Optimal work-in-process inventory levels for high-variety, low-volume manufacturing systems

Srinivasan, Mandyam M.; Viswanathan, S.

2010

Srinivasan, M. M., & Viswanathan, S. (2010). Optimal work-in-process inventory levels for high-variety, low-volume manufacturing systems. IIE transactions, 42(6), 379-391.

https://hdl.handle.net/10356/100803

https://doi.org/10.1080/07408170902761406

UIIE #376310, VOL 42, ISS 6

Optimal work-in-process inventory levels for high-variety low-volume manufacturing systems

MANDYAM M. SRINIVASAN AND S. VISWANATHAN

QUERY SHEET

This page lists questions we have about your paper. The numbers displayed at left can be found in the text of the paper for reference. In addition, please review your paper as a whole for correctness.

Q1: Au: You do not define **e**.
Q2: Au: Please ensure that all vectors and matrices are in bold roman.
Q3: Au: What is PR.
Q4: Au: Department & University address.
Q5: Au: Page no.
Q6: Au: Publishers location & page nos.
Q7: Au: Publication city.
Q8: Au: Full conference title, publishers location & page nos.

TABLE OF CONTENTS LISTING

The table of contents for the journal will list your paper exactly as it appears below:
Optimal work-in-process inventory levels for high-variety low-volume manufacturing systems
Mandyam M. Srinivasan and S. Viswanathan

IIE Transactions (2010) **42**, 1–13

ISSN: 0740-817X print / 1545-8830 online
DOI: 10.1080/07408170902761406

Optimal work-in-process inventory levels for high-variety low-volume manufacturing systems

MANDYAM M. SRINIVASAN[1,*] and S. VISWANATHAN[2]

[1]*College of Business Administration, The University of Tennessee, Knoxville, TN 37996, USA*
E-mail: msrini@utk.edu
[2]*Nanyang Business School, Nanyang Technological University, Singapore, Republic of Singapore*
E-mail: asviswa@ntu.edu.sg

Received July 2008 and accepted November 2008

This paper considers a manufacturing system that operates in a high-variety low-volume environment, with significant setup times. The goal is to determine the optimal Work-In-Process (WIP) inventory for operating the system to meet the required demand for each product. The decision variables are the number of pallets (containers) for each product and the number of units in each pallet (lot size). The objective is to minimize the total WIP inventory across all products. To capture congestion in the system, it is modeled as a closed queueing network with multiple product types. However, this leads to a complex non-linear integer program with a non-convex objective function. A lower bound on the objective function is developed which is used to develop upper and lower bounds on the number of pallets for each product. The bounds on the number of pallets allow the use of exhaustive enumeration within these bounds to obtain the optimal solution to this complex queueing network-based optimization problem. A simple heuristic is developed to further reduce the number of candidate configurations evaluated in the search for the optimal solution. A computational study reveals that the heuristic obtains the optimal solution in many of the test instances.

[Supplementary materials are available for this article. Go to the publisher's online edition of *IIE Transactions* for free supplemental resources containing details on some procedures and heuristics.]

Keywords: Job shops, shop floor planning and control, CONWIP, closed queueing networks, multiple chains, non-linear integer program

1. Introduction

Once seen as just another new management fad, lean manufacturing and just-in-time manufacturing concepts are still finding increased application and acceptance, as organizations continue their efforts to remain competitive in a global economy. These concepts are well understood and readily applied to organizations with a relatively stable demand and a relatively small product mix. For such organizations, there is a well-developed methodology, primarily based on the Toyota Production System (Ohno, 1988; Monden, 1993). A key feature of lean manufacturing systems is their use of pull signals to trigger production. The pull mechanism is typically implemented using kanbans or a Constant Work-In-Process (CONWIP) protocol. These implementations place a cap on the Work-In-Process (WIP) inventory.

Placing a cap on WIP inventory has a number of benefits. It reduces flow times, reduces variation and improves quality (Suri, 1998). Setting WIP inventory levels for pull systems is, however, not straightforward. It is well known that pull systems set WIP inventory levels and observe throughput (Hopp and Spearman, 2000). Determining WIP inventory levels that will satisfy a given set of throughput requirements presents a challenge, especially for plants operating in a high-product-variety low-demand-volume environment. In such an environment, a small change in WIP inventory level for one product can dramatically affect the throughputs of other products that share common resources, particularly if these resources are potential bottlenecks. A poor choice of WIP levels can lead to demands not being satisfied. This problem is often exacerbated by the presence of large setup times, where smaller lot sizes could result in non-bottleneck stations becoming bottlenecks (see, for instance, Goldratt and Fox (1986)).

This paper is motivated by earlier work carried out by one of the co-authors at Woodward Aircraft Engine Systems in

*Corresponding author

Rockford, Illinois. This organization manufactures a wide range of products such as fuel nozzles and governors for the aircraft industry. Consequently, demand for these products is typically very small. A typical cell in Woodward produces over 200 products with annual demand ranging from 40 to 200. Setup times at the workstations are typically very large, often about 30 to 50 times the processing time per unit. Woodward required a methodology to determine the batch size (lot size) for each product type machined in a cell, and the amount of WIP inventory to carry in order to meet the demand placed on each product type.

A spreadsheet-based methodology was developed for Woodward by Srinivasan *et al*. (2003), to set the lot size and WIP inventory levels needed to meet the required demand on the different products. Each cell was modeled as a Closed Queueing Network (CQN) with multiple customer classes, where each customer class represented a different product type. A heuristic bisection search procedure was used to determine the number of pallets for each product type and the number of parts in each pallet (the lot size). Using this methodology, Woodward was able to reduce the WIP inventory in its cells from 3 weeks to less than 1 week of inventory. While Woodward found the bisection approach effective in determining the lot sizes and number of pallets, there was a question whether the bisection approach could be improved and whether the optimal solution to the problem could be determined.

In this paper, we consider a problem setting very similar to the situation faced by Woodward, and present an approach to determine the optimal lot sizes (in each pallet) and number of pallets (or *containers* or *lots*) for each product type, so as to meet the customer demands on each product type. As in Srinivasan *et al*. (2003), we use a CQN model with multiple customer classes to model the different product types. The decision variables are the number of pallets for each product and the number of products placed in each pallet (lot size). The problem of minimizing the value of the WIP inventory is formulated by developing state equations for the CQN using mean value analysis. However, this leads to a complex non-linear integer program with a non-convex objective function.

We first develop a lower bound for this objective function. Using this bound and further analysis, we develop upper and lower bounds on the number of pallets for each product. The bounds on the number of pallets allow us to use exhaustive enumeration within these bounds to obtain the optimal solution to this complex queueing-network-based optimization problem.

The optimal algorithm works well for problems with up to ten or 20 products but the computational effort required increases exponentially with the number of products in the system. Therefore, we also develop a simple heuristic to further reduce the number of configurations evaluated in the search for the optimal solution. Our computational study reveals that the heuristic obtains the optimal solution in many of the test instances.

The next section reviews related literature on production control in manufacturing systems. Section 3 introduces the notation and the optimization model. In Section 4, we develop lower bounds on the objective function as well as upper and lower bounds on the optimal number of pallets for each product and upper bounds on the total number of pallets in the system. We use these bounds to develop an enumeration-based optimal algorithm. In Section 5, we develop a heuristic for the problem. Section 6 provides several numerical examples and discusses the results of a comprehensive computational study. Concluding remarks are provided in Section 7.

2. Literature review

A significant amount of research exists on determining lot sizes for manufacturing systems. Similarly, many papers exist on determining the number of containers/pallets that will satisfy some objective function. We review work on lot size models, and the use of queueing network models to optimally determine the number of pallets under various assumptions.

Approaches to setting parameters such as lot sizes or number of pallets typically fall into one of two categories. One approach uses optimization models with deterministic processing times and either deterministic/constant demand (see, for instance, Maxwell and Muckstadt (1985) and Maes and Van Wassenhove (1986)) or time-varying demand (see, for instance, Billington *et al*. (1983), and Erenguc and Tufekci (1987)). These models typically do not consider congestion effects (queueing delays). Hill and Raturi (1992) include a constraint for queue times at workstations; these times are derived using an $M/G/c$ queueing system to model each workstation.

The other approach is based on queueing network models. Queueing network models present an analytical challenge. Even with some simplifying assumptions, the resulting expressions for performance measures of interest are quite complex. However, despite their complexity, queueing network models are widely used to model and analyze manufacturing systems because they explicitly consider congestion effects. Moreover, as shown by Suri (1985), these models are robust in the sense that they can analyze real-world manufacturing systems with rather forgiving assumptions, such as the product form assumption (Baskett *et al*., 1975).

Queueing models, in turn, can be categorized as descriptive (provide values for performance measures of interest for a given configuration), or prescriptive (provide guidelines for running the system most effectively). An alternate categorization is Open Queueing Network (OQN) versus CQN models. OQN models are more appropriate for push systems, whereas CQN models are appropriate for pull systems. See Govil and Fu (1999) for a comprehensive survey of queueing models in manufacturing.

