Optimization of a Multiproduct CONWIP-based Manufacturing System using Artificial Bee Colony Approach

Saeede Ajorlou, *Member, IAENG,* Issac Shams, *Member, IAENG,* and Mirbahador G. Aryanezhad

***Abstract*—In this paper, A Mixed Integer Non-linear Programming Model (MINLP) is developed to simultaneously generate an optimal sequence of jobs and WIP level in a serial CONWIP production line in order to minimize the overall completion time. Artificial bee colony algorithm , a novel heuristic optimization approach, is proposed to solve this model. Unlike many existing approaches, which are based on deterministic search algorithms such as nonlinear programming and mixed integer programming, the proposed method does not use a linearized or simplified model of the system. A numerical example is used to test the developed model and algorithm. Computational results validate the modeling and computational efficiency of the solution method.**

***Index Terms*—Artificial bee colony, Mixed Integer Non-linear Programming, CONWIP.**

I. Introduction

FLOWING materials through the manufacturing system is one of the most important decisions that manufacturing companies are now facing to improve their excellence. Material flow control is to address the problems of when and how much to allow parts to be processed at each station in order to achieve acceptable customers service level while minimizes Work-In-Process.

The focus of this study is on CONWIP production control systems. Similar to Kanban systems, CONWIP uses cards to manage the number of WIPs. However, there is only one set of cards flowing backward from the end of the production line to its beginning in order to precisely monitor current inventory level of the system under study. It is assumed that as long as all required manufacturing modules are accessible, the requested demands are taken into account for early production. Inasmuch as no job can enter the system without its related card, once completed in the last station, the card is released and then sent back again to the first station, where it is attached to the subsequent job to be processed. Obviously, the system is identical to Kanban in that the production of the first workstation is also activated by the demand. In contrast, it differs from Kanban system in the sense that CONWIP is only pulled between the last and the first workstation, so it may be considered as a single-stage Kanban. Because of this reason, as stated in [1], the implementation, modeling and optimization for CONWIP is much easier than Kanban. For more details about CONWIP systems and its differences with Kanban see, e.g., [2], [3], [4], [5], [6] and [7].

In order to effectively set up a CONWIP system in a specific manufacturing environment, some common issues have to be addressed, most importantly are included forecasting the backlog list (which gives the sequence of orders to be introduced into the line), determining the number of cards, and sequencing the jobs in the system. Due to some of its important merits, such as flexibility and robustness in dynamic and uncertain environments, CONWIP production control system has been applied not only to various manufacturing firms but also to different echelons of a supply chain in recent years [8]. Different aspects of the CONWIP system such as operation, applicability, and also comparisons of CONWIP with other production systems can be found in the literature, and are well classified by [9]. [10] use CONWIP in a merging/assembly line and also make some comparisons with the single Kanban system. It is shown there that, with the use of both analytical and simulation models, CONWIP surpasses single Kanban in case of variable processing times. [3] propose a new method, namely statistical throughput control (STC), which uses real-time data to adjust WIP level under a make-to-order CONWIP protocol subject to environmental changes. This study can be categorized as card controlling [9], which deals with devising some rules in order to maintain or change the current number of cards with respect to certain events such as abrupt changes in the demand.

[11] propose a mathematical model to address the optimal number of cards and job sequencing simultaneously in a multi-cell, multi-family production environment with different routes, and solve it via a simulated annealing (SA) heuristic. Moreover, they compare two variations of CONWIP control policies, the multi-loop CONWIP system (in which containers are restricted to stay in given cells) and the single-loop CONWIP system (in which containers can circulate everywhere within the system), and show the superiority of the latter in all scenarios through simulations. [12] suggest a new procedure for card controlling which can be applied both to make-to-stock and make-to-order environments, and examine it on the CONWIP system. The procedure deals with adding or subtracting extra cards along with consistently monitoring throughput rate or service level. In order to reach a target throughput rate, they use only two parameters, namely, the initial Kanban cards of the CONWIP system and the number of maximum (initial) extra cards. They also demonstrate that their method is robust pertaining to the values of the required parameters. In [13] a deterministic mathematical programming model is developed for a multi-product CONWIP flowshop system in order to find the optimal job order and schedule, given demand and forecasted rate of throughput, via linear programming (LP). The goal of this model is to minimize total cost function consisting of finished goods holding cost, shortage cost, WIP holding cost, and overtime cost. However, they do not consider lot sizes and the effects of bottleneck machine on job orders, and

do not propose an algorithm to efficiently solve their model. [14] study Sikorsky Aircraft, a single CONWIP job shop production line, and develop a mathematical programming model in order to minimize weighted penalties on tardiness and earliness at a given WIP level. The model is partially solved by dynamic programming (DP) and heuristic methods with the help of Lagrangian relaxation.

