Order Sequencing and Capacity Balancing in Synchronous Manufacturing

Jan Riezebos

Order Sequencing and Capacity Balancing in Synchronous Manufacturing

Journal:	*International Journal of Production Research*
Manuscript ID:	TPRS-2008-IJPR-0957.R2
Manuscript Type:	Original Manuscript
Date Submitted by the Author:	01-May-2009
Complete List of Authors:	Riezebos, Jan; University of Groningen, Faculty of Economics and Business
Keywords:	SYNCHRONOUS MANUFACTURING, SEQUENCING, ORDER RELEASE, HEURISTICS, INTEGER PROGRAMMING
Keywords (user):	SYNCHRONOUS MANUFACTURING, SEQUENCING

Order Sequencing and Capacity Balancing in Synchronous Manufacturing

JAN RIEZEBOS*

Department of Operations, Faculty of Economics and Business, University of Groningen, The Netherlands

* Email: j.riezebos@rug.nl

Order Sequencing and Capacity Balancing in Synchronous Manufacturing

Abstract

Synchronous manufacturing aims at achieving the benefits of intermittent production lines in production situations that operate without lines. Benefits such as short and constant throughput times and predictable capacity loading can be acquired through an appropriate design of the synchronous manufacturing system and its control system. The order release mechanism is an essential part of this control system. It determines the sequence in which orders are released to the shop floor. As orders may differ in the amount and distribution of their capacity requirements over subsequent production stages, total capacity load may vary over time. If the available capacity per period is not flexible, capacity balancing becomes an issue in the order release decision. In practice, heuristics or rules of thumb are used to solve this problem, but their effectiveness is questioned. This paper examines the effectiveness of some new heuristics that are based on insights from assembly system design and work load control, and compare their performance with an optimal solution approach. The approaches are evaluated in a rolling schedule environment, and under different levels of capacity fluctuations and problem sizes. The results show that the performance of the heuristic solutions deteriorates if capacity fluctuations between the stages increase. If we measure both the amount and frequency of shortages over a long period of time in a rolling schedule environment, a quite simple rule that only takes the available capacity during the first stage into account outperforms more intelligent rules.

Keywords*:* Synchronous manufacturing; order release; sequencing, heuristics; integer programming

1 Introduction

Synchronous manufacturing is a concept that is behind several popular shop floor management approaches, such as Lean and Quick Response Manufacturing. The basic idea is to create a flow of work through the manufacturing system, either continuous or intermittent, in order to achieve short and constant throughput times and a predictable loading of the resources in the system. Both the structure of the production system and the control system are to be designed such that this flow can be achieved (Schmenner and Swink 1998).

One of the earliest and most famous synchronization methods is single product assembly line balancing (see Scholl and Becker (2006) for an overview). The production system is designed as a connected line, progress control is eliminated at all, and the order release decision reduces to the decision on the cycle time of the line, i.e. the time between subsequent releases. The concept was applied first in the high volume automobile plant of Henry Ford. As changing the cycle time (a control decision) may require a different design of the production line (i.e., number of stations and assignment of tasks), the relation between both design decisions is central in the assembly line balancing problem.

More recent approaches give attention to the issues in case the manufacturing process is not connected as a single product line. For multi-product lines, the concept of assembly line balancing has been extended to mixed and multi model assembly line balancing and generalized (Becker and Scholl 2006). If the product(s) are no longer produced within a single line but in several stages of production, a control system can be used to connect several autonomous parts of the manufacturing system. Period Batch Control (PBC) (Burbidge 1962, 1996, Benders and Riezebos 2002) is to be considered as the earliest example in this category. PBC

synchronizes the assembly stage with one or more preceding and feeding stages of production, such as parts manufacturing and material acquisition. Ahmadi and Wurgraft (1994) give attention to the issue of designing such manufacturing systems. The control system becomes more essential in achieving a synchronized flow of work through the manufacturing system. Control within a stage of production is often decentralized to relatively autonomous production units, denoted as cells or teams. Here the concept of cellular manufacturing or team oriented assembly emerges (Bukchin and Masin 2004). PBC provides the control between the stages through short but fixed offset times that are identical for all stages. Orders are released at fixed time intervals. The resulting capacity load variation over time is predictable and managed through the order release decision and capacity management instruments, as sufficient time remains for capacity level changes and outsourcing decisions.

An important characteristic of this synchronization approach is the use of a fixed cycle time for each stage. Other approaches, such as the Kanban system and the Drum Buffer Rope system, do not use a fixed cycle time for each stage, which complicates the capacity management problem. They focus on the order release decision and use a type of master production schedule (denoted as a level schedule in Kanban systems (Miltenburg 1989), and as a drum schedule in DBR systems (Riezebos *et al.* 2003)) to balance the load for specific stages over time.

Recently, fixed cycle-time synchronization approaches have been developed that no longer implicitly assume an inflexible capacity. They consider total capacity to be limited, but capacity to be flexible between the stages of the production system. The reason for relaxing this assumption is that workers are nowadays increasingly multi-skilled and cross-trained. This flexibility makes it possible to handle capacity fluctuations between stages in a production system. However, total capacity in terms of the number of workers available does not increase through such measures. Therefore, these synchronization approaches aim at an order release decision that effectively uses the available capacity of the multi-skilled workers while still realizing a high output for the whole production system. Examples of papers in this area are Lee and Vairaktarakis (1997), Vairaktarakis *et al.* (2002), Gronalt and Hartl (2003), Bukchin and Masin (2004), and Cevikcan *et al.* (2007). The complexity of the resulting synchronization problems has been analyzed by Vairaktarakis and Cai (2003). They showed that most levelling problems in such systems are NP-complete, even if they only consist of two stages. Therefore, in practice heuristic solutions are being used to solve the order release decision, as the number of stages often is much larger than two.

This paper discusses various single pass heuristics for the order release decision in such a synchronization approach with a fixed cycle time in a multi-product multi-stage situation. Single pass heuristics determine a sequence without back tracking or pair wise interchanging parts of a solution. Such heuristics are often used in practice. The question is whether these heuristics can be improved by incorporating insights from related fields, such as workload control and assembly line balancing. Moreover, the performance of these heuristics may or may not be sensitive to problem size and capacity fluctuation. This paper will examine the performance of several heuristics in a rolling schedule environment as faced by firms that apply such a synchronization approach. Testing in a rolling schedule environment provides better insights in the performance of these heuristics in the long term, as it prohibits the negative impact of postponing problems to the end of the cycle.

This paper is organized as follows. We introduce the problem by discussing an illustrative problem, based on the case of a jewellery firm in Ssection 2. Section 3 presents the heuristics that we developed for supporting the order release decision. Section 4 develops a mathematical programming model that determines the sequence of order releases. This model not only aims at a feasible solution for the near future, but takes also in account the possibility of achieving feasible solutions behind the planning horizon. Section 5 evaluates the performance of both approaches using a simulation approach. It describes the experimental design and presents the results. Section 6 gives our conclusions and suggestions for future research.

