Proceedings of the 2011 International Conference on Industrial Engineering and Operations Management
Kuala Lumpur, Malaysia, January 22 – 24, 2011

A Lot-sizing Algorithm for Pull Production Environments

Olufemi Adetunji, VSS Yadavalli
Department of Industrial Engineering and Systems Engineering,
University of Pretoria, Hatfield, Pretoria 0002, South Africa

Abstract

This paper presents an algorithm that that can be used in a pull production environment for increasing the product mix through reduced lot sizing while at the same time minimising the system inventory holding cost. Such pull system could be governed by the principle of Lean Manufacturing, Theory of Constraints (TOC) or Constant Work-in-Process (CONWIP). It utilizes the extra time that is available in the non-bottleneck (NBN) resource to produce smaller lot sizes that can be made available to the downstream section of the production. It was shown through a numerical example that the algorithm fares better than a previous algorithm in which the holding cost was not explicitly included, and that the previous model is an instance of the more generalised algorithm presented in this paper.

Keywords
Lot-sizing, Utilisation, Setup, Pull, Algorithm

1. Introduction

Potts and Wassenhove [1] defined batching as making decision about whether or not to schedule similar jobs contiguously, and lot sizing as the decision about when and how to split the production of identical items into sob-lots. They noted that these decisions were traditionally taken as if lot sizing are independent of scheduling of jobs. This is obviated by the majority of the body of literature available on both subjects that are separate, with the impression being given that scheduling decisions are taken only after lot sizes of the various products have been decided. This assumption of independence is not usually true in most cases as the decisions are always inter-twined. Paul and Wassenhove also proposed a general model for integrated batching, lot sizing and scheduling. Drexl and Kims [2] noted that lot-sizing and scheduling are two short term decisions of production planning that must be tied together with the medium term plan, which is the Master Production Scheduling of the system. Many models have since been published addressing integrated batching, lot sizing and scheduling. Potts and Kovalyov [3] and Webster and Baker [4] together with [1] and [2] are good readings.

There is a close relationship between system utilisation and other system parameters like the Work-in-Process Inventory (WIP) and consequently the system holding cost and profitability. Variability in resource processing time and/or input arrival pattern have degrading influence on WIP level as the system gets close to full utilisation. This is succinctly summarised in the Little's law. This effect of resource utilisation on the production plan and the level of WIP appears not to have been well studied. Among the few known models incorporating resource utilisation into production scheduling include Rappold and Yoho [5], and a model proposed in Hopp [6]. The model proposed in [6] is simple and straightforward to use, and that is what has been extended in this paper.

2. Literature review

Solving integrated batching, lot sizing and scheduling problems have received a lot of research attention recently. This could have also been buoyed by the development of many heuristics and techniques for solving difficult combinatorial problems. Among the recently published work in this area include Toledo et al [7], which evaluated different parallel algorithms for multi population genetic algorithm and used the algorithm to solve a synchronised and integrated lot sizing and scheduling problem. Dye and Hsieh [8] extended the Economic Order Quantity model to a case of price-dependent and time varying demand, and fluctuating unit purchasing cost. They utilised Particle Swam Optimisation to find the optimal replenishment number, time scheduling and periodic selling price to

optimise the discounted total profit. Pahl and Voß [9] presented a model that included product deterioration and perishability constraints on lot-sizing and scheduling.

Condotta et al [10] presented a parallel batch scheduling model of jobs that are assumed to be of equal length with release and due dates. Konstantaras and Skouri [11] presented lot sizing model for a joint manufacturing and re-manufacturing problem with variable setup. Chen et al [12] investigated the integration of lot-sizing and dispatch related decisions for a wafer fabrication facility. The objective is to minimise output variability, and they proposed a dynamic Lot Release Planning (DynaLRP) model. They also designed a Lagest-Remaining-Quota-First rule for dispatching of jobs. Srinivasan and Viswanathan [13] considered a manufacturing system that operates in a high-variety, low-volume environment with significant setup times. The objective was to minimise the total WIP inventory across all products.They modelled the system as a closed queuing network with multiple product types.

