Overtime Capacity Expansion in Order Acceptance with Node Based Estimation of Distribution Algorithms

Watcharee Wattanapornprom[1,3], Tieke Li[1], Warin Wattanapornprom[2,3] and Prabhas Chongstitvatana[2]
[1]Dongling School of Economics and Management, University of Science and Technology Beijing, Beijing, China
[2]Department of Computer Engineering, Faculty of Engineering, Chulalongkorn University, Bangkok, Thailand
[3]Department of Research and Development, Thai Ocean Industries Co., Ltd., Bangkok, Thailand
(wwatcharee@gmail.com)

Abstract – **Order acceptance with overtime capacity expansion requires trading off between over and under capacity utilization in order to gain more profits. This research proposes an overtime capacity utilization order acceptance model and proposes adaptations of node based estimation of distribution algorithm to solve the order acceptance decisions in multi-process environments. The results show that node based coincidence algorithm is a potential algorithm which can maximize both profit and can maximize the capacity utilization at the same time.**

Keywords – **order acceptance, overtime capacity, genetic algorithm, node based coincidence algorithm, node histogram based sampling algorithm**

I. INTRODUCTION

Overtime (OT) is a temporary capacity expansion strategy, which inevitably used in many industries. OT is generally used to handle the insufficient capacity caused by uncertainties in production processes such as defects, miscalculation plans or fluctuation of demands. OT certainly increases costs and expenses. Higher OT usage can reflect more inefficient productivity. Under high labor incentive and high competition circumstances, a manufacturer cannot just invest on capacity expansion in order to serve the momentary exceeding of demands. Subsequently, OT becomes more and more irrefutable.

Process uncertainty is one of the important cause of OT. It usually results in idle time which is incapable to utilize. Consequently, the more expanded capacity causes the more idle time. Without idle time restriction constraint, no matter how much manufacturers expand the capacity, they cannot gain actual maximize utilization. Generally speaking, the profits gained from capacity expansion would be too far deviated from the expectation.

Order acceptance (OA) problem [1] is a combination of knapsack and bin packing problems where sequences of orders also affect the results of selected orders. This NP-hard problem has gained increasing attention amongst the manufacturers. It plays important roles in decision-making process between selling and manufacturing divisions. Exceeding of demands under capacity constraints conditions make manufacturers necessitate to pick the most profitable sequence of orders, while maintaining the competitive production costs. Effective OA appears to be one of the most important key performance indicators as it reflects the incoming profits, and determines costs of the future manufacturing processes.

Recent papers [2][3] proposed to include in-house OT and outsourcing as additional capacity into OA problems in order to gain more profits. They suggested that maximizing capacity utilization does not always result in maximum profit. Conversely, from the maximum profit, manufacturers can compensate the cost of OT and outsourcing while retaining the profits level.

In order to efficiently select orders, using the appropriate objective functions is important. The objective functions of OA should be consistent with the manufacturing objectives. However, these objectives can be conflicted to each other. The objective functions in OA can be divided into two points of view. From the accepting perspectives, the main objectives rather depend on the maximization of profits. On the other hand, from the rejecting perspectives, the main objectives rather depend on the minimization of costs. Profits maximization can reflect the costs minimization. For example, minimizing make span can result in increasing of capacities utilization. Practically, it is not necessary to calculate both profits and costs consecutively. However, in many situations, manufacturers could not raise the capacity level to support more orders to gain more profits. Manufacturers typically ask their employee to work overtime which inevitably require higher production cost. This research add monetary and time related penalty into OA model, and illustrated cost-effective solution to improve in-house productivity by comparing with the result of OT costs.

In OA, trading off between under and over utilization is not a new concept, it was used in microeconomic models for pricing and selecting the profitable orders under the time constraints [4][5][6]. However, there exists a research gap in trading off between under capacity and over capacity utilization. This is the first research that uses two conflicting capacity constraint as the objectives in OA to select and sequence the orders to maximize the profits while maintaining the balance of the production capacities at the same time.