2 Problem definition

An illustration of the order sequencing and capacity balancing problem is found in the longitudinal study of Süer (see e.g. Süer and Gonzalez 1993, Süer 1998) on a jewellery firm. Production of jewellery requires similar operations for various product types, such as earrings, rings, bracelets and chains. The processing plans may vary per product type in terms of type of plating. Most operations are performed manually, using simple tools, small machines and equipment. Examples are casting, cleaning, and tumbling. The employees of a cell have been trained to perform all tasks in all stages of the production process. Speed differences between employees are minor. Capacity management is therefore mainly oriented towards work force scheduling, and hiring/firing decisions.

The firm operates in synchronous manufacturing mode. The stages represent a subset of operations that are to be performed within a single period of time (four hours in case of the jewellery firm). At the end of each period, the various subsets of operations all need to be completed, as all jobs that are in progress are transferred to their next stage at the same moment in time. At such a transfer moment, employees may have to switch to other tasks within the cell. Some operations, such as tumbling and plating, are necessarily performed in batch. The minimal throughput time for a batch of products determines the length of the period that can be used in a synchronous manufacturing system.

The cell starts each period with a new order that is selected from a list of orders that should be completed during the cycle (i.e., at the end of the week). The batch size, process plans, and work content per stage may differ per order. Therefore, capacity requirements may vary strongly, both per order and per stage. The expected throughput time of a batch within a stage is a function of the number of employees allocated to this stage. In a synchronous manufacturing system, the minimal number of employees required in each stage in order to finish this batch within the end of each period can be calculated.

An important problem faced in such synchronous manufacturing systems concerns the capacity balance over time (Cesani and Steudel 2005). A cellular system makes it less appropriate to vary the total number of employees over time. Employees should feel themselves responsible for the whole task of the cell. A relatively constant number of employees over various periods is therefore preferred.

There are two planning decisions that affect the objective of achieving a uniform loading of total capacity in multi-model systems. The batch-size decision, and the order sequencing or release decision (Karabati and Sayin 2003). This paper will give attention to the latter decision. The release decision can be supported either by intelligent design of an order sequencing heuristic, or by creating beforehand one or more feasible order release sequences from which the cell can choose. This paper compares these two approaches for direct support of the order release decision.

2.1 Illustrative example

Table 1 presents a realistic example with 10 orders that have to be released during one week. We have 10 orders (A,…,J) and five stages j=1,…,5. An order that starts in period t=1 in stage j=1 arrives in period t=2 in stage j=2, and leaves the system at the end of period t=5, if stage j=5 has been finished. Orders are represented using different shades. The amount of capacity required in a stage differs per order and is presented in the cells of the table. A row shows the fluctuating capacity requirements of that stage. Orders may also require a different total number of employees (i.e. the sum of the cells with identical shade). However, the sum of the cells in the same column is more important for the capacity management of the firm. This shows the total number of employees needed in a single period. If this number exceeds the available capacity, the firm has to hire additional employees or change the sequence of orders. The problem is to determine one or more sequences for the orders A,…,J such that the available capacity of 20 employees in each period t is not exceeded as long as possible.

For the first four periods, earlier decisions on the sequence in a previous week affect the loading of the available capacity. The cells at the bottom-left side express the capacity requirements of these already started orders that still have to complete one or more stages. The same effect appears at the end of the week, as the last four orders in the sequence affect not only capacity in this week, but also capacity requirements in the next week. Here the end-of-horizon effect or truncated-horizon effect appears (see e.g. Baker 1977, Stadler 2000).

INSERT TABLE 1 ABOUT HERE

Literature uses various approaches to cope with the end-of-horizon effect. Some papers (e.g., Ding and Cheng 1993, Lee and Vairaktarakis 1997, Ouenniche and Boctor 1998) totally neglect the end-of-horizon effect. Others (e.g., Karabati and Sayin 2003, Vairaktarakis and Cai 2003) assume that exactly the same order release pattern will be repeated during the next week. These methods do not address the end-of-horizon effect in a realistic way. Gronalt and Hartl (2003) introduce the overall average worker assignment in their numerical experiments. We propose a slightly different approach that uses the stage-average capacity requirements as capacity load factor if the actual capacity load is unknown. According to Stadler (2000), this provides a terminal condition that improves the quality of the solutions in a rolling schedule environment. The upper-right side of Table 1 shows the average required capacity per stage. For example, stage j=1 has an average capacity load of $(3+5+3+2+2+3+2+4+3+6)/10 = 3.3$ employees, which is used to estimate the load in stage j=1 in periods t=11-14.

Note that in Table 1 no order exceeds the capacity limit of 20 employees. For example, order A requires 3+5+4+6+2=20, and J requires 6+2+2+2+2=14 employees. The mean required capacity per order equals 18.6 employees.

The loading should result in a sequence for which the available capacity per period is not exceeded. The sequence presented in Table 1 results in four periods that encounter a capacity shortage: t=2,6,9,10. Can a better order sequence be found?

2.2 *Mathematical problem formulation*

Let $i=1,\ldots,n$ index for orders

$j=1,\ldots,m$ index for stages

$t=1,\ldots,n+m-1$ index for periods

Parameters

C_{ij} Capacity requirements to complete stage j of order i

$i=1,\ldots,n;\ j=1,\ldots,m$

CR_{tj} Capacity requirements to complete stage j in period t for an order from a former cycle

$t=1,\ldots,m-1;\ j=t+1,\ldots,m$

ARC_j Average required capacity in stage $j = \dfrac{\sum_{t=1}^{j-1} CR_{tj} + \sum_{i=1}^{n} C_{ij}}{n+j-1}$

$j=1,\ldots,m-1$

AC_t Available capacity in period t (assuming the capacity is fully cross-trained)

$t=1,\ldots,n+m-1$

w_t Weight factor for capacity shortage in period t (i.e. $w_t = 1$ *if* $t = 1,...,n; w_t \leq 1$ *if* $t > n$)
t=1,...,n+m-1

Decision variable

X_{it} *=1* if order i starts in period t (determines the sequence)
=0 else
i=1,...,n; t=1,...,n

Intermediate variables

Y_{tj} Capacity requirements to complete stage j in period t
t=1,...,n+m-1; j=1,...,m

CS_t Capacity shortage (expected) in period $t =$ $\max\left(0, \sum_{j=1}^{m} Y_{tj} - AC_t\right)$
t=1,...,n+m-1

The problem can now be formulated as:

Given:

- a set of *i=1,...,n* orders that have to start during periods *t=1,...,n*;
- their capacity requirements C_{ij} during *j=1,...,m* stages of completion;
- the capacity requirements CR_{tj} for orders that have already started but are not yet completed;
- available capacity per period AC_t,

determine X_{it} $\left(i = 1,...,n; t = 1,...,n\right)$ (the sequence of the n orders)

such that $\sum_{t=1}^{n+m-1} w_t CS_t$ is minimized.

INSERT FIGURE 1 ABOUT HERE*

Figure 1 provides an overview of the variables and parameters in terms of the structure of Table 1. Note that the value of the variable Y_{tj} in this table is area-specific.

