < M$, then PR performs worse than XP.

With $\wedge$ denoting the minimum function, the throughput of the static, PR, and XP policies are summarized in Table 2. These results are for a balanced system, which, from our previous analysis, we expect yields maximum opportunity for agile worksystems. It is helpful to plot the throughput

Number of Jobs, K	TH^{SCW}	TH^{PR}	TH^{XP}
$K = 1$	$\frac{M}{N^2}$	$\frac{1}{N}$	$\frac{M}{N}$
$K \in I\!N$	$\frac{KM}{N(N+K-1)}$	$\frac{K \wedge M}{N}$	$\frac{M}{N}$

Table 2: Closed System Throughput Rate Performance.

and the cycle time resulting from these policies as a function of WIP level. In addition, it is insightful to include the performance of the corresponding open queueing network with a traffic intensity for the open system that yields an average WIP of K: $\rho = K/N(1 + K/N)^{-1}$. This gives some indication of the magnitude of the potential performance increase from full worker cross-training as compared with the inherent benefit of employing a CONWIP release policy. Throughput and cycle time, respectively, are plotted in Figures 3(a) and 3(b). We summarize our insights in a theorem

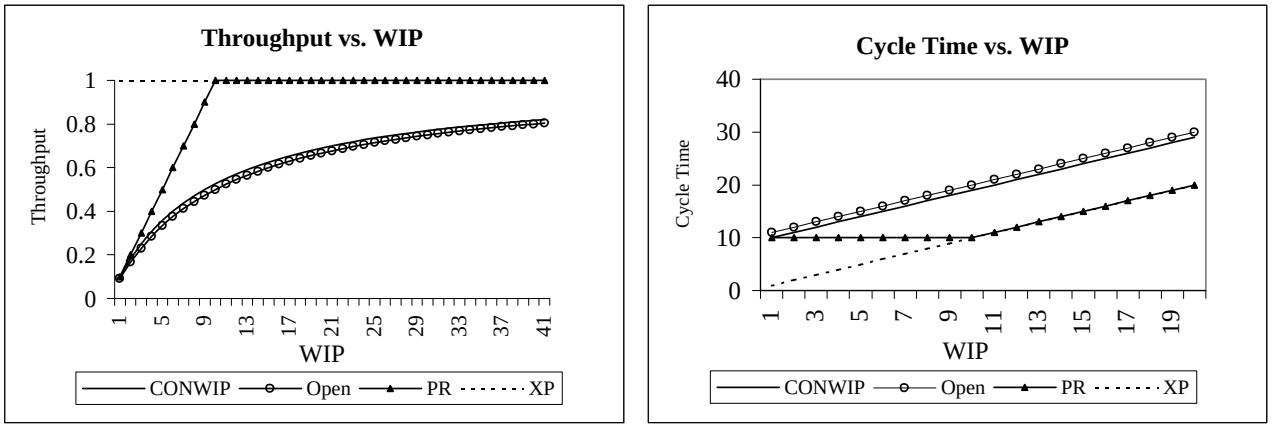

Figure 3: (a) Throughput, Θ, versus WIP, K, for CONWIP, PR, and XP; (b) Cycle time, W, versus WIP, K, for CONWIP, PR, and XP.

that asserts the optimality of XP and sometimes PR in CQN's. Note that the optimality of XP in a closed system does not follow directly from Theorem 1, because the open system model assumed an independence of the arrival process and the control policy, which is violated in a CONWIP system.