[15] consider an assembly station feeding by two parallel fabrication lines. The model is partially linearized and a nonlinear mixed integer programming algorithm is proposed in order to obtain simultaneously the optimal job sequencing and lot sizes. However, WIP level and number of containers are not discussed in their study. [16] formulate a multi-phase multi-product CONWIP control system with continuous-time Markov chain (CTMC) in a steady state mode. Their approach can effectively estimate the probability distribution of transient solution, number of WIP as a function of time as well as some important performance measures such as the average time in system, utilization percent, etc. [17] uses simulation experiment to investigate the impact of integration of Advance Demand Information (ADI) with pull systems to improve operation efficiencies. It is shown that the performance of underlying system is more responsive to order cancellation and less sensitive to variability in demand information lead times.

Considering the high complexity of production system models, typically a simplified model is instead used to find the optimal WIP level and job sequencing in a CONWIP system. Linearizing the model, and considering a single bottleneck machine to be able to model the unbalanced workload constraints are a few examples of such simplifying assumptions [18], [15], [13]. However, even the corresponding optimization problem for the simplified models are often NP-complete ([19]), and hence are intractable for close to real world problems which deal with large number of parts, machines, and production lines. Also less attention is paid in the literature to simultaneously finding the optimal WIP level and job sequencing due to this complexity. To overcome these shortcomings, we apply the artificial bee colony algorithm (ABC) [20], a novel heuristic optimization approach, to simultaneously finding the optimal WIP level and job sequencing to minimize the overall makespan time in a multi-product multi-machine serial production line under a CONWIP protocol. The highly nonlinear dynamics of the system is modeled via a production line simulator implemented on MATLAB, and is used to evaluate the candidate solutions. Numerical examples validate the efficacy of the proposed approach even for systems of large size.

The remainder of this paper is organized as follows. The problem formulation and model development are introduced in Section II. The artificial bee colony algorithm and its application in solving the problem described in Section III. The efficiency of the proposed method is verified via numerical examples in Section IV. Finally, concluding remarks are drawn in Section V.

II. Problem Formulation

The CONWIP system considered in this section is a single-stage production line with a number of machines in a sequence in which each machine can process a certain number of dissimilar part types.

There is a process time corresponding to each pair of (machine, part type), which may differ from part to part for a certain machine. Also, there is a set up time required to change the line from processing one part type to another type. It is also assumed that the line uses a CONWIP production control strategy. There is also a demand list determining the number of required parts of each type. Furthermore, the following assumptions are made in developing the mathematical programming model for the given problem:

- There is no machine breakdown
- Set up times and process times are known and deterministic
- Parts follow the same routing process at all machine and are processed on each machine sequentially

The objective function of the mathematical programming model is to minimize the total makespan time while at the same time the WIP and job sequencing are determined. To the best of the authors' knowledge and as pointed out in the Introduction section, card setting and job sequencing are treated separately in the literature. However, it is beneficial to consider these two problems at the same time since they both affect the performance of the CONWIP system.

Known Parameters

- n the number of different part types
- P_i part type i, $i = 1, \ldots, n$
- m the number of machines in the line
- d_i the number of required parts of type P_i in the demand list, $i = 1, \ldots, n$
- T_{ij} the setup time required to switch the line from processing part type P_i to part type P_j, $i, j = 1, \ldots, n$, and $i \neq j$. $T_{ij} = 0$ for $i = j$
- P_{ij} the processing time of machine i on part type P_j, $i = 1, \ldots, m$ and $j = 1, \ldots, n$

Decision Variables

$$y_{ij} = \begin{cases} 1, & \text{if } P_i \text{ is followed by } P_j,\ i, j = 1, \ldots, n, \text{ and } i \neq j \\ 0, & \text{otherwise} \end{cases}$$

The decision variables y_{ij} should satisfy following constraints:

$$\sum_{\substack{j=1 \\ j\neq i}}^{n} y_{ij} = 1, \quad i = 1, \ldots, n \tag{1}$$

$$\sum_{\substack{i=1 \\ i\neq j}}^{n} y_{ij} = 1, \quad j = 1, \ldots, n \tag{2}$$

Using these notations, the optimization problem can be formulated as

$$\min \sum_{i=1}^{n} \sum_{\substack{j=1 \\ j\neq i}}^{n} y_{ij} T_{ij} \tag{3}$$