- In the lower-left side of the table, Y_{tj} is determined by the given capacity requirements CR_{tj} of already started but not yet completed orders.
- In the upper-right side of the table, it is estimated by calculating the average required capacity in the stage.
- In the remaining part of the table, it is calculated as a linear function of the order sequence decision and the known capacity requirement of the orders. We use the following expression for calculating this stage load:

$$Y_{t+j-1,j} = \sum_{i=1}^{n} X_{it} \cdot C_{ij} \quad \forall t = 1,...,n \ \forall j = 1,...,m \qquad (1)$$

Analogues to job scheduling, this problem can be viewed as a weighted tardiness problem, where tardiness is not related to exceeding the due date of a job but to exceeding the available capacity in a period. Morton and Pentico (1993: p. 54) denote that problems using this objective are very difficult to solve exactly. The

complexity of this problem has not yet been determined, but similar problems that are discussed in Vairaktarakis and Cai (2003) are NP complete in the strong sense for situations with the number of stages *m* greater or equal to three.

3 Order release heuristics

This section describes several single pass heuristics to determine a sequence of orders for release. Single pass heuristics do not reconsider decisions taken during the procedure. The heuristics are based on insights from assembly system design and work load control. We will introduce the heuristics and show the results of applying the heuristic on the data of Table 1.

FillCap This heuristic is actually being applied in the jewellery firm. It chooses at the start of period *t* from the set of assignable orders the order that maximally fills capacity in that period. The set of assignable orders contains all orders that still have to be released and have a capacity requirement in stage *j*=1 nearest to but not exceeding the available capacity. If there are more orders to select from, it chooses randomly. We denote this heuristic as FillCap. See the Appendix for a pseudo-coded description of the heuristic.

From Table 2 we see that the heuristic FillCap does not take into account the effect of a selected order on the available capacity for the next *m*-1 periods. As a consequence, in period 9 a capacity shortage occurs.

*** INSERT TABLE 2 ABOUT HERE ***

AvgLoad The AvgLoad heuristic applies the principle of bottleneck loading that is known from mixed model assembly line balancing (e.g. Askin and Standridge 1993: 56-59). The idea is to select the job that will result in a future load on the bottleneck as close to its average rate as possible. Each new assignment will try to correct the gap between the average rate and the cumulative load on the bottleneck that has resulted from earlier assignments.

The bottleneck of this synchronous manufacturing system is not a single stage, as is usually the case in mixed model line assembly problems, but the number of employees available. The heuristic should select an order that (if possible) can be assigned to the first period and that brings the total released workload as close as possible to the long time average.
The calculation of the long time average uses all information available, i.e. the load of the set orders to be released in this cycle (each order requires on average 18.6 employee periods) and the load of the already released orders from the former cycle (these four orders generate a total workload of 37 employee periods during 10 buckets (4+3+2+1). As each order generates workload during 5 buckets (*m*=5 stages), the backlog from the preceding cycle corresponds to 2 complete orders. The average load of an order on the bottleneck equals (10*18.6 + 37) / (10+2) = 18.58 employees per order.
So for the first release decision, we would like to choose an order with a total workload as near as possible to the average rate minus already released load, i.e. 3*18.58–37 = 18.75. After selecting the order (either D or I), the load aimed at for the next release decision is calculated. We denote this heuristic as AvgLoad. See Table 2 for the results in the example problem.

StageLoad The heuristic StageLoad aims at achieving a balanced workload distribution over the stages. It is based on insights from the field of workload control (Land and Gaalman 1996, Riezebos *et al.* 2003, Stevenson *et al.* 2005). It searches for an order that fits best within the expected peaks and troughs of the already generated workload distribution from the past assignments. The order to be selected should have a dissimilar workload pattern as compared to the already generated pattern. This generates a less variable

stageload pattern, which also reduces the variability of the sum of the *m* stageloads. In this way we might be able to avoid suddenly appearing large fluctuations in the required number of employees.

StageLoad considers the release of one order as a release of *m* separate workloads for the stages. Each one of them should receive a workload that brings the total as close to the desired average amount as possible. StageLoad selects the order that minimizes the sum of these *m* quadratic distances. The sum of squares in the first step of StageLoad is minimal for order C, so C is chosen. See Table 2 for the results on the example problem.
By choosing the order with minimal quadratic sum of distances to the average stage loads, the creation of sudden peaks and troughs in capacity load is avoided. However, this does not guarantee that sufficient capacity will always be available. For example, in period 4 order B is selected, as this order realizes the best stage load distribution, but this order creates a conflict with the available capacity in period 4.

AvailStageLoad The fourth heuristic tries to avoid the conflicts with available capacity and applies the StageLoad heuristic only to the set of orders that fit within the available capacity. Note that the FillCap and AvgLoad heuristic do the same. If no orders fit within the available capacity, the whole set of unassigned orders is used to determine the order that fits best. The heuristic AvailStageLoad therefore combines the two approaches.

The results of the heuristic AvailStageLoad in the example problem are exactly equal to heuristic StageLoad, as in each stage the preferred order belongs to the set of orders from which it should be chosen (see Table 2). In general, the performance of heuristic AvailStageLoad is expected to be better than heuristic StageLoad.
The computational complexity of all four heuristics is $O\left(n^2\right)$, as in a single pass heuristic the whole cycle is repeated *n*-1 times, and step 2-3 is of order *n*. The first heuristic FillCap is a simple straightforward rule. The other three are more sophisticated in their computations, as they aim at achieving not only a short-term advantage, but a long-term balance as well. However, they allow a lower than possible fill in the current period in order to achieve the long-term balance.
No guarantee is given that any of the heuristics is able to find an optimal solution. For the example problem, all four heuristics were unable to find an order sequence without violating the capacity constraint in at least one period. This raises the question on the possibility of finding a sequence that results in a feasible solution. The total number of sequences is *n*!. Complete enumeration is therefore no realistic alternative for large *n*. However, the advantages of a good order sequence for the productivity and acceptance of this synchronous manufacturing system makes it worthwhile to search for an improved solution procedure. The next section will proceed with such a procedure.

4 A mathematical model to support the order release decision

This section discusses a mathematical model that generates an optimal order sequence for all orders that have to be released in the cycle. The mathematical model is as follows:

$$Minimize \sum_{t=1}^{n+m-1} w_t \cdot CS_t \quad (2)$$

such that :

$$\sum_{i=1}^{n} X_{it} = 1 \qquad \forall t = 1,...,n \qquad \text{(each period exactly one order has to start)} \quad (3)$$

$$\sum_{t=1}^{n} X_{it} = 1 \qquad \forall i = 1,...,n \qquad \text{(each order is assigned to one starting period)} \quad (4)$$

$$Y_{tj} = CR_{tj} \qquad \forall j = 2,...,m \; \forall t = 1,...,j-1 \qquad \text{(stage load already determined in former cycle)} \quad (5)$$

$$Y_{tj} = ARC_j \qquad \forall t = n+1,...,n+m-1 \qquad \text{(stage load in future cycles assumed}$$

$$\forall j = 1,...,t-n \qquad \text{to be equal to the average load)} \quad (6)$$

$$Y_{t+j-1,j} = \sum_{i=1}^{n} X_{it} \cdot C_{ij} \qquad \forall t = 1,...,n \; \forall j = 1,...,m \qquad \text{(stage load resulting from release sequence)} \quad (7)$$

$$\sum_{j=1}^{m} Y_{tj} - CS_t \le AC_t \qquad \forall t = 1,...,n+m-1 \qquad \text{(exceeding available capacity results in shortage)} \quad (8)$$

$$X_{it} \in [0,1]; CS_t \ge 0; Y_{tj} \ge 0; \quad (9)$$

This mathematical model uses X_{it} as binary decision variable, indicating that order i starts in period t whenever the variable has value 1. The other variables are non-negative and continuous (see (9)). Constraints (3) and (4) guarantee that a feasible assignment is obtained. The third constraint (5) shows that we use this model in a rolling schedule environment, as we take the effects of the release decisions in the former cycle into account in the current decision. Constraint (6) calculates the consequences for future periods that will be affected by the current decision. We use the average load in a stage if the actual load is not being determined in the current decision round. Constraint (7) determines the consequences of the release decision for the stage load in subsequent periods and is similar to Formula (1). Constraint (8) calculates the capacity shortage per period.