Dolgui et al [14] studied the problem of optimal scheduling and lot-sizing of a number of products on a given number of unrelated parallel machines to satisfy a given demand, where a sequence dependent setup time is required between lots of different products. A greedy type heuristic was proposed and tested empirically. Narayanan and Robinson [15] considered a joint replenishment problem in a dynamic rolling horizon. They evaluated nine joint replenishment lot-sizing heuristics and found that none of them excel on both dimensions of schedule cost and stability.

Most models discussed above assume that there is a fixed setup cost that must be incurred for every extra setup, against which they are optimising the inventory holding cost. They also largely neglected the effect on WIP (and by extension the holding cost of the system) of capacity utilisation, especially as it approaches full utilisation.

The approach presented in Hopp [6] and re-presented here is a bit different from others in that setup cost is not really considered to be increasing with frequency of setup. This is in tandem with many TOC and Lean implementation principles where it is assumed that most resources have extra capacities (e.g. spare labour time) that could be utilised for Kaizen group meetings and/or batch reduction initiatives. This means the number of setups can be reasonably increased without degrading the system profit and/or cost functions significantly since the system is assumed to be replete of extra capacities in most of the processing stations where utilisation is much lower than the maximum level possible (also called the Bottleneck rate. In this paper, such extra capacity is utilised for lot-size reduction. So, such increased setup due to lot size reduction is assumed to come at no cost to the system.

3. Pull philosophies and Hopp's Joint Production Algorithm

One of the trends in production management is a shift from the traditional push to some form of pull approach. Manufacturing philosophies like Lean, TOC and the more recent CONWIP are built around this approach. They focus more on the flow of products through the production system than on the management of capacity. Many companies have reported significant improvement in their returns as a result of such shift.

Hopp [6] stated three main advantages of a pull system over a push system. Firstly, pull works on observability, which makes it easy to implement. Secondly, it is generally more efficient than an equivalent push system. Thirdly, it is more robust to production errors and changes in orders and/or manufacturing design. He stated that the most distinguishing factor of any pull technique is the ability to keep a constant work-in-process in such system, irrespective of how this is achieved.

A common nature of all systems built around the pull is the accommodation of extra capacities. The flows are balanced, while capacities are typically not. In Lean, for example, the labour can use the extra time they have to organise functional kaizen team meetings. This should help in continuous improvement of the system. In Theory of Constraints (TOC) also, the extra capacities of the non-bottle neck (NBN) could be used to further the reduction of the production batches and in the management of quality. Batch size reduction is also advocated in the CONWIP system. This will increase the flexibility of the system, both in terms of the number of varieties that could be made available to the downstream members of the system, and also in terms of the frequency of release of batches to the market. This will also reduce the waiting time of customers when orders are placed just after a batch of a variety of products has just been released. Aryanezhad et al [16], for instance, has also shown how the spare capacities in the non-constrained stations of a system utilising the theory of constraints philosophy can be used to augment the capacities of the constrained station.

3.1. Hopp's Procedure and motivation for modification

Hopp [6] presented a procedure that can be used to batch and sequence jobs on a NBN resource. This has been formalised into an algorithm in this section. Consider a resource on which some n products must be processed over a time window, say one month, with the demand for each product $i = 1, 2, \dots, n$ during the time window being $d_1, d_2, \dots, d_n$, and the processing time (in hour) for each product being $t_1, t_2, \dots, t_n$. He defined the Batch Size (BS) for each of the i products to be

$$BS_i = \frac{d_i}{N_i} \tag{1}$$

where N_i is the number of setups assigned to product i during the time window.
The question then is to determine the number of setups, N_i, assigned to each of the i products, given that the Setup Time for each product, ST_i is known. To address this, he first defined a variable called the Total Process Time (TPT) as

$$TPT = \sum_{i=1}^{n} d_i . t_1 \tag{2}$$

He then defined another variable called UTIL(zero setups) as the level of utilisation of the resource without including the time taken for the setup for any of the products on the resource. This can be represented as