There is a very large body of literature on descriptive or prescriptive single-queue models (see, for instance, Karmarkar (1987) and Benjaafar (1996)), OQN models (see, for instance, Karmarkar *et al*. (1985), and Dessouky (1998)), and CQN models (see, for instance, Solberg (1980), Spearman and Zazanis (1992), Dar-El *et al*. (1999) and Lee *et al*. (2006)). For the problem we consider, CQN models are more appropriate. More specifically, since we are considering multiple product types in the manufacturing system, the appropriate model is a CQN with multiple customer classes (Baskett *et al*., 1975). Thus, we restrict discussion to a *prescriptive* CQN with multiple customer classes.

In a manufacturing context, CQN models are often referred to as being CONWIP models (Spearman *et al*., 1990). Golany *et al*. (1999) study a CONWIP model with multiple customer classes, where products are grouped into families and machines are grouped into cells. They provide a heuristic solution to a mathematical program to determine the optimal allocation of a fixed number of containers to cells. Their numerical experiments suggest that flow times are lower if containers are allowed to migrate freely among the cells.

Ryan *et al*. (2000) study a model with multiple products that have distinct routings, with the objective of determining the minimum total WIP inventory and the WIP mix to satisfy a given service level across product types. Ryan and Choobineh (2003) develop a non-linear programming model to determine WIP inventory level for each product in a CQN with multiple product classes. Ryan and Vorasayan (2005) develop a non-linear program to optimize the number of containers for each class in a multiple class CQN model with lost sales.

Suri *et al*. (2005) present a CQN model with multiple classes where the number of customers in each class need not be an integer. The rationale for allowing a non-integer number of customers is that with fractional numbers it is possible to meet a required throughput exactly. They interpret these values to represent the average number of customers in each class in the network, and implement a control rule to achieve these averages. Askin *et al*. (2006) consider a multiple class CQN model and obtain a fixed point solution that determines the (fractional) number of customers in each class to meet a given throughput for each class. These papers use an approximation technique, the Schweitzer–Bard approximation technique, developed independently by Schweitzer (1979) and Bard (1979). The technique is based on the Mean Value Analysis (MVA) algorithm (Reiser and Lavenberg, 1980). The MVA algorithm is an exact algorithm that provides performance measures such as throughput, mean response times and the mean number of customers at individual stations in the network for a CQN model. However, the computational complexity increases exponentially with the number of customer classes, which makes the MVA algorithm impractical to use in real-world problems.

The above papers typically focus on determining the number of pallets required to satisfy a given demand mix. Further more, they assume that setup times at workstations are insignificant. The problem we address requires explicit consideration of both the lot size *and* the number of pallets for each product type for two reasons: (i) there is a significant setup time involved at each workstation; and (ii) there is a limit on the number of parts a pallet can hold.

3. The model

The manufacturing system uses M workstations to process J different products. The mean setup time for a batch of product j at workstation m is S_{mj} and the mean processing time per unit of product j at workstation m is P_{mj}. The demand per unit time for product j is D_j.

The number of batches or pallets of product j in the system at any time (as per the CONWIP protocol) is N_j; that is, as soon as a pallet of product j is processed and exits the system, a new pallet of product j is released into the system to maintain a constant WIP inventory. The batch size or number of units in a pallet of product j is B_j. All units in a pallet are processed with a single setup at a workstation and each pallet incurs a separate setup. The maximum amount of WIP in the system is controlled by the decision variables, N_j and B_j.

The manufacturing system described above is modeled as a CQN. Based on the MVA algorithm of Reiser and Lavenberg (1980), the performance of the system, for particular values of $\mathbf{N}$ and $\mathbf{B}$, is described using the following equations.

$$R_{mj}(\mathbf{N},\,\mathbf{B}) = (S_{mj} + P_{mj}\,B_j)(1 + Q_m(\mathbf{N} - \mathbf{e}_j,\,\mathbf{B})) \quad m = 1, \ldots, M, \quad j = 1, \ldots, J, \tag{1}$$

$$CT_j(\mathbf{N},\,\mathbf{B}) = \sum_{m=1}^{M} R_{mj}(\mathbf{N},\,\mathbf{B}) \quad j = 1, \ldots., J, \tag{2}$$

$$Q_m(\mathbf{N},\,\mathbf{B}) = \sum_{j=1}^{J} q_{mj}(\mathbf{N},\mathbf{B}) \quad m = 1, \ldots., M, \tag{3}$$

$$q_{mj}(\mathbf{N},\,\mathbf{B}) = N_j\,R_{mj}(\mathbf{N},\,\mathbf{B})/CT_j(\mathbf{N},\,\mathbf{B}) \quad m = 1, \ldots., M,\ j = 1, \ldots., J. \tag{4}$$

Note that $R_{mj}(\mathbf{N},\,\mathbf{B})$, $CT_j(\mathbf{N},\mathbf{B})$, $Q_m(\mathbf{N},\mathbf{B})$ and $q_{mj}(\mathbf{N},\mathbf{B})$ are all functions of $\mathbf{N}$ and $\mathbf{B}$. The notation presented so far will now be summarized.

J	: number of products in the manufacturing system;
M	: number of machines or workstations in the system;
S_{mj}	: mean setup time for product j on workstation m;

P_{mj} : mean processing time per unit of product j on workstation m;

D_j : demand per unit time for product j;

N_j : number of pallets (or batches) of product j allowed in the system;

B_j : number of units per pallet (or batch size) for product j;

B_{LO}, B_{HI} : constraint on smallest and largest batch size: $B_{\mathrm{LO}} \leq B_j \leq B_{\mathrm{HI}}$;

K_j : dollar value per unit of WIP of product j;

$\mathbf{N}$: $j = 1,\ldots,J$;

$\mathbf{B}$: $j = 1,\ldots,J$;

$R_{mj}(\mathbf{N}, \mathbf{B})$: average time spent by a batch of product j at workstation m;

$CT_j(\mathbf{N}, \mathbf{B})$: average cycle time (time in system) for a batch of product j;

$q_{mj}(\mathbf{N},\mathbf{B})$: average number of batches of product j at workstation m;

$Q_m(\mathbf{N}, \mathbf{B})$ $\sum_{j=1}^{J} q_{mj}(\mathbf{N}, \mathbf{B})$: average number of batches of all products at workstation m;

$NTOT$: $\sum_{j=1}^{J} N_j$, the total number of batches of all products in the system, note that $NTOT$ is also equal to $\sum_{m=1}^{M} Q_m(\mathbf{N}, \mathbf{B})$.

The reader might note that this system does not satisfy the Product Form (PF) assumption since the presence of setup times implies that the service time for a pallet (setup time plus processing time for the batch) is unlikely to follow an exponential distribution. However, we know that the PF assumption is satisfied for general service times if we assume the Last-Come First-Served (LCFS)-PR queue discipline (Baskett *et al.*, 1975). We also know that the expected time spent at a station is unaffected by the queue discipline (at least with a single class of customers). Thus, if we ignore the preempt–resume requirement, but simply assume that the station completes service on an entire batch before picking up the next batch (in LCFS order), we can model the system using the PF assumption reasonably well (also, see, Suri (1985)).

Let $\lambda_j(\mathbf{N}, \mathbf{B})$ be the throughput of product j per unit time achieved by the system for a given $\mathbf{N}$ and $\mathbf{B}$. As the demand for product j is D_j, we require $\lambda_j(\mathbf{N}, \mathbf{B}) \geq D_j$. By Little's law, $\lambda_j(\mathbf{N}, \mathbf{B}) = N_j B_j / CT_j(\mathbf{N}, \mathbf{B}) \geq D_j$. Therefore,

$$N_j B_j \geq D_j CT_j(\mathbf{N}, \mathbf{B}), \quad j = 1, \ldots, J. \tag{5}$$

Practical considerations on pallet size and the number of units that can be processed as a batch at a workstation without preempting it impose constraints on the smallest batch size, B_{LO}, and the largest batch size, B_{HI}, allowed for any product. Therefore, we require:

$$B_{\mathrm{LO}} \leq B_j \leq B_{\mathrm{HI}}. \tag{6}$$

To evaluate $R_{mj}(\mathbf{N}, \mathbf{B})$ for a particular value of $\mathbf{N} = \mathbf{N}_1$, we must evaluate $Q_m(\mathbf{N}_1 - \mathbf{e}_j, \mathbf{B})$ which, in turn, requires us to determine $R_{mj}(\mathbf{N}_1 - \mathbf{e}_j, \mathbf{B})$. This essentially entails evaluating $R_{mj}(\mathbf{N}, \mathbf{B})$ for all possible values of vector $\mathbf{N}$ in the range $0 \leq \mathbf{N} \leq \mathbf{N}_1$. We face the curse of dimensionality and typically approximations such as the Schweitzer–Bard approximation are used to obtain the values for performance measures of interest. We note that we use the Schweitzer–Bard approximation only to determine the batch sizes, $\mathbf{B}$, for a particular value of $\mathbf{N}$, and the corresponding cycle times. However, the lower bound on the optimal value of the objective function and the bounds on $\mathbf{N}$ used in the enumerative optimal algorithm are all determined using the (exact) MVA algorithm rather than any approximation technique.