The objective function (2) minimizes a weighted sum of the capacity shortages over time. Through the use of positive weights < 1 for periods behind the sequence horizon, the end of horizon effect (Stadtler 2000) is avoided. Most sequencing algorithms that are known from assembly line balancing (Scholl and Becker 2006) or suggested for synchronous systems (e.g. Vairaktarakis *et al.* 2002, Gronalt and Hartl 2003) do not treat the end-of-horizon problem. As a result, infeasible solutions may be encountered at the next decision moment. Sometimes the special case of a cyclic schedule (Vairaktarakis and Cai 2003) is used to tackle this problem. Our mathematical model avoids this effect by taking into consideration that the loading in the periods behind the horizon should not exceed the on-average available remaining capacity.

Note that the proposed mathematical model is rather easy to read due to the introduction of the intermediate variable Y_{tj}. This holds especially true for modelling the stage load resulting from the release sequence (7).

The model has been implemented in Lingo in order to find a solution for the problem of Table 1. As weights we selected $w_t = 1 \; for \; t \le n; w_t = 0.5 \; for \; t > n$. The last column of Table 2 shows the results. During the first ten periods no shortages occur, neither are they expected for the periods 11-14, i.e. for the periods behind the sequence horizon. Hence we conclude that none of the four heuristics achieved an optimal result. The question is whether there are differences in the performance between these heuristics.

5 Experimental design and simulation results

5.1 Experimental design

The former sections have introduced the four heuristics and the mathematical model and illustrated their performance using a realistic example. We would like to extend this comparison to a rolling schedule context, as this is the usual situation where such a solution procedure is being applied in synchronous manufacturing systems.

The basic premise we would like to test is whether the solution approaches perform different when the distribution of workload over the stages changes ($MixVar = f\left(\sigma^2_{C_{ij}}\right)$) or the variation in the total workload per order increases ($VolVar = f\left(\sigma^2_{\sum_{j=1}^{m} C_{ij}}\right)$). The first is a mix-oriented factor, while the second is a volume-oriented factor. In addition, we would like to test whether the performance of the approaches under these experimental settings changes if production characteristics (the number of stages m) or planning characteristics (the schedule horizon n) are modified. Table 3 shows the full factorial experimental design.

*** INSERT TABLE 3 ABOUT HERE***

Each of the configurations (experiments) is replicated 100 times with a run length (rolling schedule) of 50 cycles. For each cycle a random problem is generated with n orders and m stages. The total workload per order and the distribution of this workload over the stages varies randomly according to the specified factorial design of the experiment. The problem that has to be solved consists of determining a sequence for the orders. The consequence of the chosen sequence is input to the scheduling problem of the same solution approach for the next cycle, as we test the approaches in a rolling scheduling environment. So, although the approaches receive the same set of new orders at the start of a new cycle, only the first sequencing problem in a replication is completely identical for all solution approaches.

Since we do not know if the size of the problem matters, we have tested the approaches using low and high values for both the scheduling horizon/number of orders n and the number of stages m.

In total 4*4*100*50 = 80000 sets of orders were generated and the five solution approaches solved the resulting problems.

We evaluate the approaches first on the average total capacity shortage per cycle. This is a kind of tardiness measure, as capacity shortage is zero if there is overcapacity in every bucket. For each bucket t=1,…,n we calculate the capacity shortage CS_t and add it to the total capacity shortage in that cycle. The average total capacity shortage is calculated over the number of cycles in the rolling schedule.

Secondly, the frequency of a shortage in a schedule of one cycle is measured. If in any of the buckets a shortage occurs, this cycle is counted as one with shortages. The mean frequency per replication of 50 runs was calculated. This performance indicator shows how often a schedule that is constructed with this approach will face a shortage. The ratio of the two performance indicators (*Exp.Shortage = Shortage / Frequency*) reveals the mean total amount of capacity short in a cycle that faces a shortage.

Available capacity is constant at 20 units per period. Average capacity requirement per order is also constant and set at 18 for all configurations. At a low volume variation, the actual capacity requirement is 18 ± 1, while a high volume variation has actual requirements equal to 18 ± 3. The capacity requirement of an order is therefore either in the interval [17,19] or [15,21], depending on the value of *VolVar*. We use a uniform distribution within an interval, so each element in the set has equal probability. As the maximal mix variation

in a stage, *MV*, may change in the course of the procedure, we first select randomly a stage and next determine the number of units in that stage. This number has to be at least one, as described in the sampling procedure (see Appendix).

The problem generator and the five solution approaches have been programmed in Delphi 6, where the procedure that determines the optimal solution to the mixed integer programming problem uses the dll of the Lingo 5 solver. Generating these 80000 problems and solving them with the five solution approaches took 37.4 hours CPU time on a 2.4 GHz Pentium IV PC running Windows XP, the majority of it being used by the optimization approach. The average CPU time of solving a problem to optimality was 1.5 seconds. The smallest problems $(n+m=15)$ took on average 0.7 seconds CPU time. The use of this version of Lingo in the rolling schedule experimental design did not allow us to solve problems with $n+m>25$.

5.2 Expectations with respect to the results

The solution approaches are expected to perform differently according to the variation in the experimental design factors. We number our expectations in order to enable testing in the next Section.

(i) The heuristic FillCap is expected to perform better if MixVar is high instead of low.

(ii) The effect of an increased variation in the total workload (VolVar high) on the performance of FillCap is uncertain.

The reason for (i) is that a high *MixVar* will result in a higher chance of orders available that fit within the available capacity, even if the amount of capacity available is small.

(iii) Compared to FillCap, AvgLoad is on average expected to result in a lower shortage, while frequency is not changed

(iv) AvgLoad will perform better if MixVar is low instead of high.

(v) AvgLoad will be more effective if VolVar is high.

The heuristic AvgLoad improves the selection procedure of FillCap, but allows idle time earlier in the cycle as it favours a total workload near the average load on the bottleneck. The effect of an increased variation in the total workload (*VolVar* high) on the performance of FillCap is uncertain. The reason for (iv) is that AvgLoad does not take the distribution of workload over the stages into account. A higher variation in this distribution will result in relatively more shortages. Further, as it aims at balancing the total volume, it will be more effective if *VolVar* is high (v).

(vi) StageLoad is not effective in managing shortage and frequency.

(vii) StageLoad is more effective for high MixVar.

We do not expect the heuristic StageLoad to be effective in managing shortage and frequency (vi), as it allows choosing an order while knowing that a shortage occurs. However, it might result in a more stable loading of the stages, which is more effective for high *MixVar* (vii).