This article introduces the application of node-based estimation of distribution algorithms (EDA) [7][8] for solving OA problem in multi-process production line. The contribution of this work is to demonstrate new approaches to the OA problems that compete successfully with

978-1-4799-6410-9/14/$31.00 ©2014 IEEE

previously purposed genetic algorithm especially in larger problems. This research also presents a new perspective of expanding the capacity to gain more profits instead of just spending OT to handle the problems.

The remaining sections of this paper are organized as follows. The OA model and the procedures are introduced in Section II. The results are discussed in Section III. Finally, Section IV concludes this work.

II. METHODOLOGY

A. Order Acceptance Model

In practice, most production lines are balanced such that they are fit with regular products that have greater demands. Each process utilizes the balanced capacity such that similar kind of products can be produced smoothly. Unfortunately, strength of these production lines are dropped when producing infrequently ordered products, it results in inefficient use of working capacity. Therefore, the prices of lower demand products become higher in order to compensate the unusual production times and materials in the stocks. The unused capacities become inevitable costs that the manufacturers have to spend. However, the manufactures can use the OT capacities in some processes to accept more profitable orders and to avoid the under capacities penalty. The characteristic of OT capacity expansion in order acceptance problem is shown in figure 1.

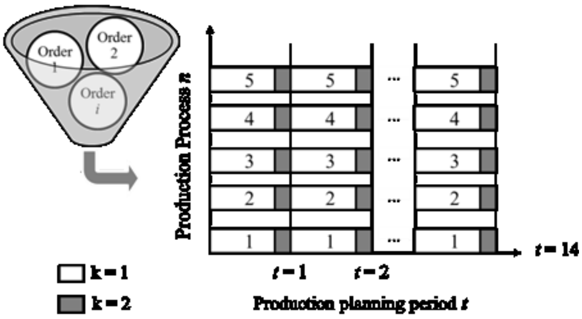


Fig.1.Characteristic of Overtime Capacity Expansion in Order Acceptance Problem

(1) Notation

i Order number (i=1,2,…, I)

j Job number (j=1,2,…, J)

n Process sequence number (n =1,2,…,5)

t Production planning period (t=1,2,…,T)

k Production time period， k=1; normal working time period, k=2; overtime period (k=1,2)

(2) Parameters

p_i Profit of order i

q_i Demand quantity of order i

D_i Due date of order i

g_n Production cost per time unit at process n

α_{1n} OT cost at process n

α_{2n} Leftover capacity cost at process n

d_{nt}^{+} Over used capacity in production planning period t at process n

d_{nt}^{-} Leftover capacity in production planning period t at process n

CTP_{knt} Unallocated capacity of production time period k in production planning period t at process n

e_{ijn} Need to consume CTP_{knt} quantity of order i by process j of process n

f_{ijknt} Time unit that CTP_{knt} of process n require to produce order i by process j at production planning period t in production time period k

(3) Decision parameters

F_{ijktn} $=$ 1 , if at production planning period t, order i was accepted to produce by process j in production time period k of process n
$=$ 0, otherwise

R_i $=$ 1, if the order i *is* accepted
$=$ 0, otherwise

One of the classic problems in the multi-processes manufacturing that produces variety of products is that the production lines need to expand the capacities d_{nt}^{+} while there remains a lot of leftover capacities d_{nt}^{-}. To avoid this problem, firms have to make clear that the being accepted orders do not disrupt each other's time and materials. Generally, each order i has different product, quantity q_i, profit p_i and due date D_i. The order acceptance decision making system has to determine that the order i should be produced at which production resource k, on which period t. As well as, job j of the accepted order i, the require capacity $e_{ijn}q_i$ at process n, should not be larger than the amount of unallocated capacity $\sum_k \sum_t CTP_{knt}$. The cost of expanded capacities d_{nt}^{+} and leftover capacities d_{nt}^{-} are transformed into wage g_n. The cost of leftover capacities can be calculated from the common operational wages while the cost of expanded capacities can be calculated from the OT wages. In order to calculate the costs, α_{1n} and α_{2n} are used to represent the cost rate of the wage g_n ($\alpha = (d_{nt}$'s cost$)\,/g_n$).