$$UTIL(zero\ setups) = \frac{TPT}{Available\ time} \tag{3}$$

He also stated the Run Length for product i mathematically as

$$RL_i = ST_i + \frac{d_i . t_i}{N_i} \tag{4}$$

Since in a typical M/M/1 queuing system, it is well known that queuing time balloons as utilisation level approaches 100 percent, a good practice is to keep utilisation low (see Hopp [6], and Webster [17]). Hopp decided to allow an utilisation level of

$$UTIL = \sqrt{UTIL\ (zero\ setups)} \tag{5}$$

The Available Setup Time (AST) is then defined as

$$AST = UTIL - UTIL(zero\ setups) \tag{6}$$

It was implicitly assumed that the resource has enough capacity for all setups and processing of products. The N_i for each of the i products would then be determined as follows:

a. For every product i, set N_i to 1.
b. Calculate the Total Setup Time, TST, for all the products from
$TST = \sum_{i=1}^{n} N_i . ST_i$
c. While $TST < AST$
    i. For every product i, calculate BS_i
    ii. For every product i, calculate RL_i
    iii. Select product i with the largest RL_i
    iv. Check if $ST_i < (AST - TST)$ is true
        - If true, increase N_i by 1
        - Return to step b
        - Else, move to the product with the next largest RL_i
d. Set $k = \max(N_i)$
e. Iterate from $j = 1\ to\ k$
    i. For every product i with $N_i \geq j$, add a batch of i to the time window
    ii. Advance j by 1

There are two main issues of interest in this algorithm. First is the determination of the maximum level of utilisation. Second is the issue of inventory holding cost associated with the mix, which has been omitted altogether in the algorithm. The initial algorithm focuses on increasing the product mix coming from the upstream to the downstream as much as possible. This addresses the issue of product variety mix quite well. But it has implicitly made an assumption that each of the products has the same contribution to the overall system profit and holding cost. We opine that this may not always be true. Also, the motivation for the maximum utilisation level being set at $UTIL = \sqrt{UTIL\ (zero\ setups)}$ could be questioned. We, therefore, decided to extend the model by introducing the

unit holding cost and unit profit margin for each product i. These would be used in the determination of: (i) the maximum level of resource utilisation and; (ii) the choice of the product to select for reduction of lot size. We then show later that the procedure followed in Hopp [6] is a special case of our more generalised algorithm.

4. Presentation of the modified Algorithm

For the modification proposed, we defined the following variables. Let
C_{TH_i} be the profit earned from selling a unit of output i
C_{OE_i} be the inventory cost per unit product i per unit time
μ_i be the Bottleneck Rate of product i on the resource per unit time (e.g. per hour) and
WHC_i be the Window Holding Cost (WHC) for each of the products during the chosen time window. This is mathematically defined as

$$WHC_i = RL_i.\frac{C_{OE_i}}{2} \tag{7}$$

$UTIL_{Min}$ be the minimum level of utilisation of the resource given the setup and processing times. This is defined as

$$UTIL_{Min} = UTIL(zero\ setups) + \frac{\sum_{i=1}^{n} ST_i}{Available\ Time} \tag{8}$$

Assuming the system follows an $M/M/1$ queue pattern, perform the following calculations

$$\beta_i = \frac{d_i}{\sum d_i}, \qquad \mu = \sum_{i=1}^{n} \beta_i \mu_i, \qquad C_{TH} = \sum_{i=1}^{n} \beta_i C_{TH_i}, \qquad C_{OE} = \sum_{i=1}^{n} \beta_i C_{OE_i}, \qquad \rho^* = 1 - \sqrt{\frac{C_{OE}}{\mu C_{TH}}} \tag{9a - 9e}$$

The revised algorithm is presented next.