(viii) AvailStageLoad is expected to perform better than StageLoad

(ix) AvailStageload is more effective if MixVar is high.

The heuristic AvailStageLoad uses the basic idea of StageLoad in selecting an appropriate order that fits in the available capacity. It is expected to perform better than StageLoad (viii) and improve if *MixVar* is high (ix).

(x) Zero shortage solutions are less easy to find in case either MixVar or VolVar is high.

In case either *MixVar* or *VolVar* is high, we expect that it will be more difficult to find a solution with zero shortage (x).

(xi) Both the average shortage per cycle as the frequency of shortages will increase if n increases.
(xii) The average shortage per bucket decreases if n increases.
(xiii) Only StageLoad and AvailStageLoad benefit from an increase in m.

The experimental variable *n* is expected to have an equal effect on all approaches. If the length of the horizon *n* increases, both the average shortage per cycle and the frequency of shortages are expected to increase (xi). However, longer cycles (*n*↑) means a larger set of orders to select for release, and hence more possibilities to reduce or avoid shortages. So the increase might not be proportional to the increase in n, in other words the average shortage per bucket might decrease (xii). More stages (*m*↑) increases the sensitivity for unbalances between the stages. We therefore expect a performance improvement only for the heuristics that take this into account (i.e., StageLoad and AvailStageLoad) (xiii). Next subsection will describe the results of the experiments, which enables us to determine if these expectations are valid.

5.3 Results

The results of the experiments were analyzed on differences in the mean performance. We used ANOVA to test whether the means were equal or not. Table 4 shows significant differences with respect to the mean shortage for all main effects and all interaction effects except one. Most contributing to differences in the mean are *MixVar*, *Approach*, and the combined effect of these two factors. For the frequency of shortages, the same factors are most important in explaining the total variation. Note that the size of the problem (*n* and *m*) has a significant, but minor contribution to explaining the variance in the data. We conclude that the mean performance of the various approaches cannot be assumed to be equal for all factorial levels.

INSERT TABLE 4 ABOUT HERE

In order to see if the performance of all approaches differs significantly (expectations iii, vi, and viii), we applied multiple comparisons and tested for significant differences using the Tamhane test statistic. Table 5 shows the results of this comparison. The type of approach used shows up to be a highly significant factor, as the difference with all other approaches is significant. We only present the differences between Optimal and the other approaches, but the same holds true for the remaining comparisons. Note that Table 5 presents the average differences, based on all values of the experimental variables *n, m, VolVar,* and *MixVar.*

INSERT TABLE 5 ABOUT HERE

The optimal solution clearly outperforms the other heuristics. The difference in mean shortage with FillCap equals 2.762 employee periods per cycle, while the frequency of cycles with a shortage is 43.7% higher, as can be seen from Table 5 (column Mean Difference (I-J)). AvgLoad performs even worse -contrary to what we expected in (iii), but still better than the AvailStageLoad heuristic. The StageLoad heuristic performs worst, as was expected in (vi) and (viii)). However, the good performance results of FillCap were not expected at all. For all factorial levels, the less sophisticated FillCap appears to be significantly better than the other heuristics that aim at an improved selection process. FillCap just chooses the order that fills the available capacity in the first stage, and this results both in a lower average capacity shortage and less frequent shortages than more sophisticated heuristic selection procedures. Sophistication in these single pass heuristics does not seem to be of value.

INSERT TABLE 6 ABOUT HERE

The question now remains how the solution approaches perform at the two levels of the experimental factors. The descriptive statistics in Table 6 show that an increase in *MixVar* has a much stronger impact on the extra capacity required and the frequency of shortages than an increase in the variation in the total workload of an

order (*VolVar*). However, this holds only true for the heuristic approaches, as the optimal solution is completely insensitive for changes in these experimental factors. For each row in Table 6, 5000 problems have been solved (80000 in total), and the number of problems that did result in a shortage in the optimal solution was zero. Hence, expectation (x) cannot be confirmed.

The performance difference between the heuristic approaches depends on the settings of the experimental factors. Figure 2 shows that a change in *MixVar* results in slightly different gradients of the curves that represent the five solution approaches. Hence, relative performance of the heuristics if compared with each other changes if *MixVar* changes. However, (i), (iv), and (ix) do not hold, as the absolute performance of all heuristics deteriorates for high *MixVar*, while expectation (iv) can be confirmed. And even for the relative performance improvement in average shortage, FillCap is better than the other heuristics.

INSERT FIGURE 2 ABOUT HERE

The results show that the difference between heuristics and the optimal solution is high. Figure 2 shows that if *MixVar* is low, the mean frequency of shortages during a cycle is between 20% and 55% if the heuristics are being used, and increases to 60%-95% if *MixVar* is high, while the optimal solution approach is still able to find a solution without shortages. The frequency of shortages equals zero if the optimal solution approach is used. The required CPU time for obtaining the optimal solution is less than 10 seconds, with an average of 1.5 seconds. Therefore, the use of the optimal solution approach is a realistic alternative if the size of the problem is restricted and leads to substantial savings and an improved capacity balance. Note that the size of problems considered in this study is realistic for many real life situations. Note also that in our experiments the optimal solution was insensitive to the value of the parameter w_t=0.5, as a trade off between short term and long term effects could be avoided (the optimal solution was always able to avoid overtime).

Table 6 shows that *VolVar* has less effect on the heuristics. FillCap is indeed relatively insensitive to changes in *VolVar*, which confirms (ii), but AvgLoad is not performing better for high *VolVar*, as expected in (v).

Finally, we take a look at the effect of the parameters *n* and *m*. An increase in the number of periods per cycle decreases the performance, and hence expectation (xi) is confirmed. However, in case of low mix variation, the effect is very small. That means that the average shortage per bucket decreases if *n* increases (xii). Apparently, longer cycles actually make capacity balancing more effective. With respect to an increase in *m*, a strange phenomenon appears. The direction of the effect depends on the value of *MixVar*. If *MixVar* is low, an increase in *m* deteriorates the performance of the heuristics, but if *MixVar* is high, an increase in *m* improves the performance of all heuristics. Therefore, expectation (xiii) is not confirmed. There is an effect for all heuristics if *m* changes, but we have to accommodate for the interaction effect with *MixVar*. Note that Table 4 already showed that this interaction is important, especially for explaining the variance in the shortage performance measure.

6 Conclusions

The concept of synchronous manufacturing is behind several popular shop floor management approaches. It aims at achieving short and reliable throughput times through the introduction of fixed transfer moments between several stages of production. This causes a loading of the resources that can be predicted in advance, which makes it easier for planners to do their job.

If capacity requirements fluctuate over time, planners may have to enhance the order release decision in order to avoid costs of changing the total available capacity. Our study reveals that the quality of the order release decision is the key to an improved capacity balance over the periods.

In order to develop solution approaches for this multi-product multi-stage problem, we used insights from mixed-model assembly line sequencing algorithms and workload control literature. Four heuristics and a

mathematical model were developed to solve the capacity balancing problem in a synchronous manufacturing system.

We tested the five solution approaches in a rolling schedule. This enabled us to take into account the consequences of the former sequence decisions of the approaches for the new cycle. We evaluate the average total capacity shortage per cycle and the frequency of shortages for the various solution approaches.