Our objective is to minimize the dollar value of the WIP inventory required to meet the demand for each product. The problem is formulated as follows:

$$\text{(P-OPT)}: \quad \min \quad Z(\mathbf{N}, \mathbf{B}) = \sum_{j=1}^{j} K_j N_j B_j,$$

$$\text{subject to: Constraints} \quad (1), (2), (3), (4), (5) \text{ and } (6),$$

$$N_j,\ B_j \geq 1 \quad \text{and integer}, \quad j = 1, \ldots, j.$$

where Constraints (1) to (4) are the MVA equations that govern the performance of the system, Equation (5) is the demand constraint developed based on Little's law and Equation (6) is a batch size constraint based on practical considerations. The remaining constraints are non-negativity and integer constraints. As it is a discrete manufacturing system, there is an integrality constraint on the batch size for a product. The number of batches or pallets allowed into the system as per the CONWIP protocol also has to be an integer.

4. The optimal algorithm

As discussed earlier, even for a given $\mathbf{N}$, the solution to the system of Equations (1) to (4) suffers from the curse of dimensionality. Furthermore, problem (P-OPT) is a very complex, non linear, integer programming problem. Even to obtain a lower bound to this problem requires the solution of a model with a non-convex objective function (with respect to $\mathbf{N}$), thus potentially resulting in multiple local optima. Therefore, developing an effective algorithm to solve (P-OPT) to optimality presents a significant challenge.

To deal with this challenge, we note at the outset that, for a given $\mathbf{N}$, we can determine the smallest value of $\mathbf{B}$ that satisfies Constraints (1) to (6). Therefore, the objective

function Z can be expressed as just a function of $\mathbf{N}$. Thus, we first obtain an expression for the lower bound on $Z(\mathbf{N})$. (An upper bound on $Z(\mathbf{N})$ is readily obtained from a feasible solution.) The expression for the lower bound on $Z(\mathbf{N})$, together with an upper and lower bound on $Z(\mathbf{N})$, are next used to obtain a lower bound, $\mathbf{N}^{\mathrm{L}}$, and an upper bound, $\mathbf{N}^{\mathrm{U}}$, on $\mathbf{N}$. The optimal solution is now obtained by explicitly enumerating $Z(\mathbf{N})$ for all values of $\mathbf{N}^{\mathrm{L}} \le \mathbf{N} \le \mathbf{N}^{\mathrm{U}}$.

4.1. *Lower bound on the objective function*

The decision variables in (P-OPT) are $\mathbf{N} = \{N_j, j = 1, \ldots, J\}$ and $\mathbf{B} = \{B_j, j = 1, \ldots, J\}$. If the integer constraint on B_j is relaxed, we get a relaxed version of the problem. From Equation (5) and the fact that we are minimizing the sum $\sum_{j=1}^{J} K_j N_j B_j$, it follows that $N_j B_j = D_j CT_j(\mathbf{N}, \mathbf{B}), \forall j$, when B_j is allowed to be a non-integer. The implication is that a larger $CT_j(\mathbf{N}, \mathbf{B})$ results in a larger value of $N_j B_j$ and, therefore, a larger value for the objective function of the relaxed problem. Similarly, a lower $CT_j(\mathbf{N}, \mathbf{B})$ implies a smaller WIP inventory value. Hence, a lower bound on $CT_j(\mathbf{N}, \mathbf{B})$ generates a lower bound on the objective function value for the relaxed problem as well as the original problem, (P-OPT).

Let $\bar{S}_j = \sum_{m=1}^{M} S_{mj}$, $\bar{P}_j = \sum_{m=1}^{M} P_{mj}$, $\breve{S}_j = \min_m\{S_{mj}\}$ and $\breve{P}_j = \min_m\{P_{mj}\}$. Then from Equations (2) and (1):

$$\begin{aligned} CT_j(\mathbf{N},\ \mathbf{B}) &\ge \sum_{m=1}^{M} ((S_{mj} + P_{mj} B_j) + (\breve{S}_j + P_j B_j)\, Q_m(\mathbf{N} - \mathbf{e}_j, \mathbf{B})) \quad (7)\\ &= \bar{S}_j + \bar{P}_j B_j + (\breve{S}_j + \breve{P}_j B_j)(NTOT - 1). \end{aligned}$$

From Equations (5) and (7):

$$K_j\ N_j\ B_j \ge K_j D_j((\bar{S}_j + \bar{P}_j B_j) + (\breve{S}_j + \breve{P}_j B_j)(NTOT - 1)). \quad (8)$$

The right-hand side of Equation (8) is a lower bound on $K_j N_j B_j$, that effectively incorporates Constraints (1) to (5) of (P-OPT). (Constraints (1), (2) and (5) are explicitly taken into account in deriving Equation (8). Constraints (3) and (4) become irrelevant since $Q_m(\mathbf{N} - \mathbf{e}_j, \mathbf{B})$ appears in Equation (8) only through its summation over all workstations, m.) Thus, ignoring Constraint (6), a lower bound on the objective function of the original problem is obtained if we can solve the following problem:

$$\begin{aligned} \text{(P-LB-1)}: \min \quad & \breve{Z}(\mathbf{N}) \sum_{j=1}^{j} K_j N_j B_j, \\ & \text{subject to Constraints (8)}, \\ & N_j \ge 1 \text{ and integer} \quad j = 1, \ldots, J. \end{aligned}$$

For a particular value of $\mathbf{N}$, clearly, the objective function of (P-LB-1) is minimized by having the lowest possible value of B_j. Therefore, for a given $\mathbf{N}$, $B_j(\mathbf{N})$ is calculated by solving Equation (8) as an equality and that results in the following expression:

$$B_j(\mathbf{N}) = \frac{\bar{S}_j + \breve{S}_j N_j + \breve{S}_j\left(\sum_{k\ne j} N_k - 1\right)}{(N_j/D_j) - \bar{P}_j - N_j P_j - P_j(\sum_{k\ne j} N_k - 1)} \quad (9)$$

$$\text{Define} \quad \alpha_j = K_j(\bar{S}_j - \breve{S}_j),\ \beta_j = K_j \breve{S}_j,$$
$$\gamma_j = \bar{P}_j - \breve{P}_j,\ \delta_j = \breve{P}_j,\ \rho_j = (1/D_j) - \breve{P}_j, \quad (10)$$

and let

$$z_j(\mathbf{N}) = K_j N_j B_j(N). \quad (11)$$

From Equations (9) and (10), and ignoring Constraint (6), we get

$$z_j(\mathbf{N}) = \frac{\alpha_j + \beta_j N_j + \beta_j \sum_{k\ne j} N_k}{\rho_j - (\gamma_j/N_j) - (\delta_j/N_j)\sum_{k\ne j} N_k}. \quad (12)$$

Note that the objective function of problem (P-LB-1) is now the sum $\sum_{j=1}^{J} z_j(\mathbf{N})$. Ignoring setup times in Equation (8), and noting that $B_j(\mathbf{N}) \ge 0$, we derive the following feasibility condition that should be satisfied by a solution to the problem (P-LB-1) as well as problem (P-OPT):

$$N_j \ge D_j(\bar{P}_j + \breve{P}_j(NTOT - 1)).$$

Using Equation (10), and noting that $NTOT$ also includes N_j, the above feasibility condition is written as

$$N_j \ge \left(\gamma_j + \delta_j \sum_{k\ne j} N_k\right)/\rho_j, \quad j = 1, \ldots.., J. \quad (13)$$

Property 1. *For a fixed N_k, $k \ne j$, $z_j(\mathbf{N})$ in Equation (12) is convex with respect to N_j for all values of N_j that satisfy the feasibility condition Equation (13). (See online supplement: Appendix A for a proof.)*

While $z_j(\mathbf{N})$ is convex with respect to N_j for fixed N_k, $k \ne j$, it is not jointly convex with respect to $\mathbf{N} = \{N_1, N_2, \ldots.., N_J\}$. Even if feasibility condition (13) is satisfied for a particular $\mathbf{N} = \mathbf{N}^{\mathrm{a}}$, it is not necessary that it is satisfied for all $\mathbf{N} > \mathbf{N}^{\mathrm{a}}$. To see this, let $\mathbf{N}^{\mathrm{b}} > \mathbf{N}^{\mathrm{a}}$ with $N_j^{\mathbf{b}} = N_j^{\mathrm{a}}$ and $N_k^{\mathrm{b}} > N_k^{\mathrm{a}}$, for $k \ne j$. For this case it is possible that $(\gamma_j + \delta_j \sum_{k\ne j} N_k^{\mathrm{a}})/\rho_j < N_j^{\mathrm{a}}$ (which satisfies the feasibility condition) but $N_j^{b} < (\gamma_j + \delta_j \sum_{k\ne j} N_k^{\mathrm{b}})/\rho_j$ (which violates the feasibility condition). As we search across $\mathbf{N}$, if feasibility is lost, attempts to recover feasibility by increasing N_j can result in high $z_j(\mathbf{N})$ values (the denominator in Equation (12) for $z_j(\mathbf{N})$ becomes very small). Therefore, there may be multiple local minima for the lower bounding problem (P-LB-1) as well as the original problem (P-OPT). Hence, to determine the optimal solution to problem (P-OPT), we have to evaluate $Z(\mathbf{N})$ for all possible values of $\mathbf{N}$. To that end, we develop upper and lower bounds on $\mathbf{N}$ to restrict the enumeration.

