Setting WIP levels with statistical throughput control (STC) in CONWIP production lines

W. J. HOPP†* and M. L. ROOF†

We develop a simple adaptive production control method for setting WIP levels to meet target production rates in a pull production operating under the CONWIP (Constant work in process) protocol. This method, termed Statistical Throughput Control (STC), uses real-time data to automatically adjust WIP levels (via kanban cards) in the face of noisy estimates of throughput. Because STC does not rely on a steady-state model, it is well-suited to systems subject to environmental changes such as those induced by continuous improvement efforts. Using simulation, we demonstrate the effectiveness of STC under a variety of conditions, including single and multiple products, simple flow lines, routeings with shared resources and assembly systems.

1. Introduction

Push systems schedule releases, while pull systems authorize them. As a result, push systems control release rate (and hence throughput) and observe work-in-process (WIP), while pull systems control WIP and observe throughput. The advantages of pull over push are (see Spearman et al. 1990, Spearman and Zazanis 1992, Hopp and Spearman 1996, for discussions):

1. Observability: WIP is directly observable, while capacity (with respect to which release rate must be set) is not.
2. Efficiency: Pull systems can achieve the same throughput rate as a push system with a smaller average WIP level.
3. Variability: Flow times are less variable in pull systems than in push systems because pull systems regulate the fluctuation of WIP level, while push systems do not.
4. Robustness. Pull systems are less sensitive to errors in WIP level than push systems are to errors in release rate.

These observations are supported by firms that have successfully used pull systems to gain a competitive edge.

In this paper, we address the problem of how to set WIP levels in a pull system and how to adjust them in response to rapid changes in the manufacturing environment. We use a 'control chart' approach that relies on statistical summaries of actual system performance. This has two important benefits: (1) It requires almost no a priori information about processing time distributions, and (2) it automatically responds to changes in the system.
The pull system we consider is a CONWIP (Constant work in process) line which maintains a constant WIP level by allowing new jobs to enter only as existing jobs are completed. This can be achieved by using kanban cards attached to jobs. Whenever a job is completed, its kanban card is removed and sent back to the front of the line where it authorizes a new job. The primary difference between CONWIP and kanban systems is that CONWIP pulls jobs into the front of the line and pushes them between stations elsewhere in the line, while kanban pulls jobs between all stations. Because a CONWIP line requires setting only a single WIP level, while kanban requires setting WIP levels for all stations, CONWIP is inherently simpler than kanban. For this reason and because CONWIP has been shown to perform well compared to other types of pull systems (see e.g., Spearman et al. 1990, Spearman and Zazanis 1992, Hopp and Spearman 1996), we choose it as our focus.

The remainder of this paper is organized as follows: §2 contains a literature survey of various WIP setting techniques for pull systems. Section 3 describes a WIP setting method and presents preliminary results for a single product production line. Section 4 adapts the method to multi-product environments. Section 5 discusses assembly systems. A summary and conclusions are presented in §6.

2. Literature review

Since pull systems control WIP and observe throughput, their performance is critically dependent on the choice of WIP level. In systems that use cards (kanbans) to govern WIP, setting WIP is done by choosing card counts. To meet customer requirements, WIP levels (or card counts) must be large enough to achieve the desired throughput. They must also be small enough to prevent excessive WIP. Therefore, a basic problem facing pull systems is determining the minimum WIP level to attain desired throughput rate.

Monden (1983) originally summarized the Toyota approach for determining the appropriate number of kanbans at a workstation as follows:

\[
\text{number of kanbans} = \left[ DL(1 + \alpha) \right],
\]

(1)

where

- D  average demand in units of standard containers,
- L  lead time (processing time + waiting time for kanbans),
- \( \alpha \)  safety factor.

While Toyota used this method successfully (Rees et al. 1987), it has some shortcomings. First, it depends on subjective parameters, L and \( \alpha \). L is somewhat circularly defined since it depends on the waiting time for kanbans and thus the number of kanbans in the system. Presumably one must use experimentation to find effective values.

Second, it is not well-suited to changing environments. If station capacity increases or demand profile shifts, the only way to reorient the WIP level is by estimating the effect of the change on L and \( \alpha \). Since these are empirically determined, a rapid and accurate adjustment is difficult.