The key objectives of this problem are (i) to maximize the gross profit and (ii) to minimize the costs of leftover and extended capacities. In order to achieve the first objective, the order acceptance decision and expand capacity decision are taking into account to find out that which unallocated capacity CTP_{knt} that each process n has left, which orders should be accepted and how much capacity to be expanded. To archive the second objective, the model keep balancing capacity usage in-between production process n based on production costs. Therefore, the model can help manufacturers to select and sequence the set of profitable orders using as much working capacity CTP_{knt}, as well as less OT working time d_{nt}^{+} as possible. The model thus can be defined as follows:

Maximize $Z = \sum_i p_i R_i - (\sum_n \alpha_{1n} d_n^{+} + \alpha_{2n} d_n^{-}) g_n$ (1)

s.t.

$$\sum_i \sum_j e_{ijn} q_i \times R_i \leq \sum_k \sum_t CTP_{knt} \qquad \forall n \qquad (2)$$

$$d_n^+ = \sum_t \sum_i \sum_j e_{ijnt} q_i F_{ij|K|tn} \qquad \forall n \qquad (3)$$

$$d_n^- = \left(\sum_t \sum_k^{k-1} CTP_{knt}\right) - \sum_t \sum_i \sum_j \sum_k^{k-1} e_{ijn} q_i F_{ijktn} \quad \forall n \qquad (4)$$

$$f_{ijknt} \geq F_{iktn} \qquad \forall i, j, k, n, t \qquad (5)$$

$$f_{ijknt} \leq e_{ijn} q_i F_{ijktn} \qquad \forall i, j, k, n, t \qquad (6)$$

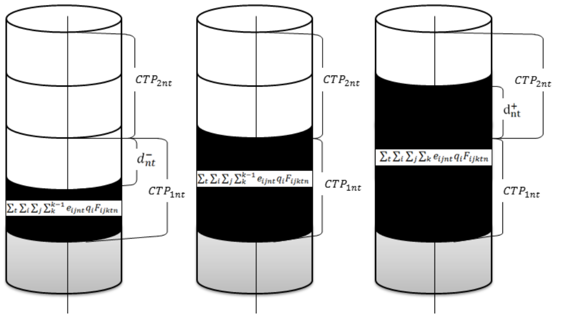
$$\sum_n tF_{i|J|knt} \leq D_i R_i \qquad \forall i, k, t \qquad (7)$$

$$\sum_k \sum_{t'=1}^{t-1} f_{i(j-1)knt'} + \sum_k f_{i(j-1)knt} \geq e_{i(j-1)n} q_i F_{ijktn} \qquad \forall i, j \backslash \{1\}, k, n, t \quad (8)$$

$$R_{ik} \quad = \{0,1\} \qquad \forall i, k \qquad (9)$$

$$F_{ijknt} = \{0,1\} \qquad \forall i, j, k, n, t \qquad (10)$$

$$f_{ijknt} \geq 0 \qquad \forall i, j, k, n, t \qquad (11)$$


CTP_{2nt}
d_{nt}^-
CTP_{1nt}
$\sum_t \sum_i \sum_j \sum_k^{k-1} e_{ijnt} q_i F_{ijktn}$
CTP_{2nt}
$\sum_t \sum_i \sum_j \sum_k^{k-1} e_{ijnt} q_i F_{ijktn}$
CTP_{1nt}
CTP_{2nt}
d_{nt}^+
$\sum_t \sum_i \sum_j \sum_k e_{ijnt} q_i F_{ijktn}$
CTP_{1nt}