While the lower bound function $\breve{Z}(\mathbf{N})$ is not jointly convex with respect to $\mathbf{N}$, we know that $z_j(N_j)$ is convex. Also, as we will see later, a lower bound on $z_k(N_j)$, $k \ne j$, can

be expressed as a linear function of N_j. Therefore, one can derive a lower bound expression that is a convex function of just N_j. Using this lower bound expression, one can derive bounds on N_j, $j = 1, \ldots, J$, provided we have an actual numerical value of the lower bound, Z^{L}, and upper bound, Z^{U}, on $Z(\mathbf{N})$. An upper bound, Z^{U}, is obtained using a heuristic solution to the problem. However, since $z_j(\mathbf{N})$ is not jointly convex with respect to $\mathbf{N}$, an actual value for the lower bound, Z^{L}, can be obtained only by relaxing $z_j(\mathbf{N})$ further. Clearly, the right hand side of Equation (12) has its lowest value, when $N_k = 1$, for all $k \neq j$. Setting $N_k = 1$, for all $k \neq j$ in Equation (9):

$$B_j(N_j) = \frac{\bar{S}_j + S_j N_j + \breve{S}_j(J-2)}{(N_j/D_j) - \bar{P}_j - N_j\breve{P}_j - \breve{P}_j(J-2)}, \tag{14}$$

and so

$$z_j(\mathbf{N}) \geq z_j^1(N_j) = \frac{\alpha_j + \beta_j N_j + \beta_j(J-1)}{\rho_j - (\gamma_j/N_j) - (\delta_j(J-1)/N_j)} = \frac{\omega_j + \beta_j N_j}{\rho_j - (\phi_j/N_j)}, \tag{15}$$

where $\omega_j = \alpha_j + \beta_j(J-1)$, and $\phi_j = \gamma_j + \delta_j(J-1)$ It can be shown that for both Equations (9) and (14), $\partial B_j/\partial N_j \leq 0$. Therefore, as N_j increases, B_j given by Equation (9) or Equation (14), decreases and *vice versa*. Let N_{HI}^j and N_{LO}^j be the value of N_j for which $B_j(N_j)$ given by Equation (14) is B_{HI} and B_{LO} respectively[1], i.e., $B_j(N_{\mathrm{HI}}^j) = B_{\mathrm{HI}}$, and $B_j(N_{\mathrm{LO}}^j) = B_{\mathrm{LO}}$. Note that $N_{\mathrm{HI}}^j \leq N_{\mathrm{LO}}^j$. Constraint (6) can then be written in terms of N_j as shown in Equation (17). Therefore, another (possibly weaker, but solvable) lower bound for problem (P-OPT) is obtained by solving the following problem for each product j:

$$\text{(P-LB-2):}\quad \min\ z_j^j(N_j) = \frac{\omega_j + \beta_j N_j}{\rho_j - (\phi_j/N_j)}, \tag{16}$$

$$\text{subject to:}\quad N_{\mathrm{HI}}^j \leq N_j \leq N_{\mathrm{LO}}^j,\quad N_j \geq 1 \quad \text{and integer.} \tag{17}$$

In problem (P-LB-1), each z_j is expressed as a function of $\mathbf{N}$, but Constraint (6) is ignored. While we do not solve (P-LB-1), the expression for $z_j(\mathbf{N})$ is used in the determination of the bounds on the optimal value of $\mathbf{N}$. In problem (P-LB-2), the objective function is relaxed, as it is now only a function of N_j, but Constraint (6) is incorporated in the form of Equation (17). Since $z_j(N_j)$ is convex, the N_j that minimizes $z_j^1(N_j)$ is obtained by solving the first-order condition for Equation (16). For high-variety low-volume manufacturing systems the setup time and processing time matrices can be sparse and a product may not be processed at all workstations. Thus, it is possible that $\bar{S}_j$ and $\bar{P}_j$ are equal to zero, i.e., $\beta_j = 0$ and $\delta_j = 0$. If $\beta_j = 0$, then $z_j^1(N_j)$ is a linear function with a negative slope and the optimal solution to (P-LB-2) has $\tilde{N}_j = N_{\mathrm{LO}}^j$. Hence, the solution to (P-LB-2) before considering Constraint (17) is

$$\bar{N}_j = \begin{cases} N_{\mathrm{LO}}^j & \text{if} \beta_j = 0, \\ \left(\dfrac{1}{2\rho_j\beta_j}\right)\left(2\phi_j\beta_j + \sqrt{(2\phi_j\beta_j)^2 + 4\phi_j\omega_j\rho_j\beta_j}\right) & \text{otherwise.} \end{cases} \tag{18}$$

As N_j is required to be integer, the solution to (P-LB-2), ignoring Constraint (17), is

$$\hat{N}_j = \arg\min(z_j(\lfloor \bar{N}_j \rfloor), z_j(\lceil \bar{N}_j \rceil)). \tag{19}$$

Taking into account constraint (17), the solution to (P-LB-2) is

$$\tilde{N}_j = \begin{cases} N_{\mathrm{LO}}^j & \text{if} \quad \hat{N}_j > N_{\mathrm{LO}}^j, \\ N_{\mathrm{HI}}^j & \text{if} \quad \hat{N}_j < N_{\mathrm{HI}}^j, \\ \hat{N}_j & \text{if} \quad N_{\mathrm{HI}}^j \leq \hat{N}_j \leq N_{\mathrm{LO}}^j. \end{cases} \tag{20}$$

The corresponding lower bound on the objective function value, denoted by Z^{L} is thus

$$Z^{\mathrm{L}} = \sum_{j=1}^{J} z_j^1(\tilde{N}_j) = \sum_{j=1}^{J} \frac{\omega_j + \beta_j \tilde{N}_j}{\rho_j - (\phi_j/N_j)}. \tag{21}$$

4.2. *Lower and upper bound on the optimal number of pallets, N*

As discussed earlier, since the objective function $Z(\mathbf{N})$ is not jointly convex with respect to all the components $\{N_j, j = 1,\ldots, J\}$ of $\mathbf{N}$, the optimal solution to (P-OPT) has to be obtained by exhaustive enumeration. Therefore, to reduce the search space we develop an upper bound, $\mathbf{N}^{\mathrm{U}}$ and a lower bound, $\mathbf{N}^{\mathrm{L}}$, on the optimal value of $\mathbf{N}$. Note that the complexity of the search is exponential in the number of products, J. For example, if $\mathbf{N}^{\mathrm{L}} = \{1, 1, 1, \ldots, 1\}$ and $\mathbf{N}^{\mathrm{U}} = \{u, u, u, \ldots, u\}$, the total number of $\mathbf{N}$ for which $Z(\mathbf{N})$ has to be evaluated is u^J. Hence, it is important to develop as tight a bound (i.e., as low a $\mathbf{N}^{\mathrm{U}}$ and as high a $\mathbf{N}^{\mathrm{L}}$) as possible.

We discuss several methods to develop lower and upper bounds, N_j^{L}, and N_j^{U}, $j = 1, \ldots, J$, as well as bounds on $NTOT = \sum_{j=1}^{J} N_j$, in the optimal solution. All these bounds rely on an upper bound, Z^{U}, on $Z(\mathbf{N})$. (The next section describes a simple heuristic to obtain an initial value for Z^{U}). For a given Z^{U} we first develop the upper bound on the optimal N_j, $j = 1, \ldots, J$. We know Z^{L}, the lower bound on $Z(\mathbf{N})$ from Equation (21). Since $Z^{\mathrm{L}} = \sum_{j=1}^{J} z_j^1(\tilde{N}_j)$, we can use the expression, $Z^{\mathrm{L}} - z_j^1(\tilde{N}_j) + z_j^1(N_j) \leq Z^{\mathrm{U}}$, to find upper and lower bounds on N_j. However, this expression only accounts for

[1] For ease of exposition, we assume that N_{HI}^j and N_{LO}^j are integers. If they are not integers, the analysis can be suitably modified by fixing B_j to its binding value (i.e., B_{LO} or B_{HI} as the case may be) and solving for N_j.

the increase in $z_j^1(N_j)$ as N_j increases. Actually, as N_j increases, $z_k(\mathbf{N})$, $k \neq j$, also increases since $z_k(\mathbf{N})$ depends on N_j as per Equation (12). We use this information to obtain the bounds on N_j. For a fixed N_k, $k \neq j$:

$$\begin{aligned} z_k(N_j) &= \frac{\alpha_k + \beta_k N_k + \beta_k \sum_{i \neq k} N_i}{\rho_k - (\gamma_k/N_k) - (\delta_k/N_k)\sum_{i \neq k} N_i} \\ &\geq \frac{\alpha_k + \beta_k N_k + \beta_k(J-2) + \beta_k N_j}{\rho_k - (\gamma_k/N_k) - (\delta_k/N_k)(J-2) - (\delta_k N_j/N_k)} \\ &\geq \frac{\alpha_k + \beta_k N_k + \beta_k(J-1) - \beta_k + \beta_k N_j}{\rho_k - (1/N_k)(\phi_k - \delta_k + \delta_k N_j)} \\ &\geq z_k^1(\tilde{N}_k) + \frac{\beta_k N_j - \beta_k}{\rho_k - (\phi_k/N_k)}. \end{aligned} \tag{22}$$