4.1. Revised algorithm and analysis of its time complexity

a. Calculate β_i, μ, C_{TH}, C_{OE}, and ρ^* from equations 9a to 9e.
b. Compute $UTIL_{Min}$ from equation 8.
c. If $UTIL_{Min} \geq \rho^*$, terminate
d. Else, set $UTIL = \rho^*$
e. For every product i, set N_i to 1.
f. Calculate the Total Setup Time, TST, for all the products from
   $TST = \sum_{i=1}^{n} N_i.ST_i$
g. While $TST < AST$
   i. For every product i, calculate BS_i
   ii. For every product i, calculate RL_i
   iii. For every product i, calculate WHC_i
   iv. Select product i with the largest WHC_i
   v. Check if $ST_i < (AST - TST)$ is true
      - If true, increase N_i by 1
      - Return to step b
      - Else, move to the product with the next largest WHC_i
h. Set $k = \max(N_i)$
i. Iterate from $j = 1\ to\ k$
   i. For every product i with $N_i \geq j$, add a batch of i to the time window
   ii. Advance j by 1

The algorithm is of the order of $O(mn)$, where n is the number of products and m is an integer that depends on the ratio of AST to ST_is. So, it is expected that it would be computationally efficient. Two main modifications have been made to the initial algorithm. The first is that the maximum utilisation has been based on an equation derived from an optimal flow rate equation as a function of the Bottleneck rates of the products through the resource, the unit holding costs and the unit profit margins for each of the products. Also, the selection of the product whose batch is to be reduced is based not only on the run length for each of the products, but also on the holding cost of each of the products. The effects of both the run length and the unit holding cost has been merged into a single variable called

the window holding cost. This addresses both the need for the availability of more product varieties as well as the need to minimise the system holding cost via the individual holding costs of the products.

5. Illustrative Example

To illustrate how the algorithm works, a four product example used in Hopps [6] has been reproduced here. The data is presented in table 1 below. This is done for the purpose of comparison. Some arbitrary holding cost values were assigned to the four products involved, and the total holding cost for the window period, in this case one month, was calculated for the two approaches (i.e. with and without holding costs).

We have included the last two rows. The unit process time is the inverse of the process rate, so it follows from row 2. We also included arbitrary values of holding cost for products 1 to 4. The proposed lot-sizing and other relevant parameters using the initial and the revised algorithms are given in tables 2 and 3. Table 2 sets the maximum utilisation as the square root of the processing time. This approach is found to under-estimate the time available for setup if the holding cost or $UTIL_{min}$ is low, for instance. Increased setup from using the optimal flow rate, ρ^*, leads to further batch, and hence cost, reduction. The system holding cost in table 2 using initial and revised algorithms came to 15,180 and 14,880 respectively. In table 3 that utilises optimal ρ, it came down to 9,325.7 and 9,180 respectively. Using optimum ρ, thus, leads to more drastic savings. This could be well justified once other resources, like labour, needed for such reduction are in spare supply as is typical of many Lean and TOC environments.

Table 1: Data for multiproduct sequential batching example

	Basic	**Standard**	**Deluxe**	**Supreme**
Demand (units/month)	15000	12000	500	250
Process rate (units/hr)	100	100	75	50
Setup Time (hr)	8	8	6	4
Demand/time rate (hr/month)	150	120	6.7	5
Holding cost	2	3	9	8
Unit processing time (hr)	0.01	0.01	0.013333	0.02
Unit Profit Margin	4	5	10	12

Table 2: Lot sizing with the maximum utilisation set as the square root of processing time

	Holding cost not applied					**Holding cost applied**			
	Basic	**Standard**	**Deluxe**	**Supreme**		**Basic**	**Standard**	**Deluxe**	**Supreme**
Number of set ups	5	6	3	1		7	5	2	1
Total Setup time	2400	2880	1080	240		3360	2400	720	240
Batch size	3000	2000	166.6667	250		2142.857	2400	250	250
Run length	2280	1680	493.3333	540		1765.714	1920	560	540
Window Holding Cost	2280	2520	2220	2160		1765.714	2880	2520	2160

6. Conclusion

An algorithm is proposed that simultaneously determines the lot sizes, number of batches and schedules of multiple products scheduled on a production resource in a pull environment. The algorithm sets the maximum utilisation level of the non-bottleneck resource based on the unit profit margin, unit holding cost and bottleneck rate for the products on the resource. This is used to determine the time available for increased setup and reduced batch sizes. A heuristic for determining the number of batches and lot sizes for each of the products is then determined based on the window holding cost for each of the products is then developed. Numerical illustration shows that including both the holding cost and the utilisation parameter, ρ, leads to lower inventory cost and smaller batch sizes, especially in cases where the actual processing time required for jobs is much lower than the maximum capacity available.