These problems were not obstacles to Toyota for two reasons. First, its system evolved over an extended period of time enabling them to empirically adjust the card count setting parameters. (An illustration of just how long this evolution took was provided by Taiichi Ohno (Ohno 1988), one of the architects of the kanban system,
who described how Toyota’s fabled reduction of setup times on presses from hours to minutes required 25 years of steady effort to accomplish. Second, at Toyota frequent card count changes were unnecessary because of large market share (so variations of actual demand from forecasted demand are minimal), stable product mix and volume and a cross-trained workforce that adjusted to short- and medium-term disruptions by shifting capacity in the system (Rees et al. 1987). Although it has served as a springboard for further research into the card count setting problem, the Toyota approach may be less effective in other manufacturing systems.

Rees et al. (1987) extended the Toyota approach to environments with fluctuating product mix by using the next period’s forecasted demand and the last period’s observed lead times in equation (1). Using lead time and forecasted demand information, they estimate the density function of lead times as well as the probability mass function (pmf) of \( n \), the number of kanbans. From the estimated pmf, they determine the number of kanbans that minimize shortage and holding costs.

Philipoom et al. (1987) used simulation to determine lead times at the workstations and thereby the number of kanbans required at each station to prevent backorders in a dynamic production environment. They also described factors that influence the number of kanbans required in implementing JIT production techniques. These include throughput velocity, process variation, machine utilization and autocorrelation of processing times.

Hall (1983), a proponent of stockless inventories, suggested that the only inventory that should exist is inventory that is actively in process. Consequently, determining the number of kanbans determines the level of inventory. Furthermore, he suggested that inventory levels are determined by not asking ‘how much’ inventory is needed but asking ‘why’ inventory is needed. Through experimentation, non-active inventories are eliminated and work-in-process levels (i.e. kanban counts) are reduced until those inventories are needed. Similarly, Schonberger (1982) reported on Japanese manufacturing techniques that buffer stocks (‘active’ inventory between workstations to cushion irregularities in the production system) are reduced or eliminated. Reducing buffer stock forces workers to correct the problems of irregularities and to continually improve the production process.

Queueing theory has been used by several researchers to determine the number of kanbans required in stochastic production systems. Deleersnyder et al. (1989) determined the appropriate number of kanbans for a manufacturing system with stochastic demands and machine failures by developing a discrete-time Markov model of a single card kanban system. Wang and Wang (1993) also used a Markov process approach to determine the optimal number of kanbans for partial systems consisting of one station feeding another and multiple stations feeding a single station. Graham (1992) developed a Markovian model that calculates the steady-state probability distribution for the expected number of kanbans required to control single stage processes feeding assembly lines.

Other approaches to determining the number of kanbans have included a dynamic programming model by Li and Co (1991) and a simple near-optimal heuristic for deterministic kanban systems by Moeeni and Chang (1990). Bard and Golany (1991) used a mixed integer linear program to determine the optimal kanban policy for each workstation of a multiproduct, multistage production system. Berkley (1992) developed a decomposition approximation method for kanban controlled flow shops with periodic material handling to determine the number of kanbans as well as the required material handling frequency. A review
of optimization models that determine the number of kanban cards at each work-station and the size of kanban lots can be found in Price et al. (1994).

3. WIP setting in a single-product CONWIP lines

We begin by considering a simple single-product CONWIP line, like that shown in Fig. 1. New jobs are admitted when a free kanban card authorizes their release into the line. Since jobs must have cards attached to them while they traverse the line, the system will maintain a constant number of jobs equal to the number of cards, which we denote by $m$. We assume unlimited raw material so production never has to wait for supplies. We also assume unlimited demand, so that kanban cards are removed from jobs as soon as they finish processing. In this environment, the WIP setting problem is to find the minimum number of cards required to achieve a target production rate.

Our approach relies on control charts like those used in statistical process control (SPC) (Montgomery 1991) and so we term it Statistical Throughput Control (STC). This method is similar in spirit to the STC methods developed by Spearman et al. (1989). However, their problem is fundamentally different in that their objective is to determine whether or not a periodic production quota in a pull system is likely to be achieved.