(2.1) (2.2) (2.3)

Fig. 2. Calculation the amount of surplus and stretch capacity

The first set of constraints (2) is established to ensure that the whole capacity of plant is not violated. Constraint (3) and (4) are set to calculate the penalty of not using the whole capacity and stretching the production capacity. Constraints (5) and (6) set the F_{ijknt} decision variables to either 1 or 0. The F_{ijknt} variables are the indicator variable; they take a value of 1 when $f_{ijknt} > 0$ indicating that process *j* of item *i* is being processed on resource *k* in the period *t*, otherwise they take a value of 0. The F_{ijknt} variables are used to sequence the precedence connections between processes. The constraint set (7) ensures that the completion time of the last process of the accepted orders do not exceed the due dates. The constraint sets (8) imposes precedence restrictions, to ensure that process *j* of item *i* can be processed in period *t* only after completing process *j* of item *i-1.* The constraints (9) and (10) are the binary constraints and constraint (11) is the negativity constraints.

B. Solution Procedures

This work compares the result of GA with two node based EDAs including NHBSA, NB-COIN. From preliminarily study, the results of the EDAs for solving OA in single machine are too far better than GA and its benchmarks [9], therefore this paper only compare the results with GA.

1. Genetic Algorithm

The first procedure is an ordering GA with Position-based crossover (PBX) [10] which preserve not only absolute order substructures but also preserve relative order substructures from two parents.

For this problem, the chromosomes are sample solutions, that is, sequenced subsets of orders. The diversity is maintained by ancestor replacement. If new candidate is better than its ancestors it is used to replace one of its own parents. In this study, the local search is also applied to the new candidates with improvement. The swapping and insertion operations are randomly applied to the candidates until the candidates are no longer improved. The pseudo code of GA are as follows:-

```
Step 1. Randomly generate the population.
Step 2. Evaluate the population.
Step 3. Perform crossover and mutation. If the newly
        generated candidate is better than its ancestors, then
        perform the local search until the candidate is no
        longer improved.
Step 4. Repeat Step 3 until the maximum number of
        generation is reached.
```

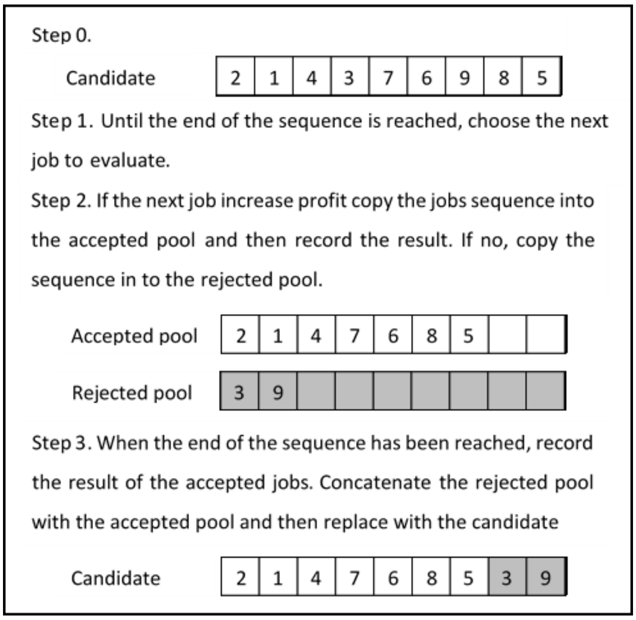
Even though, the encoded solution of GA is a full set of the orders in the pool, however, the evaluation process considers only the accepted orders. The evaluation process does not only evaluate the orders sequence, but also re-sorts the sequences of orders to separate the accepted and rejected orders as illustrated in the figure 3. The sequence of the accepted orders are kept in the accepted pool while the remaining orders are kept in the rejected pool. The candidate solution is re-sorted by concatenating the accepted pool with the rejected pool.