Our experimental design evaluates the performance of the approaches for two levels of variation of total capacity requirements per order and for two levels of variation of the distribution of the workload over the stages. Moreover, we evaluated the effect of changing the length of the planning horizon and the number of stages.

Surprisingly, the simple FillCap heuristic that selects an order that consumes as much capacity in the first stage of its release as possible performs in all experiments better than the more sophisticated capacity balancing heuristics. This holds true even if the number of stages *m* increases. We expect that the main reasons are to be found in the bad balance at the end of the cycle, as the more sophisticated heuristics do aim at a very good balance at the start of the cycle, without taking into consideration the effects of their choices at the end of the cycle. However, this study has not examined the main reasons for the bad performance of the sophisticated heuristics, as it expected them to perform better. Future research should elaborate on this.

The performance of the heuristics is very sensitive to an increase in the variation of the distribution of workload over the stages. Increased variation of the total capacity requirements of an order has only a marginal impact on performance.

The use of the mathematical model instead of one of the heuristics results in an almost total elimination of shortages in our experiments. Shortage frequencies of at least 30-60 percent were avoidable by using the mathematical model. The time needed for finding an optimal solution is very short for the problems we have solved (up to 15 orders and 10 stages). The size of problems that we have investigated is realistic for real life manufacturing systems, and implementation of the mathematical optimization model is easy to accomplish. We conclude therefore that the use of such a solution approach should be considered, as it strongly improves the capacity balance in synchronous manufacturing systems.

Future research should verify if modifications of the proposed heuristics for synchronous manufacturing lead to performance improvements. We suggest to develop and test heuristics that take into account uncertainty with respect to the work content and available capacity. Genetic algorithms can be applied as well. The current study has applied deterministic optimization in a rolling schedule horizon. We think that the use of a rolling schedule has important benefits when testing the performance of solution approaches and should be applied to other problems as well.

Acknowledgements

We like to express our thanks to Professor G.A. Süer, University of Ohio, for providing us many details on the jewellery firm and stimulating discussions.

References

Ahmadi, R.H., and Wurgaft, H., 1994. Design for synchronized flow manufacturing. *Management Science*, 40 (11) 1469-1483.

Askin, R.G., and Standridge, C.R., 1993. *Modeling and analysis of manufacturing systems*. New York: John Wiley & Sons.

Baker, K.R., 1977. An experimental study of the effectiveness of rolling schedules in production planning. *Decision Sciences,* 8, 19-27.

Becker, C., and Scholl, A., 2006. A survey on problems and methods in generalized assembly line balancing. *European Journal of Operational Research*, 168, 694–715.

Benders, J., and Riezebos, J., 2002. Period batch control: classic, not outdated. *Production Planning and Control*. 13 (6), 497-506.

Bukchin, J., and Masin, M., 2004. Multi-objective design of team oriented assembly systems. *European Journal of Operational Research,* 156, 326-352.

Burbidge, J.L., 1962. *The principles of production control*. Plymouth: MacDonald and Evans.

Burbidge, J.L., 1996. *Period Batch Control*. Oxford: Oxford University Press, Clarendon Press.

Cesani, V.I., and Steudel, H.J., 2005. A study of labor assignment flexibility in cellular manufacturing systems. *Computers & Industrial Engineering,* 48 (3), 571-591.

Cevikcan, E., Durmusoglu, M.B., and Unal, M.E., 2007. A team-oriented design methodology for mixed model assembly systems. *Computers & Industrial Engineering*, In Press. Available from: DOI:10.1016/j.cie.2007.11.002.

Ding, F-Y., and Cheng, L., 1993. An effective mixed-model assembly line sequencing heuristic for Just-In-Time production systems. *Journal of Operations Management*, 11, 45-50.

Gronalt, M., and Hartl, R.F., 2003. Workforce planning and allocation for mid-volume truck manufacturing: a case study. *International Journal of Production Research*, 41 (3), 449-463.

Karabati, S., and Sayin, S., 2003. Assembly line balancing in a mixed-model sequencing environment with synchronous transfers. *European Journal of Operational Research*, 149, 417-429.

Land, M.J., and Gaalman, G.J.C., 1996. Workload control concepts in job shops: a critical assessment. *International Journal of Production Economics,* 46-47, 535-548.

Lee, C-Y., and Vairaktarakis, G.L., 1997. Workforce planning in mixed model assembly systems. *Operations Research,* 45 (4), 553-567.

Miltenburg, J., 1989. Level schedules for mixed-model assembly lines in Just-In-Time production system. *Management Science,* 35 (2), 192-207.

Morton, T.E., and Pentico, D.W., 1993. *Heuristic scheduling systems – with applications to production systems and project management*. New York: John Wiley & Sons.

Ouenniche, J., and Boctor, F., 1998. Sequencing, lot sizing and scheduling of several products in job shops: the common cycle approach. *International Journal of Production Research,* 36 (4), 1125-1140.

Riezebos, J., Korte, G.J., and Land, M.J., 2003. Improving a practical DBR buffering approach using Workload Control. *International Journal of Production Research*, 41 (4), 699-712.

Schmenner, R.W., and Swink, M.L.,1998. On theory in operations management. *Journal of Operations Management*, 17 (1) 97-113.

Scholl, A., and Becker, C., 2006. State-of-the-art exact and heuristic solution procedures for simple assembly line balancing. *European Journal of Operational Research,* 168, 666–693.

Stadler, H., 2000. Improved rolling schedules for the dynamic single-level lot-sizing problem. *Management Science*, 46 (2), 318-326.

Stevenson, M., Hendry, L.C., and Kingsman, B.G., 2005. A review of production planning and control: the applicability of key concepts to the make-to-order industry. *International Journal of Production Research,* 43(1), 869-898.

Süer, G.A., 1998, Operation and control of cellular systems at Avon Lomalinda, Puerto Rico. *In*: N.C. Suresh, and J.M. Kay, eds. *Group technology and cellular manufacturing: state-of-the-art synthesis of research and practice*. Boston: Kluwer Academic Publishers, 339-361.

Süer, G.A., and Gonzalez, W., 1993. Synchronization in manufacturing cells: a case study. *International Journal of Management and Systems,* 9 (3), 313-337.

Vairaktarakis, G.L., and Cai, X., 2003. Complexity of workforce scheduling in transfer lines. *Journal of Global Optimization,* 27, 273-291.

Vairaktarakis, G.L., Cai, X., and Lee, C-Y., 2002. Workforce planning in synchronous production systems. *European Journal of Operational Research,* 136, 551-572.