3.1. Basic procedure

We start by establishing a target throughput rate, $\lambda$, for the line. Then we monitor average throughput by measuring the mean, $\mu$, and standard deviation, $\sigma$, of inter-output times. Whenever average interoutput time falls below the inverse of target throughput by more than $3\sigma$, we declare the system ‘out of control on the low side’ and the card count is increased by one. If average interoutput time climbs above the inverse of target throughput by more than $3\sigma$, we term it ‘out of control on the high side’ and the card count is decreased by one. As long as the average interoutput time is within $3\sigma$ of the target throughput, we consider the system to be ‘in control’, and the WIP level is not changed. (Note that there is a distinction between the way we are using the term ‘out of control’ and how it is used in statistical process control. In SPC, ‘out of control’ indicates that the underlying process has changed relative to a previously stable level (i.e., because the control limits are stated in terms of the process mean). In STC, ‘out of control’ indicates that the system is not capable of attaining the specified throughput rate (i.e. because the control limits are stated in terms of the target throughput rate.)) Each time a card is added or subtracted, statistics on interoutput times are cleared. Whenever this happens, a ‘warmup

![Diagram of a single-product CONWIP line](image)

Figure 1. A single-product CONWIP line.
period’ (i.e. time until a specified number of outputs occurs) is observed before further decisions are made regarding WIP level.

One additional issue that arises in this context is that of feasibility. If the target throughput is greater than the capacity of the line (i.e. the rate of the slowest station or bottleneck), WIP will increase without bound. So, at the outset the throughput target should be subjected to a capacity check. However, even if a throughput target less than the bottleneck rate is chosen, the amount of WIP an hence the cycle time (flow time) through the line could still become excessive (e.g. if bottleneck utilization is high and/or variability in the system is extreme). To protect against this, we add a cycle time check to the procedure. If observed cycle time exceeds a specified maximum cycle time, then either the throughput target must be reduced or system capacity must be increased.

We can summarize the STC approach to WIP setting in a single-product CONWIP line as follows:

- **Step 0.** Set initial card count $m$ and warm-up period $n$.
- **Step 1.** Set target production rate $\lambda$ and maximum cycle time $CT_{max}$.
- **Step 2.** Clear statistics and wait until $n$ jobs have been output.
- **Step 3.** After each job completion, calculate:
  - (a) Average interoutput time $\mu$.
  - (b) Standard deviation of interoutput time $\sigma$.
  - (c) Average cycle time $CT$.
- **Step 4.** IF
  - (a) $CT > CT_{max}$ then revise capacity and/or $\lambda$; Go to step 1.
  - (b) $\mu > 1/\lambda + 3\sigma$ then $m \leftarrow m + 1$; Go to step 2.
  - (c) $\mu < 1/\lambda - 3\sigma$ then $m \leftarrow m - 1$; Go to step 2.
- **Step 5.** Go to step 3.

This procedure adjusts the card count dynamically on the basis of current line status. If the system remains steady (i.e. machine characteristics do not change), then this approach will cause the card count to converge to the minimum level that attains the target throughput, if such a level exists. Card counts may cycle, for instance if $m = 9$ results in a throughput above the target rate and $m = 8$ results in a throughput below the target rate. However, if the system automatically adjusts the card count (e.g., the ‘cards’ are really electronic signals generated by the same system performing the monitoring and STC calculations), such cycling is not a problem from an implementation standpoint. Finally, by alerting decision makers to potential cycle time problems, the system notifies them of the need to seek capacity expansion or a relaxation of the throughput target.

In this same vein, the control chart approach used in the STC method could be applied to the cycle time test. We could establish a target cycle time and monitor average cycle time and its standard deviation. When average cycle time exceeds the target cycle time by three standard deviations, the system alerts the decision makers to potential cycle time problems. As before, the system notifies decision makers of the need to seek capacity expansion or a relaxation of the target throughput. A potential problem in using this approach is determining the target cycle time. In general, we do not know what our target cycle time should be a priori. The same problem arises when setting $CT_{max}$ in the STC method as an ‘upper bound’ on the average cycle time. When specifying $CT_{max}$, we do not know the variance of cycle time a priori. However, since we are only using $CT_{max}$ as a rough warning signal that
the system is not capable of meeting both throughput and cycle time goals, a great deal of precision is not really required.