Theorem 6: *With respect to stationary Markov policies in a tandem queueing network of N stations with exponential(μ_i) processing times at station i, the expedite policy (XP) is optimal in a closed queueing network with respect to throughput and cycle time. Moreover, if $K \geq M$, the same is true for pick-and-run (PR).*

Proof: Let the steady state average number of active (i.e., non-idling) workers in station n under policy g be denoted by a vector $r \stackrel{\Delta}{=} (r_1, r_2, \ldots, r_N)$, where $\sum_{n=1}^{N} r_n = M$. We characterize the region of potential throughput performance (which will not be achieved by policies with idling) as follows. Given r, the bottleneck station limits the maximum potential throughput $\Theta(r) \stackrel{\Delta}{=} \min_n[r_n\mu_n]$. The upper bound is characterized by the problem

$$\begin{array}{ll} \max & \Theta(r) = \min_n[r_n\mu_n] \\ \text{subject to} & \sum_{n=1}^{N} r_n \leq M \\ & r_n \geq 0, \quad n = 1, 2, \ldots, N \ . \end{array} \tag{6.3}$$

The solution is obtained when every station is a bottleneck station and for some $c \in \mathbb{R}$, we have $r_n \mu_n = c$ such that $\sum_{n=1}^{N} r_n = M$. This implies $c = M(\sum_{n=1}^{N} \mu_n^{-1})^{-1}$. Thus, the maximum throughput is bounded above by

$$\bar{\Theta} = M(\sum_{n=1}^{N} \mu_n^{-1})^{-1} \ . \tag{6.4}$$

For XP, there is zero time in queue at stations $2, 3, \ldots N$, and so the effective total service time is $\sum_{n=1}^{N} (M\mu_n)^{-1}$. The resulting throughput is $\Theta^{XP} = \bar{\Theta}$, which establishes that XP achieves the maximum throughput and (by Little's Law) the minimum achievable cycle time, $W^{XP} = (K/M)(\sum_{n=1}^{N} \mu_n^{-1})$.

In the case of the PR policy with $K \geq M$, each worker achieves a throughput of $\sum_{n=1}^{N} \mu_n^{-1}$. With M workers operating in parallel (without idling), we get $\Theta^{PR} = \bar{\Theta}$, and optimality with respect to cycle time follows by Little's Law. □

In this investigation of a make-to-stock production environment using a pull protocol, we have found that XP and PR are highly effective in systems for which they are feasible. Indeed, we can make a stronger theoretical case for the efficacy of PR in the closed queueing network case than in the open case.

7 Conclusion

In this paper, we make a strong case for the logistical benefits of workforce agility in collaborative and noncollaborative production environments with significant worker cross-training. While some previous papers in the literature have offered performance analysis of queueing networks with flexible servers, we provide a comprehensive production control approach to the organization of flexible workers through an optimal queueing network control paradigm. For serial make-to-order systems with collaborative work, we proved that a simple policy, the expedite policy (XP), optimizes along each sample path the cycle time for each job. Therefore, XP also minimizes work in process and maximizes throughput rate along every sample path for any time t. We compared the performance of XP to that of the optimal static allocation of effort and demonstrated that the opportunity for cross-training and agile workforce policies is always positive and highest in balanced systems with high utilization and high variability. We showed how XP and a noncollaborative version of it, called pick-and-run (PR), achieve dynamic line balancing. However, we were also able to give a surprising counter-example to the optimality of the PR policy. Nevertheless, we present evidence that PR is effective in noncollaborative work environments. We also examined a closed queueing network model of a serial make-to-stock system with collaborative work operating under the CONWIP release policy. To our knowledge, the analysis of agile workforce policies in CONWIP systems is new to the literature. For this system, we showed that XP is still an optimal policy, but it is not uniquely optimal. Provided there is at least one job per worker, the PR policy, in which

each worker produces jobs one at a time from start to finish independent of the other workers, is also optimal.

Our analyses lead us to several insights for practical systems. Managers can get significant logistical benefits from workforce agility where the XP policy is practicable. The PR policy, while not as efficient as XP, is more widely applicable, due to the restrictive assumptions required for XP. We suspect that the main factors we found to enhance the performance improvement of the XP policy extend to all agile workforce systems: that is, systems with high utilization, volatile demand, and highly variable production processes are good candidates for worker cross-training. However, the success of such policies in various environments depends on the nature of the tasks (collaborative vs. noncollaborative), the type of equipment (general purpose, cheap tooling vs. specialized, high cost machinery), the cost of holding work-in-process (cost trade-offs between inventory and worker flexibility as forms of buffer capacity) and characteristics of the workforce (high-skilled vs. low skilled, terms of union contracts, pace of workers at tasks, etc.). The benefits of the XP and PR policies for quality and customer interaction (e.g., workers take on a higher responsibility for the product when each job is associated with a specific worker or group of workers) and ease of adaptation (e.g., to changing demand levels or production rate by simply changing the number of workers) probably also extend to other environments.