Therefore, an upper bound on N_j can be obtained by solving the following expression:

$$Z^{\mathrm{U}} \geq z_j^1(N_j) + \sum_{k \neq j} z_k(N_j). \tag{23}$$

Substituting Equations (16) and (22) into Equation (23):

$$Z^{\mathrm{U}} \geq \frac{\omega_j + \beta_j N_j}{\rho_j - (\phi_j/N_j)} + \sum_{k \neq j} \left(z_k^1(\tilde{N}_k) + \frac{\beta_k N_j - \beta_k}{\rho_k - (\phi_k/N_k)} \right).$$

Rewriting the above, we get

$$\psi_j^{\mathrm{U}} \geq \frac{\omega_j + \beta_j N_j}{\rho_j - (\phi_j/N_j)} + \sum_{k \neq j} \frac{\beta_k N_j - \beta_k}{\rho_k - \left(\phi_k/N_k^{\mathrm{U}}\right)}, \tag{24}$$

where $\psi_j^{\mathrm{U}} = Z^{\mathrm{U}} - (Z^{\mathrm{L}}(N_j) - z_j^1(\tilde{N}_j))$. An upper bound, N_j^{U}, and lower bound, N_j^{L}, on the optimal N_j is now found by solving Equation (24) as an equality, resulting in the following quadratic equation for N_j:

$$a N_j^2 + b N_j + c = 0,$$

where

$$\begin{aligned} a &= \beta_j + \rho_j \sum_{k \neq j} \frac{\beta_k}{\rho_k - \left(\phi_k/N_k^{\mathrm{U}}\right)}, \\ b &= \omega_j - \rho_j \sum_{k \neq j} \frac{\beta_k}{\rho_k - (\phi_k/N_k^{\mathrm{U}})} \\ &\quad - \psi_j^{\mathrm{U}} \rho_j - \phi_j \sum_{k \neq j} \frac{\beta_k}{\rho_k - \left(\phi_k/N_k^{\mathrm{U}}\right)} \end{aligned}$$

and

$$c = \phi_j \left(\psi_j^{\mathrm{U}} + \sum_{k \neq j} \frac{\beta_k}{\rho_k - \left(\phi_k/N_k^{\mathrm{U}}\right)} \right).$$

Solving the above quadratic equation, we get

$$\left.\begin{aligned} N_j^{\mathrm{U},1} &= \left\lceil \frac{1}{2a}(-b + \sqrt{b^2 - 4ac}) \right\rceil \\ N_j^{\mathrm{L},1} &= \max\left\{ \left\lfloor \frac{1}{2a}\left(-b - \sqrt{b^2 - 4ac}\right) \right\rfloor, 1 \right\} \end{aligned}\right\}. \tag{25}$$

Note that the bounds given in Equation (25) depend on N_k^{U}, $k \neq j$, the upper bounds on the optimal number of pallets for the other products. As N_k^{U} increases, the linear component of the quadratic expression (24) (which is convex) increases, and hence the bounds given by Equation (25) will improve. Therefore, $N_j^{\mathrm{U},1}$ and $N_j^{\mathrm{L},1}$, $j = 1,\ldots, J$, is determined iteratively. A second upper bound $N_j^{\mathrm{U},2}$ is obtained by noting that $z_j^1(N_j) \geq K_j N_j B_{\mathrm{LO}}$. Therefore, substituting the first term on the right-hand side of Equation (24) with $K_j N_j B_{\mathrm{LO}}$, we get

$$N_j^{\mathrm{U},2} = \frac{\psi_j^{\mathrm{U}} + \left(\sum_{k \neq j} \beta_k/\left(\rho_k - (\phi_k/N_k^{\mathrm{U}})\right)\right)}{K_j B_{\mathrm{LO}} + \left(\sum_{k \neq j} \beta_k/\left(\rho_k - \phi_k/N_k^{\mathrm{U}}\right)\right)}. \tag{26}$$

Similar to $N_j^{\mathrm{U},1}$, the upper bound $N_j^{\mathrm{U},2}$ depends on the other upper bounds, N_k^{U}, $k \neq j$, and is thus improved iteratively. After deriving the two upper bounds on N_j, we choose the best one:

$$N_j^{\mathrm{U}} = \min\left\{ N_j^{\mathrm{U},1},\ N_j^{\mathrm{U},2} \right\}, \quad j = 1, \ldots, J. \tag{27}$$

A second lower bound on N_j is obtained using Equation (8) and some algebraic manipulations, to get

$$\begin{aligned} N_j^{\mathrm{L},2} = D_j \Bigg(&\left(\bar{P}_j + \breve{P}_j \left(\sum_k N_k^{\mathrm{L}} - 1 \right) \right) \\ &+ \frac{\left(\bar{S}_j + \breve{S}_j \left(\sum_k N_k^{\mathrm{L}} - 1 \right) \right)}{B_{\mathrm{HI}}} \Bigg). \end{aligned} \tag{28}$$

Similar to the expressions for the upper bound, $N_j^{\mathrm{L},2}$ depends on prior lower bounds, N_k^{L}, for all products k, and so improved lower bounds are obtained iteratively. The best lower bound is thus

$$N_j^{\mathrm{L}} = \max\left\{ N_j^{\mathrm{L},1},\ N_j^{\mathrm{L},2} \right\}, \quad j = 1, \ldots, J. \tag{29}$$

Apart from the bounds on the optimal number of pallets for each product, we also obtain upper bounds on $NTOT$, the total number of pallets to further reduce the number of evaluations of the objective function. For example, if a particular value of $\mathbf{N}$ is within the bounds, we may be required to evaluate the corresponding value of the WIP inventory, $Z(\mathbf{N})$. However, it is possible that $\mathbf{N}$ is such that $\sum_j N_j$ exceeds this upper bound on the total number of pallets. In that case we do not evaluate $Z(\mathbf{N})$ for this particular value of $\mathbf{N}$. Thus, an upper bound on $NTOT$ helps reduce the number of evaluations of $Z(\mathbf{N})$ in the optimal algorithm.

Since $B_j \geq B_{\mathrm{LO}}$, we have $Z^{\mathrm{U}} \geq Z(\mathbf{N}) \geq \sum_j K_j N_j B_{\mathrm{LO}} \geq \sum_j \breve{K} N_j B_{\mathrm{LO}} = \breve{K} B_{\mathrm{LO}} NTOT$, where $\breve{K} = \min_j\{K_j\}$. This gives an upper bound on $NTOT$ as

$$NTOT^{\mathrm{U},1} = Z^{\mathrm{U}}/\breve{K} B_{\mathrm{LO}}.$$

Another upper bound on $NTOT$ is obtained as follows. Equation From (12):

$$z_j(\mathbf{N}) = \frac{\alpha_j + \beta_j NTOT}{\rho_j - (\gamma_j/N_j) - (\delta_j/N_j)\sum_{k\neq j} N_k} \geq \frac{\alpha_j + \beta_j NTOT}{\rho_j - (\gamma_j/N_j) - (\delta_j(J-1)/N_j)}.$$

Therefore

$$Z^{\mathrm{U}} \geq \sum_{j=1}^{J} z_j(\mathbf{N}) \geq \sum_{j=1}^{J} \frac{\alpha_j + \beta_j NTOT}{\rho_j - (\phi_j/N_j)} \geq \sum_{j=1}^{J} \frac{\alpha_j + \beta_j NTOT}{\rho_j - (\phi_j/N_j^{\mathrm{U}})}. \quad (30)$$

From Equation (30), we get

$$NTOT^{\mathrm{U},2} = \frac{Z^{\mathrm{U}} - \sum_{j=1}^{J}\left(\alpha_j/\left(\rho_j - (\phi_j/N_j^{\mathrm{U}})\right)\right)}{\sum_{j=1}^{J}\left(\beta_j/\left(\rho_j - (\phi_j/N_j^{\mathrm{U}})\right)\right)}. \quad (31)$$

The upper bound on $NTOT$ is thus

$$NTOT^{\mathrm{U}} = \min\{NTOT^{\mathrm{U},1}, NTOT^{\mathrm{U},2}\}. \quad (32)$$

In addition to the above bounds, the number of objective function evaluations is further restricted through dynamic bounds. One such dynamic bound is the upper bound on $NTOT$. For a given $\mathbf{N}$, if $\sum_j N_j > NTOT^{\mathrm{U}}$, there is no need to evaluate $Z(\mathbf{N})$. There are two other dynamic bounds used in the enumeration algorithm. Similar to the lower bound in Equation (28), from Equation (8):

$$N_j \geq D_j\left((\bar{P}_j + \breve{P}_j(NTOT(\mathbf{N})-1)) + \frac{\left(\bar{S}_j + \breve{S}_j(NTOT(\mathbf{N})-1)\right)}{B_{\mathrm{HI}}}\right). \quad (33)$$

Therefore, for a particular decision vector, $\mathbf{N}$, there is no need to evaluate $Z(\mathbf{N})$ if

$$N_j < D_j\left((\bar{P}_j + \breve{P}_j(NTOT(\mathbf{N})-1)) + \frac{(\bar{S}_j + \breve{S}_j(NTOT(\mathbf{N})-1))}{B_{\mathrm{HI}}}\right).$$

Finally, for a given $\mathbf{N}$, $Z(\mathbf{N}) \geq \sum_j K_j N_j B_{\mathrm{LO}}$. Therefore, for a given $\mathbf{N}$, we do not evaluate $Z(\mathbf{N})$ if we find that $\sum_j K_j N_j B_{\mathrm{LO}} > Z^{\mathrm{U}}$.