3.2. Steady-state performance tests

To illustrate the performance of the STC method we used simulation to apply it to a variety of four station CONWIP lines. Our first set of tests considered steady-state systems (i.e. machine parameters remain constant over time). We considered two sets of examples: (1) An exponential case, in which processing times are assumed exponential with mean processing times of 2, 4, 3 and 3 minutes for machines 1, 2, 3 and 4 respectively, and (2) a deterministic processing random outages (DPRO) case, in which processing times are deterministic but machines are subject to exponential failures and repairs, with various values of MTTF (mean time to failure) and MTTR (mean time to repair). Processing times in the DPRO model are 2, 4, 3 and 3 minutes for machines 1, 2, 3 and 4 respectively. In all runs, we made use of a warm-up period equivalent to the time required for one hundred outputs to occur before any decisions were made regarding WIP level. The simulation was run for 100,000 time units using 30 different seeds. The convergence rate of the system is given in terms of the mean number of outputs and the corresponding standard deviation until the optimal WIP level is achieved.

Table 1 lists the scenarios considered for the exponential model and for the DPRO model. In both cases, target production rates are specified as fractions of the bottleneck rate $r_b$. In the exponential case, we considered various starting card count levels and target throughput rates and their effect on the STC method. In the DPRO case, we considered different values of MTTF and MTTR for a given target production rate, namely 90% of the bottleneck rate and looked at its effect on the STC method. The final card counts achieved for the various examples are also presented in Table 1.

From these we can conclude the following:

1. For a given system, the higher the target throughput rate, the more cards required. As indicated in Table 1, the STC method appropriately adds cards when higher throughput is required.

2. The card count does indeed stabilize, although the ‘optimum’ level may not be integer (i.e. it can cycle). This can cause the procedure to appear to converge to different levels depending on the starting solution. For example, as shown in Table 1, the exponential case converged to either 8 or 9 cards. When

<table>
<thead>
<tr>
<th>Model</th>
<th>Target TH</th>
<th>(MTFF,MTTR)</th>
<th>Initial card count</th>
<th>Ending card count</th>
<th>Mean output</th>
<th>Std dev. output</th>
</tr>
</thead>
<tbody>
<tr>
<td>EXP</td>
<td>0.75$r_b$</td>
<td></td>
<td>10</td>
<td>4–5</td>
<td>2775</td>
<td>908</td>
</tr>
<tr>
<td>EXP</td>
<td>0.9$r_b$</td>
<td></td>
<td>5</td>
<td>7–8</td>
<td>11,721</td>
<td>5328</td>
</tr>
<tr>
<td>EXP</td>
<td>0.9$r_b$</td>
<td></td>
<td>10</td>
<td>8–9</td>
<td>11,076</td>
<td>4859</td>
</tr>
<tr>
<td>EXP</td>
<td>0.9$r_b$</td>
<td></td>
<td>20</td>
<td>8–9</td>
<td>18,327</td>
<td>3665</td>
</tr>
<tr>
<td>EXP</td>
<td>0.95$r_b$</td>
<td></td>
<td>10</td>
<td>10</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>DPRO</td>
<td>0.9$r_b$</td>
<td>(100,10)</td>
<td>10</td>
<td>9–10</td>
<td>18,138</td>
<td>1422</td>
</tr>
<tr>
<td>DPRO</td>
<td>0.9$r_b$</td>
<td>(1000,100)</td>
<td>10</td>
<td>2–3</td>
<td>17,950</td>
<td>4918</td>
</tr>
</tbody>
</table>

Table 1. Results for exponential and DPRO cases in a single-product CONWIP line.
the true optimum is between integers, then very slow cycling between the two values can occur.

(3) Given the same availability, machines with more variability require more WIP to achieve the same throughput as machines with less variability. The STC approach compensates for variability by adding cards.