2. Estimation of Distribution Algorithms

The EDAs used in this research are Node Histogram Based Sampling Algorithm (NHBSA) and Node Based Coincidence Algorithm (NB-COIN). They generate solution strings in sequences, ensuring that only valid permutations are sampled. The differences of these two node based EDAs are the learning methods. NHBSA belongs to the ad hoc learning methods, while NB-COIN belongs to the incremental learning methods. The pseudo code of EDAs are simplified as follows:-

```
Step 1. Initialize the model
Step 2. Sample the population
Step 3. Evaluate the population
Step 4. Select candidates
Step 5. Update the model
Step 6. Repeat steps 2 to 5 until terminated.
```

Although, GA and EDAs are in the same group of evolutionary algorithms, however, the evaluation process and the updating process of EDAs for the order acceptance are slightly different. GA needs to maintain the genetic materials, therefore the whole set of orders need to be maintained. However, EDAs can reproduce the missing sequences by themselves, in addition, the sequences of the rejected pool are considered to be the useless information, therefore, EDAs only update the models from the accepted sequences of orders. Consequently the evaluation process doesn't need to concatenate the rejected pool with the accepted pool. The evaluation processes in the figure 3 simply use the accepted pool as the candidate for the EDAs.



Step 0.

Candidate 2 1 4 3 7 6 9 8 5

Step 1. Until the end of the sequence is reached, choose the next job to evaluate.

Step 2. If the next job increase profit copy the jobs sequence into the accepted pool and then record the result. If no, copy the sequence in to the rejected pool.

Accepted pool 2 1 4 7 6 8 5

Rejected pool 3 9

Step 3. When the end of the sequence has been reached, record the result of the accepted jobs. Concatenate the rejected pool with the accepted pool and then replace with the candidate

Candidate 2 1 4 7 6 8 5 3 9

Fig. 3. Evaluation with cutting off. [9]

2.1. Node Histogram Based Sampling Algorithm

NHBSA was proposed by Tsutsui in 2006.[7] It utilizes Node Histogram Matrix (NHM) to learn the mutual information of absolute position. Matrix $NHM = [h_{ij}]$, where $h_{ij} = P(\sigma_i = j)$ and $i, j \in \{1, 2, \ldots, n\}$. Hence, h_{ij} represents the probability of the index j to be in the *i*-th position in the selected individuals. h_{ij} is added to a ε value denoted as

$$\varepsilon = \frac{N}{n} B_{ratio} \qquad (12)$$

to control the pressure in sampling and to avoid individuals with probability 0.

2.4. Node Based Coincidence Algorithm

NB-COIN [8] is a variation of coincidence algorithm (COIN) [11] proposed by Wattanapornprom and Chongstitvatana in 2013. It learns substructures from absolute positions, similar to NHBSA. The matrix H_{xy} represents the probability of y found in the absolute position x. The update equation is

$$H_{xy}(t+1) = H_{xy}(t) + \frac{k}{(n)}\left(r_{xy}(t+1) - p_{xy}(t+1)\right) + \frac{k}{(n)^2}\left(\sum_{j=1}^{n} p_{xj}(t+1) - \sum_{j=1}^{n} r_{xj}(t+1)\right), \qquad (13)$$

where *k* denotes the learning step, *n* is the problem size, r_{xy} is the number of xy found in the better-group, and p_{xy} is the number of xy found in the worse-group. The incremental and detrimental step is $\frac{k}{(n-1)}$, and the term $\frac{k}{(n-1)^2}\left(\sum_{j=1}^{n} p_{xj}(t+1) - \sum_{j=1}^{n} r_{xj}(t+1)\right)$ represents the adjustment of all other H_{xj}, where $j \neq x$ and $j \neq y$.

NB-COIN has a special characteristic, that is, it not only learns from the good samples but also learns from the poor samples as well. After each population was evaluated and ranked, two groups of candidates are selected according to their fitness values: better-group and worse-group. The better-group is selected from the top $c\%$ of the rank and is used as a reward, and H_{xy} is increased for every pair of xy found in this group. The punishment is a decrease in H_{xy} for every pair of xy found in the worse group of the bottom c$\%$ of the population rank.