4.3. *Optimal algorithm to determine the number of pallets and batch size for each product*

Once the upper and lower bounds on $\mathbf{N}$ and the dynamic bounds on $\mathbf{N}$ and $NTOT$ are determined, the optimal solution to (P-OPT) is determined by exhaustive enumeration of $Z(\mathbf{N})$ for all possible values of $\mathbf{N}$ within the bounds. For a given vector, $\mathbf{N}$, the corresponding batch size for each product $\mathbf{B} = B_j$, $j = 1,\ldots, J$ and $Z(\mathbf{N})$ are calculated by solving the system of Equations (1) to (6). The algorithm searches among all possible values of $\mathbf{N}$ to determine Z^*, the optimal value of $Z(\mathbf{N})$. Apart from restricting the enumeration between the lower and upper bounds of $\mathbf{N}$, the algorithm also employs the dynamic bounds on $NTOT$ and $\mathbf{N}$ to restrict the total number of enumerations. When an improved solution, Z, is found during the enumeration, it becomes the new upper bound, Z^{U} on the optimal Z^*, and the upper bounds on $\mathbf{N}$ and $NTOT$ are then recalculated. Our computational experience reveals that these dynamic bounds and recalculation of the upper bounds on $\mathbf{N}$ and $NTOT$ significantly reduce the total number of cost evaluations required compared to just using the initial upper and lower bounds on $\mathbf{N}$. The steps of the enumeration based optimal algorithm are formally stated in the Appendix.

4.4. *Determining the batch sizes and value of the WIP inventory for a given* $\mathbf{N}$

The optimal enumeration algorithm, as well as the heuristic presented in the next section, evaluates the batch size $\mathbf{B}(\mathbf{N})$ and $Z(\mathbf{N})$ for every given $\mathbf{N}$. For a given $\mathbf{N}$, the behavior of the manufacturing system is governed by the MVA Equations (1), (2), (3) and (4). The batch size $\mathbf{B}(\mathbf{N}) = \{B_j\}$ and the value of WIP inventory, $Z(\mathbf{N})$, are determined precisely from Equations (1) to (6). However, as discussed earlier, the exact solution for the MVA equations suffers from the curse of dimensionality and can be very time-consuming even for a single value of $\mathbf{N}$. As our optimal algorithm (as well as the heuristic) evaluates $Z(\mathbf{N})$ for a large number of $\mathbf{N}$, we resort to the Schweitzer–Bard approximation for the MVA equations. As per this approximation:

$$q_{mk}(\mathbf{N} - \mathbf{e}_j, \mathbf{B}) = \begin{cases} q_{mk}(\mathbf{N}, \mathbf{B}) & k \neq j, \\ q_{mk}(\mathbf{N}, \mathbf{B})\dfrac{(N_j - 1)}{N_j} & k = j. \end{cases} \quad (34)$$

Using the Schweitzer–Bard approximation, Equations (1) to (6) are solved iteratively to determine the optimal $\mathbf{B}$ for a fixed $\mathbf{N}$. The detailed procedure (PROC-B) to determine $\mathbf{B}(\mathbf{N})$ and $Z(\mathbf{N})$ for a given $\mathbf{N}$ is provided in the online supplement: Appendix B.

5. Heuristic algorithm

The optimal algorithm developed in Section 4 is based on exhaustive enumeration and so, despite the bounds on $\mathbf{N}$ and $NTOT$, the computational complexity increases exponentially with the number of products in the system. Therefore, a heuristic approach to solve problems with a large number of products is highly desirable. This section develops such an approach.

The initial value of $\mathbf{N}$ used in the heuristic is $\tilde{\mathbf{N}}$, the value corresponding to the lower bound obtained using Equation (20). The initial value of $\mathbf{N} = \tilde{\mathbf{N}}$ need not generate a feasible

solution. Feasibility could be violated, if the value of the batch size B_j obtained with the initial value of N_j is either negative or greater than B_{HI} for a particular product j, in which case N_j is increased until a feasible value of B_j is obtained. Thus, starting with $\tilde{\mathbf{N}}$, the N_j' values are increased if necessary (for products with infeasible values of the batc h size, B_j), until a feasible initial value of $\mathbf{N}$ is obtained. For a particular value of $\mathbf{N}$, the corresponding batch size vector, $\mathbf{B}(\mathbf{N})$, and the objective function value $Z(\mathbf{N})$, is obtained using procedure (PROC-B) explained in Section 4.4.

The search proceeds by incrementing or decreasing the components of $\mathbf{N}$, N_j, one at a time. That is, we evaluate a new $\mathbf{N} = \mathbf{N} = \mathbf{N} \pm \mathbf{e}_j$ for each j. If the new value of $\mathbf{N}$ decreases $Z(\mathbf{N})$, the solution is updated. If no improvement in the solution is obtained after a round that evaluates all the J directions of change, we stop the search and use the currently available best solution.

It is possible to increase (or decrease) the value of more than one component of $\mathbf{N}$ at a time. This approach would significantly increase the complexity of the heuristic, as the number of search directions increases from J to almost $J^2/2$. However, searching only along the unit vectors $\mathbf{e}_j$ in step sizes of ± 1 might result in not getting good solutions, as $Z(\mathbf{N})$ may have many local optima. Therefore, we search only along all unit vector directions $\mathbf{e}_j$ for incremental changes in $\mathbf{N}$, but consider step sizes of ± 1, $\pm$ 2and $\pm$ 3. The formal statement of the heuristic is given in online supplement: Appendix C.

6. Numerical example and computational results

To illustrate the performance of the optimal algorithm and the heuristic we first discuss two numerical examples. The data for the first example is given in Table 1. For this example, $Z^L = \$4241$, and the corresponding $\tilde{\mathbf{N}} = \{1, 2, 2, 1\}$. The heuristic as well as the optimal solution gives the value of the WIP inventory as $Z^* = Z^H = \$5300$, and the corresponding $\mathbf{N}^* = \{1, 3, 2, 1\}$. The batch sizes corresponding to this solution $\mathbf{B}^* = \{3, 9, 10, 3\}$. The heuristic solution required 21 cost evaluations. Using only the initial lower and upper bound on $\mathbf{N}$, 805 evaluations would have been necessary. However, the dynamic bounds and dynamic recalculation of the upper bounds on $\mathbf{N}$ and *NTOT* reduced the actual number of enumerations required for the optimal solution to just 186. In this example, the heuristic obtained the optimal solution.

Table 1. Data for numerical example 1 $J = 4$, $M = 2$, $D_1 = D_4 = 0.5$, $D_2 = D_3 = 5$, $K_1 = K_2 = K_3 = K_4 = 100$, $B_{LO} = 1$, $B_{HI} = 10$

	Setup times				*Processing time*			
m	*j = 1*	*j = 2*	*j = 3*	*j = 4*	*j = 1*	*j = 2*	*j = 3*	*j = 4*
1	0.5	0.5	0.5	0.5	0.001	0.004	0.005	0.001
2	0.6	0.7	0.3	0.3	0.003	0.001	0.004	0.004
3	0.3	0.3	0.3	0.8	0.003	0.005	0.005	0.003

Table 2. Data for numerical example 2 $J = 2$, $M = 15$, $D_1 = 2.20244$, $D_2 = 2.667455$, $K_1 = K_2 = \$1.0$, $B_{LO} = 1$, $B_{HI} = \infty$

	Setup times		*Processing times*	
	j = 1	*j = 2*	*j = 1*	*j = 2*
1	0	0.5	0	0.085
2	0	0.5	0	0.1
3	0.5	0	0.1	0
4	0	0.1	0	0.061
5	0	0.4	0	0.15
6	0	0	0.02	0.025
7	0.5	0	0.0388 89	0
8	0.25	0	0.143	0
9	0	0	0.005	0.005
10	0	0	0.01	0.01
11	0	0	0	0.196
12	0	0	0.03	0
13	0	2.5	0	0.001
14	2	0	0.001	0
15	0	0	0.005	0.005

Our second example is a partial extract of an industrial data set from Woodward Aircraft Engine Systems (Srinivasan *et al.*, 2003). This example contains two products and 15 workstations. The data for the example is given in Table 2. In addition to the processing time data given in Table 2, for this example, there is also a processing time or holding time outside the system, L_{0j}, that is independent of the batch size for each pallet. This processing time could be due to transportation and additional processing at a subcontractor. For this example, $L_{01} = L_{02} = 51.2$. We note that a processing time outside the system, L_{0j}, can be readily incorporated into our model. Essentially, the term L_{0j} is added to the right-hand side of Equation (2), and procedure (PROC-B) can be modified appropriately based on this approach. For this example, the value of the lower bound, $Z^L = \$20$, and the corresponding $\tilde{\mathbf{N}} = \{8, 12\}$. The value of the WIP inventory in the optimal solution $Z^* = \$327$, and $Z^H = \$332$. $\mathbf{N}^* = \{15, 16\}$ and $\mathbf{B}^* = \{9, 12\}$. $\mathbf{N}^H = \{10, 16\}$ and $\mathbf{B}^H = \{14, 12\}$. The heuristic solution required 35 cost evaluations. The total number of cost evaluations required for the optimal solution was 52, 987. The number of evaluations based on the initial bounds on $\mathbf{N}$ would have been 100 385. In this example, the objective function has more than one local optimum and our heuristic gives a solution inferior to the optimal solution.