Table 3: Lot sizing with the maximum utilisation set as maximum optimal flow rate

	Holding cost not applied					Holding cost applied			
	Basic	**Standard**	**Deluxe**	**Supreme**		**Basic**	**Standard**	**Deluxe**	**Supreme**
Number of set ups	3	2	1	1		2	3	1	1
Total Setup time	1440	960	360	240		960	1440	360	240
Batch size	5000	6000	500	250		7500	4000	500	250
Run length	3480	4080	760	540		4980	2880	760	540
Window Holding Cost	3480	6120	3420	2160		4980	4320	3420	2160

Acknowledgements

Thanks are due to NRF for their financial support.

References

1. Potts C.N. and Van Wassenhove L.N., 1992, "Integrating Scheduling with Batching and Lot-sizing", Journal of Operations Research Society, 43(5), 395-406
2. Drexl A. And Kimms A., 1997, "Lot sizing and scheduling – surveys and extensions", European Journal of Operational Research, 99, 221-235
3. Potts C.N. and Kovalyov M.Y., 2000, "Scheduling with batching: A review", European Journal of Operational Research, 120, 228-249.
4. Webster S. And Baker K.R., 1995, "Scheduling groups of jobs on a single machine", Operations Research 43(4), 692-703
5. Rappold J.A. and Yoho K.D., 2008, "A model for level-loading production in the process industries when demand is stochastic", Production Planning and Control, 19(7), 686 – 701.
6. Hopp W.J., 2008, Supply chain science, McGraw-Hill/Irwin, New York.
7. Toledo C.F.M., de Oliveira R.R.R., de Oliveira L. and Pereira M.R., 2010, "Parallel Genetic Algorithm Approaches to Solve a Synchronised and Integrated Lot Sizing and Scheduling Problem", Proceedings of the ACM Symposium on Applied Computing, March 22 – 26, Sierre, Switzerland, 1148 – 1152.
8. Dye C-Y. And Hsieh T-P., 2010, "A particle swarm optimisation for solving joint pricing and lot sizing problem with fluctuating demand and unit purchasing cost", Computers and Mathematics with Applications (in press).
9. Pahl J. And Voß S., 2010, "Discrete Lot-Sizing and Scheduling Including Deterioration and Perishability Constraints", Advanced Manufacturing and Sustainable Logistics, Lecture Notes in Business Information Processing, 46(4), 345 – 357.
10. Condotta A., Knust S. and Shakhlevich N., 2010, "Parallel batch scheduling of equal-length jobs with release and due dates", journal of Scheduling (in press).
11. Konstantaras I. and Skouri K., 2010, "Lot sizing for a single product recovery system with variable setup numbers", European Journal of Operational Research, 203, 326 – 335.
12. Chen M., Sarin S.C. and Peake A., 2010, "Integrated lot sizing and dispatching in wafer fabrication", Production Planning and Control, 21(5), 485 – 495.
13. Srinivasan M.M. and Viswanathan S., 2010, "Optimal work-in-process inventory levels for high-variety, low-volume manufacturing systems", IIE Transactions, 42, 379 – 391.
14. Dolgui A., Eremeev A.V., Kovalyov M.Y. and Kuznetsov P.M., 2010, "Multi-product lot sizing and scheduling on unrelated parallel machines", IIE Transactions, 42, 514 – 524.
15. Narayanan A. and Robinson P., 2010, "Evaluation of joint replenishment lot-sizing procedures in rolling horizon planning systems", International Journal of Production Economics, 127, 85 – 94.
16. Aryanezhad M.B., Badri S.A. and Rashidi K.A., 2010, "Threshold-based method for elevating the system's constraint under theory of constraints", International Journal of Production Research, 48(17), 5075 – 5087.
17. Webster S., 2008, Principles and Tools for Supply Chain Management, McGraw-Hill/Irwin, New York.