The objective function in (3) corresponds to the setup time required to switch from processing the current part type to the next one for a specific job sequencing.

$$|W_k - W_{k-1}| < \epsilon_k, \quad k = 2, \ldots, n \tag{4}$$

where W_k and W_{k-1} are the line workloads on the kth and $(k-1)$th part batches. Calculating W_k is highly complicated and depends on many parameters such as the part processing times P_{ij}, setup times T_{ij}, WIP level, demand list, and

ISBN: 978-988-19251-2-1
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

job sequencing. Deterministic search algorithms, such as integer programming require simplified (and usually linear) estimates for W_k. This can be done by assuming that the line work load on a part batch is determined by the bottleneck machine in the line, and there is only one bottleneck station. Under these assumptions W_k can be estimated as

$$W_k = \sum_{i=1}^{n} x_{ik}(d_i P_i^b + \sum_{\substack{j=1 \\ j \neq i}}^{n} y_{ij} T_{ij}) \tag{5}$$

where P_i^b denotes the process time of part P_i by the bottleneck machine and

$$x_{ik} = \begin{cases} 1, & \text{if the } k\text{th part batch is of type } P_i \\ 0, & \text{otherwise} \end{cases} \tag{6}$$

However, heuristic search algorithms such as the one employed in the present work, do not require such a simplified formula and can easily handle problems involving complicated nonlinear models. This is one of the advantages of the present work over the existing approaches to CONWIP production system.

III. Artificial bee colony

In an artificial bee colony algorithm, there are three groups of bees: employed bees, onlooker bees, and scout bees. A bee going to the food source which is visited by itself in the last round is called an employed bee. An onlooker bee, on the other hand, waits on the dance floor to gather information about the positions and the nectar (fitness) of the food sources from the employed bees and then chooses one of them probabilistically. A bee doing a random search for food sources is called a scout bee. Employed and onlooker bees are responsible for exploitation part of the search while the scout bees carry out the exploration part. The number of food sources is equal to the number of employed bees; in other words, for any food source there exists a unique employed bee. The number of onlooker bees is assumed to be the same as that of employed bees. An employed bee corresponding to an exhausted food source becomes a scout bee. A food source (i.e. a solution) is assumed to be exhausted if its quality is not improved after a certain number of cycles called *limit*. The employed and onlooker bees also do a local search in the neighborhood of their food sources and switch to them if they have more nectar (higher fitness).

The detailed steps taken in a typical artificial bee colony algorithm are listed below:

- Initialize the population of the solutions x_i
- Evaluate the population using the fitness function
- While the maximum cycle number is not reached
- Produce new random solutions v_i in the neighborhood of the existing solutions (x_i) for the employed bees
- Replace x_i with the newly generated solution v_i if its fitness is higher than x_i
- Assign probabilities to the solutions x_i according to their finesses
- Assign a solution to each onlooker bee based on the probabilities of x_i's, and produce random solutions v_i for the onlookers in the neighborhood of x_i's
- Replace x_i in the memory of an onlooker bee with v_i, if its fitness is higher than x_i
- Determine an abandoned solution and replace it with a new randomly generated solution for the scout bee
- Memorize the best solution found so far
- End While

The probabilities using which the onlookers choose food sources x_i are calculated as

$$p_i = \frac{fit(x_i)}{\sum_{j=1}^{n} fit(x_j)} \tag{7}$$

where n is the size of the population and $fit(x_i)$ is the fitness of solution x_i.

To solve this problem using the artificial bee colony algorithm, we fix the WIP level in each run and then we use an artificial bee colony search to find the optimal job sequencing. Each solution x_i is a vector in which $x_i(j)$ represents the jth part type to be processed in the production line. Assuming the number of part types being n, the space of possible solutions is of size $n!$, which makes the search intractable using deterministic search methods for large values of n. The function $f(x_i)$ to be minimized is the makespan time corresponding to the job sequencing x_i. We assign the following fitness function $fit(x_i)$ to any solution x_i

$$fit(x_i) = \frac{1}{1 + f(x_i)} \tag{8}$$

This fitness function will be used in evaluating the population and the associated probabilities for every solution according to (7). A very important step in any ABC algorithm is how to choose a random solution v_i in the neighborhood of x_i. In the case where the search space for x_i is continuous, this is simply done by randomly choosing a solution x_k and letting v_i to be a randomly selected point on the segment between x_i and x_k. However, this method is not applicable to the discrete search spaces, which makes the problem more challenging. In this paper we employ the following algorithm to construct v_i from x_i