We now provide the results of a computational study on randomly generated problems. The number of products in these problems varied from four to 20 (the specific values of J were 4, 5, 8, 10 and 20) and the number of workstations varied from four to 12 ($M = 4$, 8, 12). For all problems, the batch size constraints were set as $B_{LO} = 1$, and $B_{HI} = 10$. The demand, D_j, dollar value per unit, K_j,

Table 3. Computational results—random data set 1

Number of Products J	*Number of workstations* M	*Lower bound* (\$) Z^L	*Optimal WIP value* (\$) Z^*	*Heuristic solution* (\$) Z^H	*Number of cost evaluations*	
					Optimal algorithm	*Heuristic*
4	4	2 611	4 071	4 071	25	20
4	8	4 252	6 893	6 893	56	27
4	**12**	**5 953**	**9 994**	**10 064**	**137**	**30**
5	4	4 838	7 729	7 729	75	26
5	8	7 517	12 834	12 834	628	29
5	12	10 362	16 449	16 449	240	29
8	4	6 476	10 198	10 198	157	38
8	8	8 844	14 104	14 104	2943	39
8	**12**	**11 416**	**18 519**	**18 554**	**3 436**	**67**
10	4	7 935	12 879	12 879	8 826	48
10	8	10 382	15 864	15 864	3 025	47
10	12	13 026	21 732	21 732	41 072	52
20	4	14 742	22 388	22 388	1 035 308	89
20	8	17 152	26 251	26 251	3 392 428	90
20	12	19 853	30 570	30 570	12 730 660	91

setup times S_{mj} and processing times P_{mj} were generated randomly from uniform distributions. The values for K_j were randomly generated from U[\$90, \$110] and the values for S_{mj} and P_{mj} were randomly generated from U[0.45, 0.55], and U[0.002, 0.003] respectively. The demand, D_j, was randomly generated from U[8, 10] with a probability 0.1 and from U[0.8, 1.0] with probability 0.9. If the generated demands violated the machine production capacity constraint (evaluated assuming that all the processing times were equal to 0.003), then they were proportionately scaled down. For each combination of number of products, J, and number of workstations M, a total of three problems were randomly generated. The computational results reported in Table 3 for each (J, M) combination is the average value for the three problems. For each (J, M) combination, Table 3 reports the value of the lower bound, Z^{L}, the optimal WIP inventory Z^*, the WIP inventory value corresponding to the heuristic solution, Z^{H}, and the number of cost evaluations required for the optimal algorithm and the heuristic.

The results provided in Table 3 show that the number of cost evaluations required for the heuristic increases only linearly with the number of products, whereas for the optimal algorithm the number of cost evaluations increases very rapidly and exponentially as the problem size increases, and for problem sizes larger than 20, it is not possible to solve the problem optimally in reasonable computational time. For all except two (highlighted) cases, the local neighborhood search heuristic generated the optimal solution. However, this should not be construed as evidence that the heuristic is efficient. One reason why the heuristic performed well on this data set is that the variances in the demand, setup times and processing times across the products are relatively small, and the setup time and processing time matrices are dense (there is no j, m for which either S_{mj} or P_{mj} is zero). In reality, for high-variety low-volume manufacturing systems, the setup time and processing time matrices will be sparse and in this case cost of the heuristic solution can be much larger than the optimal solution. We give a third numerical example to illustrate this.

The data for the third numerical example is given in Table 4. For this example, the value of the lower bound, $Z^{\mathrm{L}} =$ \$17, 200, and the corresponding $\tilde{\mathbf{N}} = \{1, 1, 1\}$. $Z^* =$ \$37, 090, and $Z^{\mathrm{H}} =$ \$104, 001. $\mathbf{N}^* = \{1, 1, 90\}$ and $\mathbf{B}^* = \{3, 7, 1\}$. $\mathbf{N}^{\mathrm{H}} = \{2, 1, 1\}$ and $\mathbf{B}^{\mathrm{H}} = \{5, 4, 1\}$. The heuristic solution required 28 cost enumerations and optimal algorithm required 24, 598 enumerations. In the case the heuristic solution is about 180% more than the optimal solution!

We performed another computational study with data that had more variance and sparse setup time and processing time matrices compared to the first data set. For this data set, the data for the first J-2 products had less variance, and the setup time and processing time for these products were non-zero with low variance. The setup time and processing time data for the remaining two products

Table 4. Data for numerical example 3 $J = 3$, $M = 3$, $D_1 = 1$, $D_2 = D_3 = 0.05$, $K_1 =$ \$10 000, $K_2 =$ \$1 000, $K_3 =$ \$1, $B_{\mathrm{LO}} = 1$, $B_{\mathrm{HI}} = \infty$

	Setup times			*Processing times*		
	$j = 1$	$j = 2$	$j = 3$	$j = 1$	$j = 2$	$j = 3$
1	0.8	20	0	0.5	0.0002	0
2	0	1	1	0	0	0
3	0	1	1	0	0	0.0001

Table 5. Data specification for random data set 2

j, m		D_j	K_j	S_{mj}	P_{mj}
$j <= J-2$	$m < M$	From U[0.8, 1.2]	From U [\$8000, \$120 000]	FromU [0.7, 0.9]	From U [0.4, 0.6]
	$m = M$			0	0
$j = J-1$	$M < M-1$	From U [0.04, 0.05]	From U [\$8, \$12]	0	0
	$M = M-1$			From U [100, 200]	From U [0.0001, 0.0005]
	$m = M$			From U [1, 2]	0
$j = J$	$m < M$	From U [0.04, 0.05]	From U [\$0.8, \$1.2]	0	0
$j = J$	$m = M$			From U [1, 2]	0

were very sparse. The values of D_j, K_j, S_{mj} and P_{mj} were generated as per the details given in the Table 5.

The computational study for this data set was carried for $J = 3$, 4, 6, 8 and 10 and $M = 3$, 4, 6 and 10 respectively. The batch size limits for all problems were $B_{\mathrm{LO}} = 5$, and $B_{\mathrm{HI}} = 1000$. As the total number of enumerations required by the optimal algorithm for the bigger problems in this data set was very large, we did not perform complete enumeration in the optimal algorithm. Instead, we stopped the algorithm when either the cost of the solution was less than 95% of the cost of the heuristic or when the number of cost evaluations reached 100 000. Clearly, solutions using the algorithm need not be optimal for these cases. Computational results for this data set (provided in Table 6) show that for 17 out of 20 cases, the best solution is much better than the heuristic solution. In the remaining three cases, the optimal algorithm was not able to find a better solution than the heuristic even after 100 000 cost evaluations. Even though the data set we generated might be an extreme scenario, these results suggest that for high-variety low-volume manufacturing systems with sparse setup time and processing time matrices, as often observed in practice, the local neighborhood search heuristic need not always be very efficient. However, the optimal algorithm may take a prohibitively long computational time for problem sizes bigger than ten or 20 products. The bounds we develop on **N** and *NTOT* can be used to create an efficient heuristic based on evolutionary optimization.

Table 6. Computational results—random data set 2

Number of products J	*Number of workstations* M	*Best WIP value (\$)* Z^{Best}	*Heuristic solution (\$)* Z^{H}	*Number of cost evaluations*	
				Optimal algorithm	*Heuristic*
3	3	146 954	189 626	831	32
3	4	240 217	265 212	1 236	36
3	6	377 823	401 673	4 033	38
3	10	856 729	880 579	20 807	45
4	3	293 885	312 448	826	22
4	4	421 207	440 488	47 596	67
4	6	678 417	832 505	851	26
4	10	1 954 604	3 309 442	598	52
6	3	294 645	332 953	125	26
6	4	564 798	564 798	100 000	97
6	6	804 251	853 391	453	32
6	10	1 439 671	1 566 494	10 697	72
8	3	353 484	361 534	100 000	34
8	4	529 050	570 245	1 312	114
8	6	710 015	710 015	100 000	44
8	10	1 346 089	1 728 983	179	82
10	3	454 977	454 977	100 000	42
10	4	489 939	498 507	100 000	42
10	6	696 246	756 098	141	42

7. Summary and conclusions

This paper presents an optimal algorithm to determine the number of pallets and pallet sizes (batch sizes) in a CONWIP-controlled high-variety low-volume manufacturing system. Unlike earlier papers using CQN models, this paper considers non-zero setup times, and treats both the number of pallets and pallet sizes as decision variables. The objective is to minimize the WIP inventory value (indirectly, minimize cycle times) while meeting the annual demand for the products. The objective function is not convex and may have several local minima. Using bounds on the WIP inventory value and on the number of pallets for each product, an enumeration-based optimal algorithm is developed using MVA.

A heuristic based on neighborhood search is provided for further reducing the number of candidate configurations evaluated in the search for an optimal solution. The heuristic appears to be efficient for problems with dense setup and processing time matrices and low variance in data across products, but for some high-variety low-volume manufacturing systems, the heuristic solution can be much larger than the optimal. However, the bounds on the optimal number of pallets for each product may be used to design heuristics based on evolutionary optimization.