3.3. The STC method: dynamic step-size

The STC procedure used in the previous examples always adjusts WIP level in response to an ‘out of control’ situation by increasing or decreasing it by one. However, this can cause the system to take a long time to converge to the optimal card count. This is particularly possible when the starting card count is far from the optimum. Obviously, one way to improve overall performance would be to use some kind of model (like those discussed in the literature survey) to compute a ‘good’ starting solution. However, because data are not always available for this (e.g. when the line changes but statistics on its new performance have not yet been collected), we seek an alternate method. Specifically, we propose a way to speed convergence by using ‘step sizes’ >1.

To do this, we note that in closed production systems (e.g. a CONWIP production line), mean cycle time is an increasing function of the number of jobs in the system, \( m \). Spearman (1991) proposed an analytic congestion model (ACM) that gives an approximate expression for mean cycle time as a function of WIP level and the following three parameters: (1) Raw process time, \( T_0 \) (sum of the process times at the stations in the line), (2) bottleneck rate, \( r_b \) (rate of slowest station), and (3) a congestion coefficient, \( \alpha \). The ACM expression for average cycle time for a WIP level of \( m \), is

\[
\frac{CT(m)}{C \left( M_0 \right)} = \frac{1}{r_b} \left( m + \frac{\alpha (M_0 - 1)}{\ln(1 + b_0)} \ln(1 + b_0 e^{-\frac{(m - M_0) \ln(1 + b_0) / \alpha (M_0 - 1)}}) \right),
\]

where

\[
M_0 = r_b T_0
\]

is the critical WIP level—that is, the minimum WIP level for which a line with raw process time \( T_0 \) and bottleneck rate \( r_b \) will achieve full throughput (equal to \( r_b \)) when there is no variability in the line (i.e. \( \alpha = 0 \)). The parameter \( b_0 \) is found by solving

\[
b_0 = 1 - \frac{1}{(1 + b_0)^{1/\alpha}}.
\]

We can use the ACM to modify the step size in the STC method by observing that any estimate of cycle time along with its corresponding WIP level gives us a point on the curve, and hence an estimate of \( \alpha \). Once we have this, we can use the ACM formula to find the WIP level that attains the desired throughput. (Note that given WIP levels, specifying throughput is equivalent to specifying cycle time, since by Little’s law, WIP = CT × TH).

Thus, we use the ACM to modify steps 4(a) and 4(b) in the STC procedure to the following:

Step 4(a) and (b): IF

\[
\mu > 1 / \lambda + 3 \sigma
\]
or

\[ \mu < 1/\lambda - 3\sigma \]

then

1. Solve for \( \alpha \) using equations 2, 3 and 4 with the current estimate of \( CT \) and value of \( m \).
2. Calculate \( \bar{CT}(m) \) for various values of \( m \).
3. Find \( m^* \) as the minimum value of \( m \) such that \( m/\bar{CT}(m) \geq \lambda \) (i.e. by Little’s law, \( m/\bar{CT}(m) \) is the throughput rate for a WIP level of \( m \)).
4. Set \( m \leftarrow m^* \); go to step 2.

Figure 2 compares the single step and dynamic step size STC method for the exponential case with an initial card count of 20 and the target production rate set at 90% of the bottleneck rate. This figure illustrates that dynamically adjusting the step size of the card counts can greatly reduce the time it takes to achieve the optimal WIP level. The single step method required on average more than 1000 additional outputs over the dynamic step-size before the optimal WIP level was achieved. The mean and standard deviation of the number of outputs observed before the optimal WIP level was achieved are 17247 and 374, respectively. Of course, this increase in responsiveness comes at the price of additional complexity—one must solve equations 2, 3, and 4 each time the card count is updated. However, this can be accomplished via a fairly simple search procedure that will be invisible to the user of the system in practice.
3.4. Dynamic performance tests

We now examine the performance of the STC method in situations where machine parameters dynamically change over time. In particular, we use simulation of a single CONWIP production line to show how the STC method responds to (1) increases in capacity of a machine and (2) decreases in repair time of a machine.