Appendix

FillCap heuristic:

$Step\,0$ $U := \{1,...,n\}$

$Y_{tj} := CR_{tj} \quad \forall j = 2,...,m \quad \forall t = 1,...,m-1$

$t := 0$

$Step\,1$ $t := t+1$

$Available_t := AC_t - \sum_{j=2}^{m} Y_{tj}$

$Aim := Available_t$

$Step\,2$ $A := \{i \in U \wedge C_{i1} = Aim\}$

$IF\ A \neq \varnothing\ (feasible\ solution)\ OR\ Aim \leq 0\ (infeasible\ solution), Goto\ Step\,3$

$ELSE\ Aim := Aim - 1, repeat\ Step\,2$

$Step\,3$ $IF\ A \neq \varnothing\ Select\ randomly\ i \in A$

$ELSE\ Select\ randomly\ i \in U$

$X_{it} := 1$

$Y_{t+j-1,j} := C_{ij} \quad \forall j = 1,...,m$

$U := U - \{i\}$

$IF\ U \neq \varnothing\ Goto\ Step\,1\ ELSE\ Stop.$

AvgLoad heuristic

Step 0 $U := \{1,...,n\}$

$Y_{tj} := CR_{tj} \quad \forall j = 2,...,m \quad \forall t = 1,...,m-1$

$ALoad := \sum_{t=1}^{m-1} \sum_{j=t+1}^{m} CR_{tj}$ (*already released load*)

$ARC := \sum_{j=1}^{m} ARC_j$ (*average load per period*)

$t := 0$

Step 1 $t := t+1$

$Available_t := AC_t - \sum_{j=2}^{m} Y_{tj}$

Step 2 $A := \{i \in U \wedge C_{i1} \le Available_t\}$

Step 3 *IF* $A \neq \varnothing$ *Select* $i \in A$:

ELSE Select $i \in U$:

$$such\ that \begin{cases} LoadAimedAt_t := \left(\frac{m-1}{2} + t\right) \cdot ARC - ALoad \\ i := \arg\min \left| \sum_{j=1}^{m} C_{ij} - LoadAimedAt_t \right| \\ \text{If there is no unique } i, \text{ select } i \text{ that minimizes } \left(Available_t - C_{i1}\right) \end{cases}$$

Step 4 $X_{it} := 1$

$Y_{t+j-1,j} := C_{ij} \quad \forall j = 1,...,m$

$ALoad := ALoad + \sum_{j=1}^{m} C_{ij}$

$U := U - \{i\}$

IF $U \neq \varnothing$ *Goto Step* 1 *ELSE Stop.*

StageLoad heuristic

Step 0 $U := \{1,...,n\}$

$$ALoad_j := \sum_{t=1}^{m-1} CR_{tj} \qquad (already\ released\ cumm.load\ to\ stage\ j) \qquad \forall j = 1,...,m$$

$t := 0$

Step 1 $t := t+1$

$$LoadAimedAt_{tj} := (t + j - 1) \cdot ARC_j - Aload_j \qquad \forall j = 1,...,m$$

$$i := \arg\min \sum_{j=1}^{m} \left(C_{ij} - LoadAimedAt_{tj}\right)^2$$

Step 2 $X_{it} := 1$

$$Aload_j := Aload_j + C_{ij} \qquad \forall j = 1,...,m$$

$U := U - \{i\}$

IF $U \neq \varnothing$ *Goto Step* 1 *ELSE Stop.*

AvailStageLoad heuristic

Step 0 $U := \{1,...,n\}$

$ALoad_j := \sum_{t=1}^{m-1} CR_{tj}$ $(already\ released\ cumm.\ load\ of\ stage\ j)$ $\forall j = 1,...,m$

$t := 0$

Step 1 $t := t+1$

$Available_t := AC_t - \sum_{j=2}^{m} Y_{tj}$

$LoadAimedAt_{tj} := (t+j-1)\cdot ARC_j - Aload_j$ $\forall j = 1,...,m$

Step 2 $A := \{i \in U \wedge C_{i1} \le Available_t\}$

Step 3 *IF* $A \neq \varnothing$ *Select* $i \in A$:

ELSE Select $i \in U$:

$i := \arg\min \sum_{j=1}^{m} \left(C_{ij} - LoadAimedAt_{tj}\right)^2$

Step 4 $X_{it} := 1$

$Aload_j := Aload_j + C_{ij}$ $\forall j = 1,...,m$

$U := U - \{i\}$

IF $U \neq \varnothing$ *Goto Step* 1 *ELSE Stop.*

Sampling Procedure

1 Set $Um := \{1,...,m\}$ (*Unassigned stages*)

$VolVar :=$ Volume Variation factor (1 for low variation, 3 for high variation)

$MixVar :=$ Mix Variation factor (1 for low variation, 2 for high variation)

2 Select randomly $VR \in [18 - VolVar, 18 + VolVar]$

$VR =$ Total volume to be allocated to remaining stages $\in Um$

$AV = \left\lfloor \frac{VR}{|Um|} \right\rfloor =$ Average volume to be allocated to remaining stages $\in Um$

($i.e.$ $\lfloor x \rfloor$ nearest integer $\le x$; $|X|$ cardinality of set X)

3 Select randomly stage $j \in Um$

$MV := \min\{MixVar,\ AV - 1,\ VR - AV - |Um| + 1\}$

($MV =$ Maximum variation of remaining volume to be allocated to stage j)

Select randomly workload $C_{ij} \in [AV - MV, AV + MV]$ $(C_{ij} \ge 1, \text{integer})$

$VR := VR - C_{ij}$

$Um := Um - \{j\}$

IF $|Um| > 1$ Repeat Step 3 *ELSE* Goto Step 4

4 $j := Um$

$C_{ij} := VR$

Table 1. Order sequencing with 10 products a week

day	Mon		Tue		Wed		Thu		Fri		Mon2		Tue2	
bucket (4 hours)	am	pm	am	pm	am	pm	am	pm	am	pm	am	pm	am	pm
period *t*	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Stage \ Order	A	B	C	D	E	F	G	H	I	J				
j=1	3	5	3	2	2	3	2	4	3	6	3.3	3.3	3.3	3.3
j=2	4	5	3	2	6	2	2	2	4	3	2	3.2	3.2	3.2
j=3	6	4	4	4	2	6	5	2	5	4	5	2	4.1	4.1
j=4	3	4	5	6	4	8	3	4	8	4	4	5	2	4.6
j=5	2	3	2	4	2	4	5	2	4	5	4	4	3	2
Total	18	21	17	18	16	23	17	14	24	22	21	22	21	28
(Exp.) Shortage	0	1	0	0	0	3	0	0	4	2	0	0	0	0

Table 2. Comparison of the solution approaches

	Sequence				Unused capacity in period *t*			
t	FillCap	AvgLoad	(Avail-) StageLoad	Optimal	FillCap	AvgLoad	(Avail-) StageLoad	Optimal
1	B	I	C	A	0	2	2	2
2	J	D	H	F	0	4	3	1
3	H	E	I	B	3	0	4	2
4	A	A	B	E	3	0	**-4**	3
5	C	G	E	C	2	2	1	1
6	F	B	A	H	5	3	2	0
7	I	C	G	J	3	**-1**	1	0
8	D	J	J	D	3	2	0	0
9	E	H	F	I	**-6**	4	0	0
10	G	F	D	G	0	**-1**	8	3
11					3.7	3.7	2.7	4.7
12					2.5	5.5	**-2.5**	1.5
13					1.4	**-2.6**	1.4	2.4
14					0.8	**-0.2**	2.8	0.8
Value objective function					-6	-3.4	-5.25	0

Table 3. Full factorial design

Variable	*n*	*m*	*MixVar*	*VolVar*
Low value	10	5	± 1	± 1
High value	15	10	± 2	± 3