The solution methodology presented in this paper will help manufacturing facilities such as Woodward set their WIP inventory levels more accurately. The methodology can readily be extended to handle manufacturing systems with multiple machines at each workstation.

There are at least two ways in which the methodology can be extended further, presenting opportunities for future research. One way is to model machine breakdowns. In practice, breakdowns are handled in two ways: continue running the other machines (while the broken machine is being repaired) until the downstream machines begin to starve for parts, or put into operation another standby machine. Woodward, for instance, was adopting both approaches as it saw fit. Modeling breakdowns presents an interesting and a challenging opportunity since it is not clear whether the product form assumption will give robust results any more.

Another way the research could be extended is to model finite buffers in front of each machine. Finite buffer models have been extensively studied for CQNs with a single product; however, these models do not readily extend to handle problems involving multiple products. Since the product form assumption is clearly violated with finite buffer models, some simplifying assumptions need to be made and carefully validated, either through an actual implementation or using simulation.

References

Askin, R.G., Bethmangalkar, G. and Szidarovszky, F. (2006) Minimal work-in-process requirements for satisfying demand in multi-product flexible manufacturing systems. Working paper.

Bard, Y. (1979) Some extensions to multiclass queueing network analysis, in *Performance of Computer Systems*, Arato, M. (ed.) North Holland, Amsterdam, The Netherlands.

Baskett, F., Chandy, K.M. Muntz, R.R. and Palacios-Gomez, F. (1975) Open, closed, and mixed networks of queues with different classes of customers. *Journal of the ACM*, **22**, 248–260.

Benjaafar, S. (1996) On production batches, transfer batches, and lead times. *IIE Transactions*, **28**, 357–362.

Billington, P.J., McClain, J.O. and Thomas, L.J. (1983) Mathematical programming approaches to capacity-constrained MRP systems: review, formulation and problem reduction. *Management Science*, **29**, 1126–1141.

Dar-El, E.M., Herer, Y.T. and Masin, M. (1999) CONWIP-based production lines with multiple bottlenecks: performance and design implications. *IIE Transactions*, **31**, 99–111.

Dessouky, M. (1998) Using queueing network models to set lot-sizing policies for printed circuit board assembly operations. *Production and Inventory Management*, **39**, 38–43.

Erenguc, S.S. and Tufekci, S. (1987) A branch and bound algorithm for a single-item multi-source dynamic lotsizing problem with capacity constraints. *IIE Transactions*, **19**, 73–80.

Golany, B., Dar-El, E.M. and Zev, N. (1999) Controlling shop floor operations in a multi-family, multi-cell manufacturing environment through constant work-in-process. *IIE Transactions*, **31**, 771–781.

Goldratt, E.M. and Fox, R. (1986) T*he Race*, North River Press, Croton-on-Hudson, NY.

Govil, M.K., and Fu, M.C. (1999) Queueing theory in manufacturing: A survey. *Journal of Manufacturing Systems*, **18**, 214–240.

Hill, A.V. and Raturi, A.S. (1992) A model for materials requirements planning systems. *Journal of the Operational Research Society*, **43**, 605–620.

Hopp, W. and Spearman, M. (2000) *Factory Physics*, Second edition, Irwin McGraw-Hill, New York, NY.

Karmarkar, U.S. (1987) Lot sizes, lead times and in-process inventories. *Management Science*, **33**, 409–418.

Karmarkar, U.S., Kekre, S. and Kekre. S. (1985) Lot sizing in multiItem multi-machine job shops. *IIE Transactions*, **17**, 290–292.

Lee, H.F., Srinivasan, M.M. and Yano. C.A. (2006) A framework for capacity planning and machine configuration in flexible assembly systems. *International Journal of Flexible Manufacturing Systems*, **18**, 239–268.

Maes, J. and Van Wassenhove, L.N. (1986) Multi item single level capacitated dynamic lotsizing heuristics: a computational comparison (part I: static case). *IIE Transactions*, **18**, 124–129.

Maxwell, W.L. and Muckstadt, J.A. (1985) Establishing consistent and realistic reorder intervals in production-distribution systems. *Operations Research*, **33**, 1316–1340.

Monden, Y. (1993) *Toyota Production System*, second edition, Institute of Industrial Engineers, Norcross, GA.

Ohno, T. (1988) *Toyota Production System: Beyond Large Scale Production*, Productivity Press, Cambridge, MA.

Reiser, M. and Lavenberg, S.S. (1980) Mean value analysis of closed multichain queueing networks. *Journal of the ACM*, **27**, 313–320.

Ryan, S.M., Baynat, B. and Choobineh, F.F. (2000) Determining inventory levels in a CONWIP controlled job shop. *IIE Transactions*, **32**, 105–114.

Ryan, S.M. and Choobineh, F.F. 2003. Total WIP and WIP mix for a CONWIP controlled job shop. *IIE Transactions*, **35**, 405–418.

Ryan, S.M. and Vorasayan, J. (2005) Allocating work in process in a multiple-product CONWIP system with lost sales. *International Journal of Production Research*, **43**, 223–246.

Schweitzer, P. (1979) Approximate analysis of multi-class closed networks of queues, in *Proceedings of the International Conference on Stochastic Control and Optimization*.

Solberg, J.J. (1980) CAN-Q User's Guide. NSF Grant number. APR 74 15256 IX, July 1980.

Spearman, M., Woodruff, D. and Hopp, W. (1990). CONWIP: A pull alternative to kanban. *International Journal of Production Research*, **28**, 879–894.

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

Srinivasan, M.M., Ebbing, S.J. and Swearingen, A.T. (2003). Woodward Aircraft Engine Systems sets work-in-process levels for high-variety, low volume products. *Interfaces*, **33**, 61–69.

Suri, R. (1985) Robustness of queueing network formulas. *Journal of the Association for Computing Machinery*, **30**, 564–594.

Suri, R. (1998) Q*uick Response Manufacturing*, Productivity Press, OR.

Suri, R., Shinde, R. and Vernon, M. (2005) Using fractional number of customers to achieve throughputs in queueing networks, in *Proceedings of the IERC*.

Appendix 1

Algorithm for determining the optimal solution to (P-OPT)

A: Determine lower bound

Step 1. For $\forall j = 1, ..., J$, determine the $\tilde{N}_j$, the optimal solution to the lower bounding problem using Equations (18), (19) and (20).

Step 2. Determine the lower bound to the optimal solution, Z^{L} using Equation (21).

B: Determine upper bound

Step 3. Determine an upper bound to the optimal solution, $Z^{\mathrm{U}} = Z(\tilde{\mathbf{N}})$, where $\tilde{\mathbf{N}} = \{\tilde{N}_j,\ j = 1, \ldots, J\}$. Alternatively, determine Z^{U} as the cost of the solution obtained using the heuristic described in Section 5.

C: Determine bounds on **N** and *NTOT*

Step 4. Determine $\mathbf{N}^{\mathrm{L}}$ and $\mathbf{N}^{\mathrm{U}}$ using Equations (29) and (27).

Step 5. Determine $NTOT^{\mathrm{U}}$ using Equation (32).

D: Enumeration of valid solutions

Step 6. Set $\mathbf{Z^*} = \mathbf{Z}^{\mathrm{U}}$, and **N*** as equal to the corresponding value of **N**.

Step 7. For all values of **N**, with $\mathbf{N}^{\mathrm{L}} \leq \mathbf{N} \leq \mathbf{N}^{U}$, do

{ If (($NTOT(\mathbf{N}) < NTOT^{U}$) & ($N_j$ satisfies Equation (33) for all j) & ($\sum_j K_j N_j B_{\mathrm{LO}} \leq Z^{\mathrm{U}}$)), then

{ evaluate **B**(**N**) and Z(**N**) using procedure (PROC-B).

If **N** does not give a feasible solution, set Z(**N**) = INFINITY.

if ($Z(\mathbf{N}) < Z*$), then set $Z* = Z(\mathbf{N})$, and $\mathbf{N}^* = \mathbf{N}$), re-evaluate bounds on **N** and NTOT using Steps 4 and 5.

}

}

Step 8. STOP. The optimal number of pallets is **N***, and the corresponding WIP inventory = Z^*.

Biographies

Mandyam M. Srinivasan ("Srini") is The Ball Corporation Distinguished Professor of Business at The University of Tennessee. He is the author of the book, *Streamlined: 14 Principles for Building and Managing the Lean Supply Chain*, and co-author of *Supply Chain Management for Competitive Advantage: Concepts and Cases*. His research interests focus on stochastic models for supply chain management and on creating flow in high-variety, low volume, production systems. His work has appeared in journals such as *Operations Research*, *Management Science*, *IIE Transactions*, *IEEE Transactions on Communications* and *Queueing Systems*.

S. Viswanathan (Vish) is Professor of Operations Management at the Nanyang Business School, Nanyang Technological University, SINGAPORE. His research interests include manufacturing systems, inventory management, joint replenishment inventory systems, supply chain coordination, reverse logistics models and hierarchical forecasting. His research has been published in journals such as *Management Science*, *Operations Research, Operations Research Letters*, *Naval Research Logistics*, *IIE Transactions*, *International Journal of Production Research*, *Journal of the Operational Research Society* and *European Journal of Operational Research*.