To illustrate the performance of the STC method in the face of increases in capacity, we simulated a DPRO CONWIP line with four machines in tandem. The initial processing times are 2, 4, 3, and 3 min for machines 1, 2, 3, and 4 respectively. The failure and repair times at each machine are exponentially distributed with a mean of 100 and 10 minutes, respectively. The target throughput rate was set at 90% of the bottleneck rate and the initial WIP level was set at 10. We simulated this line using 30 different seeds for 100,000 time units and a warm-up period of 100 outputs was observed before any new decisions were made after a change in WIP level. At 50,000 time units, the processing time at the bottleneck decreased from 4 to 3 min. Because of the increase in capacity, the STC method adjusted the WIP level downward from 10 to 3–4. Figure 3 illustrates the actual and target throughputs as well as the corresponding WIP levels of a CONWIP production line for this scenario for a single sample path. The mean number of outputs before the optimal WIP level was achieved (i.e. when the cycling between 3 and 4 cards begins) is 14,999. The standard deviation is 805.

As a second test of the performance of the STC method during dynamic changes, we examine the same CONWIP production line described above with an initial WIP level of 10 but this time, failures occur only at machine 2. The failure times at machine 2 are assumed to be exponentially distributed with a mean of 50 min.

Figure 3. Dynamic performance results: a CONWIP production line subject to an increase in capacity.
Repair times are also assumed to be exponentially distributed. At the beginning of the simulation, the repair time has a mean of 5 min. However, after 5000 time units, the repair time is decreased to 1 min. Total simulated time is 15 000 time units and we simulated the exact same line using 30 different seeds. Target throughput is set at 0.225 (90% of the bottleneck rate). Figure 4 illustrates the actual and target throughputs as well as the corresponding WIP levels of a CONWIP production line under this scenario for a single sample path. The mean number of outputs observed before reaching the optimal WIP level is 2046 with a standard deviation of 187. This simulation illustrates the manner in which the STC method enables a line to respond to a change in the equipment maintenance policy. We also simulated the same CONWIP line subject to an incrementally decreasing repair time from 5 min to 1 min and observed similar results.

From the results illustrated in Figs 3 and 4, we can conclude the following concerning the response of the STC method to dynamic changes:

1. The method is able to adjust to the dynamic changes of a production line including increases in line capacity and decreases in repair times. Cycling can occur, as happened in Fig. 4 as capacity stabilized and the ‘optimum’ WIP level was between 2 and 3 cards.
2. Increases in capacity reduce the WIP level needed to achieve a target throughput. For instance, WIP decreased from 10 to 3–4 in Fig. 3.
3. Decreases in repair times reduce the WIP level needed to achieve a target throughput. For instance, WIP decreased from 10 to 2–3 in Fig. 4.

4. **WIP setting in multi-product CONWIP systems**

To illustrate the performance of the STC method in multi-product systems, we used simulation to apply it to a variety of four station CONWIP lines. We only considered steady-state systems (i.e. machine parameters remain constant over time)
and used the single step size option of the STC method. Of course, the dynamic step size method could be used and the method would be expected to respond to dynamic changes in the system parameters in a manner similar to that noted above for single product systems. But for clarity of exposition, we restrict attention to the steady-state case.

We considered two sets of examples: (1) a single line with multiple products and (2) multiple lines with shared machines.

In the single line example, two products are produced and processing times for both products are assumed deterministic. The mean processing times for product A were assumed to be 1, 2, 1, and 1 min for machines 1, 2, 3, and 4, respectively. For product B, the mean processing times were 3, 3, 4, 3 min for machines 1, 2, 3, and 4, respectively. These numbers were chosen to reflect the situation of a shifting bottleneck (i.e. machine 2 is the bottleneck for A, while machine 3 is the bottleneck for B). The failure and repair times at each machine are exponentially distributed with a mean of 400 and 40 min, respectively.

Table 2 lists the parameters considered for the single line with multiple products. The target production rate for the line is listed as well as the initial card count for the line. Note that the card count controls total WIP, that is of both products A and B. The actual mix in the system is controlled by the production sequence. We considered two different sequences, both of which produce an output mix of 2/3 A and 1/3 B. In the first sequence, two product As are scheduled followed by one product B and the sequence is repeated. In the second sequence, eight product As are scheduled followed by four product Bs and the sequence is repeated. For both sequences, we simulated the line for 5000 minutes and a warm-up period of 10 outputs was observed before any new decisions were made after a change in WIP level. The same system was simulated using 30 different seeds. The average convergence to the optimal WIP level is given in terms of output and time and these are also presented in Table 2.