- $j = 1$
- While ($j \leq n$)
- Generate a random number $p \in [0, 1]$
- If $p > p_{th}$ or $x_i(j) \in \{v_i(1), \ldots, v_i(j-1)\}$, then choose $v_i(j)$ randomly from $\{1, \ldots, n\} \backslash \{v_i(1), \ldots, v_i(j-1)\}$, otherwise, set $v_i(j) = x_i(j)$
- $j = j + 1$
- End While

In the above algorithm, p_{th} is a threshold used to keep v_i close to x_i. It is clear that to keep v_i closer to x_i, p_{th} should be chosen close enough to 1.

We run this algorithm for every possible WIP level varying from 1 to the size of the demand list to find the optimal job sequencing in each case. Finally we choose the job sequencing with the minimum WIP level associated with the minimum makespan time.

IV. Numerical Examples

In this section, we consider a single serial CONWIP production line with 3 machines, producing 6 part types. All 6 part types are processed by 3 machines sequentially, and it is assumed that all the assumptions mentioned in Section II hold. Processing times on 3 machines for 6

ISBN: 978-988-19251-2-1
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

TABLE I
PART PROCESSING TIMES

	P_1	P_2	P_3	P_4	P_5	P_6
Machine1	12	15	12	13	12	18
Machine2	9	16	4	7	15	5
Machine3	19	6	7	12	5	17

TABLE II
SEQUENCE DEPENDENT SETUP TIMES

T_{ij}	$j=1$	$j=2$	$j=3$	$j=4$	$j=5$	$j=6$
$i=1$	0	12	15	8	4	6
$i=2$	13	0	11	5	12	18
$i=3$	19	8	0	5	6	3
$i=4$	15	6	14	0	13	7
$i=5$	3	8	15	6	0	5
$i=6$	12	9	4	6	10	0

TABLE III
DEMAND LIST

	P_1	P_2	P_3	P_4	P_5	P_6
Demand	9	4	6	8	7	9

products are shown in Table I. Sequence dependent set up times are also generated randomly with uniform distribution over $\{1, \ldots, 20\}$. These numbers are shown in Table II, with T_{ij} denoting the set up time required to change from processing part type P_i to processing part type P_j (for instance, T_{12} in this production line is 12 time units). Finally, Table III shows the number of parts to be produced through the production line.

For any WIP level, the search space for finding the optimal job sequencing has 6! elements. The control parameters of the ABC algorithm are chosen as follows. The colony size is set to be 20. The maximum number of cycles is 1000 and the limit value is 7. The threshold probability is chosen to be $p_{th} = 0.6$.

The optimal job sequencing and makespan time for various WIP levels are shown in Table IV. As can be seen, up to some point, any increase in the WIP level leads to a decrease in the makespan time in the optimal job sequencing. However, at some point, increasing WIP level will not improve the makespan time of the system (WIP level equal to 9 in this example). This is the best WIP level in the sense that its corresponding optimal job sequencing results in the minimum possible makespan time, and using a higher WIP level cannot improve the performance of the system.

V. CONCLUSIONS

This paper considers a CONWIP-based multi-product multi-machine serial production line. An artificial bee colony approach is proposed to simultaneously find the optimal WIP level and job sequencing to minimize the overall makespan time for the developed mathematical model of the system. Unlike many approaches in the literature, the proposed method treats both WIP level optimization and job sequencing at the same time. Moreover, it does not use simplifying assumptions or linearized model of the

TABLE IV
MAKESPAN TIME AND OPTIMAL JOB SEQUENCING FOR VARIOUS WIP LEVELS

WIP level	Makespan	Optimal job sequencing
1	1512	$P_1P_5P_3P_6P_4P_2$
2	787	$P_1P_5P_3P_6P_4P_2$
3	697	$P_5P_1P_2P_4P_6P_3$
4	666	$P_2P_4P_1P_6P_5P_3$
5	648	$P_1P_6P_4P_2P_5P_3$
6	639	$P_6P_1P_2P_4P_5P_3$
7	635	$P_6P_5P_1P_2P_4P_3$
8	632	$P_1P_5P_2P_4P_6P_3$
9	629	$P_2P_4P_6P_1P_5P_3$
10	629	$P_2P_4P_6P_1P_5P_3$

production system, and utilizes a production line simulator developed on MATLAB to model the nonlinear complex dynamics of the system and to calculate the fitness for candidate solutions. The proposed method is also applicable to real world problems involving large number of parts, machines, and production lines, as illustrated by numerical simulations.