Table 4. ANOVA results

	Shortage				*Frequency*			
Source	Sum of Squares	df	F	Sig.	Sum of Squares	df	F	Sig.
Model [a,b]	236367.162	80	9581.563	.000	2562.783	80	13035.129	.000
MixVar	57556.686	1	186653.009	.000	292.712	1	119106.098	.000
Approach	39059.124	4	31666.621	.000	495.800	4	50435.970	.000
n	3037.496	1	9850.425	.000	7.517	1	3058.548	.000
VolVar	599.304	1	1943.510	.000	1.350	1	549.502	.000
m	463.213	1	1502.172	.000	32.362	1	13168.366	.000
*MixVar * Approach*	22250.719	4	18039.449	.000	75.846	4	7715.564	.000
*m * MixVar*	4468.346	1	14490.588	.000	28.429	1	11567.998	.000
*n * MixVar*	1789.586	1	5803.524	.000	.022	1	8.943	.003
*n * Approach*	1729.621	4	1402.265	.000	1.972	4	200.636	.000
*m * Approach*	1641.318	4	1330.675	.000	8.338	4	848.199	.000
*VolVar * Approach*	275.833	4	223.627	.000	.823	4	83.688	.000
*VolVar * MixVar*	239.135	1	775.499	.000	.373	1	151.965	.000
*n * VolVar*	31.408	1	101.853	.000	.008	1	3.174	.075
*m * VolVar*	2.554	1	8.282	.004	.915	1	372.171	.000
*n * m*	.061	1	.197	.657	.702	1	285.648	.000
*m * MixVar * Approach*	3734.784	4	3027.922	.000	8.738	4	888.889	.000
*n * MixVar * Approach*	1003.472	4	813.550	.000	1.697	4	172.672	.000
*n*m*VolVar*MixVar*Approach*	615.358	41	48.672	.000	3.432	41	34.058	.000
Error	2442.227	7920			19.464	7920		
Total	238809.388	8000			2582.247	8000		

a R Squared *Shortage* = .990 b R Squared *Frequency* = .992

Table 5. Multiple comparisons of performance solution approaches

Dependent Variable	(I) *APPROACH*	(J) *APPROACH*	Mean Difference (I-J)	Standard Error	Sig.	95% Confidence Interval	
						Lower Bound	Upper Bound
Shortage	Optimal	FillCap	-2.762(*)	.058	.000	-2.925	-2.598
		AvgLoad	-3.638(*)	.081	.000	-3.864	-3.412
		Stageload	-6.817(*)	.150	.000	-7.237	-6.397
		AvailStageLoad	-4.271(*)	.086	.000	-4.513	-4.029
Frequency	Optimal	FillCap	-.437(*)	.007	.000	-.456	-.418
		AvgLoad	-.497(*)	.008	.000	-.518	-.475
		Stageload	-.760(*)	.006	.000	-.778	-.742
		AvailStageLoad	-.543(*)	.007	.000	-.562	-.524
Exp. Shortage	Optimal	FillCap	-4.887(*)	.066	.000	-5.074	-4.701
		AvgLoad	-5.539(*)	.083	.000	-5.770	-5.307
		Stageload	-7.492(*)	.139	.000	-7.882	-7.101
		AvailStageLoad	-6.308(*)	.088	.000	-6.556	-6.061

* The mean difference is significant at the .05 level.

Table 6. Performance of solution approaches

				Shortage					*Frequency*					*Exp. Shortage*				
n	*m*	*MIXVAR*	*VOLVAR*	FillCap	AvgLoad	Stageload	AvailStageLoad	Optimal	FillCap	AvgLoad	Stageload	AvailStageLoad	Optimal	FillCap	AvgLoad	Stageload	AvailStageLoad	Optimal
10	5	± 1	± 1	.05	.07	.45	.12	.00	.03	.04	.26	.07	.00	1.39	1.56	1.75	1.67	.00
			± 3	.19	.20	.79	.67	.00	.08	.09	.36	.22	.00	2.18	2.26	2.18	3.01	.00
		± 2	± 1	4.15	5.44	11.84	6.83	.00	.59	.70	.98	.75	.00	7.08	7.81	12.04	9.15	.00
			± 3	4.72	7.36	12.15	7.47	.00	.60	.77	.98	.74	.00	7.85	9.62	12.45	10.05	.00
	10	± 1	± 1	.83	1.04	2.03	1.29	.00	.29	.34	.66	.39	.00	2.89	3.07	3.07	3.32	.00
			± 3	.96	1.25	2.02	1.50	.00	.31	.37	.64	.39	.00	3.07	3.37	3.15	3.78	.00
		± 2	± 1	3.53	4.13	6.64	4.97	.00	.64	.69	.93	.74	.00	5.54	5.97	7.12	6.74	.00
			± 3	4.15	5.34	7.19	5.91	.00	.64	.72	.93	.74	.00	6.53	7.40	7.77	7.98	.00
15	5	± 1	± 1	.05	.08	.74	.13	.00	.03	.04	.39	.07	.00	1.33	1.50	1.89	1.69	.00
			± 3	.23	.18	1.12	1.12	.00	.09	.08	.47	.33	.00	2.41	2.15	2.37	3.44	.00
		± 2	± 1	5.43	6.98	17.86	8.71	.00	.68	.77	1.00	.80	.00	7.96	9.11	17.90	10.88	.00
			± 3	6.07	9.70	18.59	10.04	.00	.70	.84	1.00	.81	.00	8.70	11.61	18.66	12.36	.00
	10	± 1	± 1	1.25	1.37	3.06	1.72	.00	.39	.43	.82	.47	.00	3.21	3.21	3.73	3.65	.00
			± 3	1.29	1.57	3.05	2.21	.00	.38	.43	.78	.49	.00	3.42	3.64	3.93	4.46	.00
		± 2	± 1	5.12	5.73	10.25	7.05	.00	.76	.81	.99	.83	.00	6.78	7.10	10.37	8.50	.00
			± 3	6.16	7.79	11.28	8.60	.00	.78	.84	.98	.84	.00	7.86	9.25	11.49	10.25	.00

Figure 1. Variables, parameters and structure of Table 1

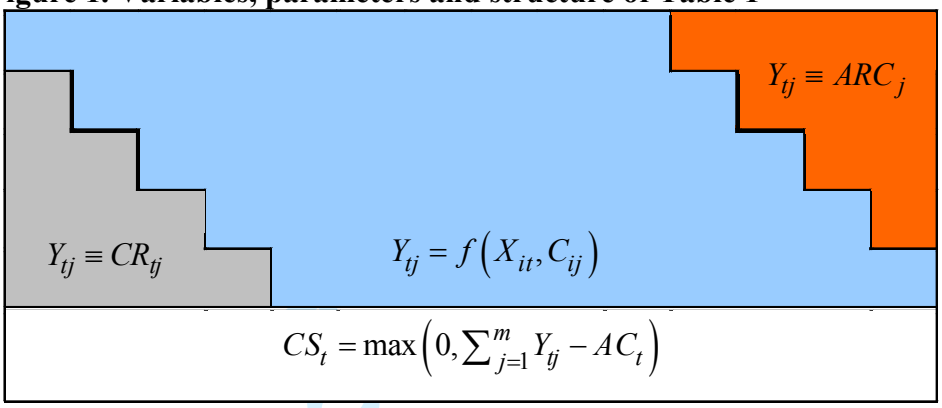

Figure 2. Performance for increased variation in distribution of workload over stages