In the cases representing multiple lines with shared machines, two products are produced (a different product on each line) on the lines illustrated in Fig. 5 and processing times for both products are deterministic. The mean processing times for product A on route 1 were assumed to be 1 min for machines 1a, 2a, 3, 4a, and 5a. For product B on route 2, the mean processing times were 2 min for machines 1b, 2b, 3, and 4b. Note that products A and B share machine 3. This line was simulated for 2000 minutes with a warm-up of 50 outputs. There are no random outages on the machines.

Table 3 lists the parameters considered for this scenario. Initial and ending card counts are given, as well as target production rates for each product. Ending card counts are also presented along with the time and number of outputs observed before the optimal WIP was achieved. Seven cases are examined.

<table>
<thead>
<tr>
<th>Sequence</th>
<th>Target TH</th>
<th>Initial card count</th>
<th>Ending card count</th>
<th>Mean output</th>
<th>Std dev. output</th>
</tr>
</thead>
<tbody>
<tr>
<td>aab</td>
<td>0:225</td>
<td>6</td>
<td>3</td>
<td>129</td>
<td>183</td>
</tr>
<tr>
<td>aaaaaaaabbb</td>
<td>0:225</td>
<td>6</td>
<td>2</td>
<td>271</td>
<td>310</td>
</tr>
</tbody>
</table>

Table 2. DPRO examples for a single CONWIP line with multiple-products.
Similar to the single-product CONWIP line, we can conclude the following for multiple-product CONWIP systems:

1. The STC approach adjusts to shifting bottlenecks for a single CONWIP line that produces multiple products and an appropriate WIP level is obtained.
2. Target throughput for a single CONWIP line with multiple products is achieved. The sequence order does have an effect on the optimum WIP level. For example, as shown in Table 2, the WIP level converged to 3 cards for sequence aab and 2 cards for sequence aaaaaaaaaabbb. For this example, the rate of convergence for sequence aab was more than twice as fast as the rate of convergence for the sequence aaaaaaaaaabbb. Sequence aab reached its optimum WIP level on average at time 393 and observed on average 129 outputs while sequence aaaaaaaaaabbb reached its optimum WIP level on average at time 925 and observed on average 271 outputs.
3. Target throughputs are also met for multiple CONWIP lines that share a single resource regardless of the initial WIP levels. As expected, the required card count increases in the target throughput rate, as seen in Table 3.
5. WIP setting in CONWIP assembly systems

Finally, we tested the performance of the STC method for assembly lines, again using simulation. We consider a CONWIP assembly system like that illustrated in Fig. 6. Note that parts are pulled into the fabrication lines only when an assembled job has completed processing, thereby maintaining constant, although possibly different, WIP levels in the fabrication lines. The STC method monitors throughput at the assembly process. Whenever throughput falls below the target throughput, WIP is added to the fabrication line that has the lowest service level (i.e. probability of having a job ready when the assembly station needs it). Similarly, whenever throughput exceeds the target throughput, WIP is removed from the fabrication line that has the highest service level. As before, the determination that the throughput is above or below the target is made using the $3\sigma$ criterion.

We tested the STC method for two assembly systems under steady-state conditions: (1) two DPRO lines feed the assembly station which is the bottleneck, and (2) two DPRO lines feed the assembly station which is not the bottleneck. In both systems, the assembly station requires a single part from each of the fabrication lines.

Each fabrication line consists of three machines in tandem, where processing times on each machine in line 1 and line 2 are deterministic with a mean of 0.5 and 1 min, respectively. Random outages and repair times are exponentially distributed and occur at machine 1 on both fabrication lines with a MTTF of 10 minutes and a MTTR of 1 minute. The assembly times are assumed deterministic and equal to 1.1 min for case (1) and 0.5 min for case (2). Initial parameters and ending card counts for the two assembly systems examined are presented in Table 4. Simulation time for these lines was 3000 minutes and a warm-up period of 25 outputs. Again, each system was simulated 30 times using 30 different seeds.