Further research is needed, however, to characterize the opportunity of workforce agility and to provide tools for analyzing the many production environments not treated here.

References

[1] Ahn, H.-S., Duenyas, I. and Zhang, R. (1997) *Optimal stochastic scheduling of a 2-stage tandem queue with parallel servers*, Working paper 97, University of Michigan.

[2] Bartholdi III, J.J. and Eisenstein, D.D. (1996) A production line that balances itself, *Operations Research*, **44:1**,21-34.

[3] Bischak, D.P. (1996) Performance of a manufacturing module with moving workers, *IIE Transactions*, **28**, 723-733.

[4] Bramson, M. (1994) Instability of FIFO queueing network, *Annals of Applied Probability*, **4**, 414-431 (correction on p.952).

[5] Downey, B.S. and Leonard, M.S. (1992) Assembly line with flexible work-force, *International Journal of Production Research*, **30:3**, 469-483.

[6] Farrar, T.M. (1993) Optimal Use of an extra server in a two station tandem queueing network, *IEEE Transactions on Automatic Control*, **AC-38**, 1296-1299.

[7] Hopp, W.J. and Spearman, M.L. (1996) *Factory Physics: Foundations of Manufacturing Management*, McGraw-Hill, Burr Ridge, IL.

[8] Iravani, S., Posner, M.J.M., and Buzacott, J.A. (1997) U-shaped lines with switchover times and costs, Technical Report, 97-13.

[9] Kim, J.M. (1993) *Optimal Design and Control of Queueing Networks*, Ph.D. Thesis, Rutgers, The State University of New Jersey.

[10] Kleinrock, L. (1976) *Queueing Systems, Volume II: Computer Applications*, Wiley, New York.

[11] Kulkarni, V.G. (1995) *Modeling and Analysis of Stochastic Systems*, Chapman & Hall, London.

[12] Mandelbaum, A. and Reiman, M.I. (1997) On Pooling in Queueing Networks, To appear in *Management Science*.

[13] McClain, J.O., Thomas, L.J. and Sox, C. (1992) On-the-fly line balancing with very little WIP, *International Journal of Production Economics*,**27**, 283-289.

[14] McClain, J.O., Thomas, L.J. and Schultz, K.L. (1996) Management of Worksharing Systems, Working Paper 96-05, Cornell University.

[15] Ostolaza, J., McClain, J. and Thomas, J. (1990) The use of dynamic (state-dependent) assembly-line balancing to improve throughput, *Journal of Manufacturing Operations Management*,**3**,105-133.

[16] Ross, S.M. (1993)*Introduction to Probability Models*, 5th Ed., Academic Press, New York.

[17] Suri, R., Sanders, J.L., and Kamath, M. (1993) Performance Evaluation of Production Networks, *Handbooks in OR & MS*, **4**, 199-286.

[18] Veatch, M.H. and Wein, L.M. (1994) Optimal control of a two-station tandem production/inventory system, *Operations Research*, **42:2**, 337-350.

[19] Weber, R.R. and Stidham, S. (1987) Optimal control of service rates in networks of queues, *Adv. Appl. Prob.*, **19**, 202–218.

[20] Wolff, R.W. (1989) *Stochastic Modeling and The Theory of Queues*, Prentice-Hall, Englewood Cliffs.

[21] Zavadlav, E., McClain, J.O. and Thomas, L.J. (1996) Self-buffering, self-balancing, self-flushing production lines, *Management Science*, **42:8**,1151-1164.