REFERENCES

[1] E. G. A. Gaury, H. Pierreval, and J. P. C. Kleijnen, “An evolutionary approach to select a pull system among kanban, conwip and hybrid,” *Journal of Intelligent Manufacturing*, vol. 11, no. 2, pp. 157–167, 2000.

[2] S. Gstettner and H. Kuhn, “Analysis of production control systems kanban and conwip,” *International Journal of Production Research*, vol. 34, no. 11, pp. 3253–3274, 1996.

[3] W. J. Hopp and M. L. Roof, “Setting wip levels with statistical throughput control (stc) in conwip production lines,” *International Journal of Production Research*, vol. 36, no. 4, pp. 867– 882, 1998.

[4] R. P. Marek, D. A. Elkins, and D. R. Smith, “Understanding the fundamentals of kanban and conwip pull systems using simulation,” in *Proceedings of the Winter Simulation Conference*, 2001, pp. 922–929.

[5] J. Geraghty and C. Heavey, “A comparison of hybrid push/pull and conwip/pull production inventory control policies,” *International Journal of Production Economics*, vol. 91, no. 1, pp. 75–90, 2003.

[6] I. Lee, “Evaluating artificial intelligence heuristics for a flexible kanban system: simultaneous kanban controlling and scheduling,” *International Journal of Production Research*, vol. 45, no. 13, pp. 2859–2873, 2007.

[7] J.-W. Li, “Simulation study of coordinating layout change and quality improvement for adapting job shop manufacturing to conwip control,” *International Journal of Production Research*, vol. 48, no. 3, pp. 879–900, 2010.

[8] G. Knolmayer, P. Mertens, and A. Zeier, *Supply Chain Management Based on SAP Systems*. Springer, 2002.

[9] J. M. Framinana, P. L. Gonzleza, and R. Ruiz-Usanoa, “The conwip production control system: review and research issues,” *Production Planning and Control*, vol. 14, no. 3, pp. 255–265, 2003.

[10] M. Lambrecht and Segaert, “Buffer stock allocation and assembly type production lines,” *International Journal of Operations and Productions Management*, vol. 10, no. 2, pp. 47–61, 1990.

[11] B. Golany, E. M. Dar-EL, and N. Zeev, “Controlling shop floor operations in a multi-family, multi-cell manufacturing environment through constant work-in-process,” *IIE Transactions*, vol. 31, no. 8, pp. 771–781, 1999.

[12] J. Framinan, P. L. Gonzalez, and R. Ruiz-Usan, “Dynamic card controlling in a conwip system,” *International Journal of Production Economics*, vol. 99, no. 1–2, pp. 102–116, 2006.

[13] Y. T. Herer and M. Masin, “Mathematical programming formulation of conwip-based production lines and relationships to mrp,” *International Journal of Production Research*, vol. 35, no. 4, pp. 1067–1076, 1997.

[14] P. B. Luh, X. Zhou, and R. N. Tomastik, “An effective method to reduce inventory in job shops,” *IEEE Transactions on Robotics and Automation*, vol. 16, no. 4, pp. 420–424, 2000.

[15] D. Cao and M. Y. Chen, “A mixed integer programming model for a two line conwip-based production and assembly system,” *International Journal of Production Economics*, vol. 95, no. 3, pp. 317–326, 2005.

[16] M. D. Al-Tahat and I. A. Rawabdeh, “Stochastic analysis and design of conwip controlled production systems,” *Journal of Manufacturing Technology Management*, vol. 19, no. 2, pp. 253–273, 2008.

[17] D. Claudio and A. Krishnamurthy, “Kanban-based pull systems with advance demand information,” *International Journal of Production Research*, vol. 47, no. 12, pp. 3139–3160, 2009.

[18] N. Li, T. Z. Mike, S. Deng, Z. H. Lee, L. Zhang, and L. Zheng, “Single-station performance evaluation and improvement in semiconductor manufacturing: A graphical approach,” *International Journal of Production Economics*, vol. 107, no. 2, pp. 397–403, 2007.

[19] M. R. Garey and D. S. Johnson, *Computers and Interactability: A guide to the theory of NP-completeness*. Freeman, 1979.

[20] D. Karaboga and B. Basturk, “A powerful and efficient algorithm for numerical function optimization: artificial bee colony (abc) algorithm,” *Journal of Global Optimization*, vol. 39, no. 3, pp. 459–471, 2007.

ISBN: 978-988-19251-2-1
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)