Figure 7 illustrates the evolution of WIP level in the two fabrication lines, as well as actual throughput and target throughput for case (1) where the assembly station is the bottleneck for a single sample path. Figure 8 illustrates similar results for case (2) where the assembly station is not the bottleneck for a single sample path. From these figures and Table 4 we can conclude the following:
Table 4. DPRO assemblies.

<table>
<thead>
<tr>
<th>System</th>
<th>Target TH</th>
<th>Initial card count</th>
<th>Ending card count</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>b/n</td>
<td>0.75</td>
<td>5</td>
<td>1</td>
<td>Line 1</td>
</tr>
<tr>
<td>not b/n</td>
<td>0.75</td>
<td>5</td>
<td>1</td>
<td>1 and 2</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>4</td>
</tr>
</tbody>
</table>

Figure 7. CONWIP assembly system: assembly station is the bottleneck.

Figure 8. CONWIP assembly system: assembly station is not the bottleneck.
(1) The STC method achieves target throughput whether or not the assembly station is the bottleneck. Card counts ‘stabilize’ for each fabrication line. However, cycling occurs when the ‘optimum’ WIP level for a fabrication line is not an integer.

(2) The STC method allocates more WIP to fabrication lines with longer processing times than those with shorter processing times. This is consistent with the goal of obtaining target throughput with minimum WIP.

6. Conclusion

The statistical throughput control (STC) method is a simple but effective way to adjust card counts in CONWIP systems to achieve desired throughput rate with minimum WIP and cycle time. By relying on statistical summaries of actual performance, the method is entirely free of assumptions about process time distributions, failure time, rework, and so forth. We demonstrated its application to simple single-product CONWIP lines, multi-product CONWIP systems and assembly systems. We also showed its effectiveness under steady-state and dynamic environmental conditions.

In short, STC is an easily implementable and practical solution to the WIP setting problem faced in all pull production systems. Of course, it is also simple and intuitive, since it is obvious that WIP (card count) must be increased if throughput is insufficient. However, the power of the STC approach is that it detects statistically significant deviations from target throughput and therefore avoids reacting to noise in the system. Furthermore, by making use of a theoretical model to select efficient step sizes for the adjustments, STC can find the appropriate WIP level more rapidly than would be possible by making ad hoc adjustments. Finally, STC incorporates these failures into an automated framework, which enables it to follow changes in the system and efficiently regulate WIP accordingly. As such, it has the potential to greatly simplify monitoring and management of pull systems.

Although we have illustrated how STC can be used in a wide range of CONWIP systems, further work is needed to apply it to other forms of pull systems. Specifically, STC should be applicable to a kanban system (in which jobs are pulled between all stations, instead of only into the front of the line, as occurs under CONWIP). However, since a kanban system has many card counts to set, the system must include a mechanism for determining which count(s) to alter when the output rate from the system is detected to be ‘out of control’. The appropriate mechanism would seem to be a straightforward extension of the assembly system logic introduced above, in which WIP was added to the line where service level indicated it was most needed. However, in a kanban system the service level at a station is defined as the probability the next station downstream has a part when needed. We would monitor service level at each station and increase the WIP level of the worst performing station when we detect an ‘out of control’ with actual throughput below target throughput. Similarly we would decrease the WIP level at the best performing station when we detect an ‘out of control’ with actual throughput greater than target throughput. The stability and speed of convergence of this method remain to be tested.

Further work is also needed in environments that are not ‘sell all you can make’. In such environments, the kanban cards might remain attached to jobs waiting in finished goods inventory (FGI) so as to link production to demand. Here the problem is not to choose the WIP level to attain a given throughput rate, but
rather to set WIP to attain a given service level. Again a logical extension seems straightforward. Each time a demand arrives it either is filled from stock or not. We could monitor the fraction of jobs filled from stock and when the actual service rate is 'out of control' relative to a target service rate, we would either add or subtract a card as needed.

Acknowledgements

This work was supported in part by a grant from the National Science Foundation Grant DMI-9322830.

References