C. Test Problems and Experimental Design

The list of products and their profit per piece were randomly generated. The generated profits are ranged between 5 to 15 currency units per piece. Then these profit attributes were used to generate the capacity used for each product such that producing the least profitable product would utilize the most balance capacity in each working process. Random time were added according to their profits. The capacities used by each processes are ranged between 0.1 to 1 pieces per minute.

Ten problems of size 50, 75 and 100 orders with and without OT were also randomly generated according to the products and their profits such that the less profitable products have more chance to be ordered. Each order was generated from a log-normal distribution with an underlying normal distribution with mean 0 and standard deviation 1. The quantities for each order were randomly generated using the range between 1×1000 pieces and 12×1000 pieces. Each product has to be processed through 5 processes × 5 parallel workstations. Each workstation has one worker. The maximum capacity were set to two weeks (14 working days). Each working day has 8 working hours plus extra 2 OT hours. The due dates of each order were generated from a uniform distribution plus calculated lead-time for each of the order. These parameters were imitated from the existent manufactures in Thailand. Therefore, the wage penalty for this problem was set to 300 baht and OT cost was set to 450 baht per one workstation per day.

To compare the results, each algorithm was given the same population sizes and maximum number of generations which are equal to the problem size × 2. The probabilities of crossover and mutation of GA are equal to 0.8 and 0.2 respectively. The learning steps, k, of NB-COIN is 0.05. The bias ratio, B_{ratio}, of NHBSA is 0.005. The selection pressure of GA and NHBSA is 50% of the whole population, while NB-COIN uses 25% of the top ranks for rewards and 25% of the bottom ranks for punishment. Test programs were coded in Lazarus and ran on Mac OS 10.11 on Intel Pentium Core i7 2.60 GHz processor with 8 GB of RAM.

The performances of GA and EDAs are compared in terms of average of the best actual profits and percentage of over and under capacity utilization.

III. RESULTS

Table I compares the performance of the benchmark algorithms. The capacity utilization is the wage penalty already deducted from the actual profit. Figure 4 compare the gained profit in a problem with 50 orders. Since the solutions of NB-COIN and NHBSA were generated from generation to generation, without keeping the elitists, the best solution in each generation does not necessary increasing.

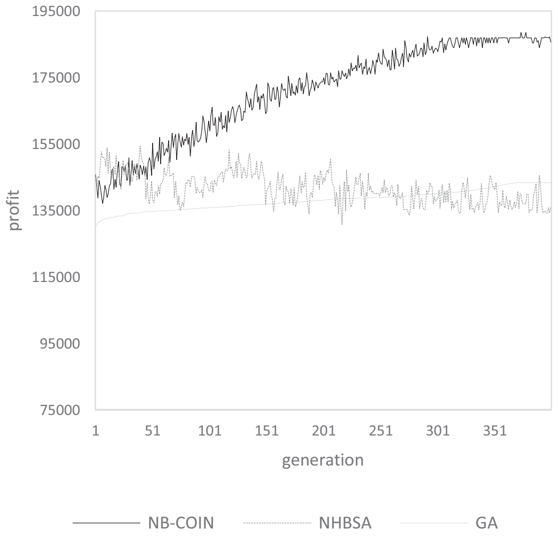


Fig. 4. Performance of NB-COIN, NHBSA and GA in maximizing the profit in the OA with OT capacity expansion problems.

From the table I, it can be clearly seen that the node based EDAs yield better results compared to GA with local search. NB-COIN can find the best solution in every benchmark as it can seek for sequences of subset orders which gain the best profits. In addition, NB-COIN can utilize the full capacity of the working hours. It can also find the set of profitable orders which could utilize more OT capacity. However, from figure 4, NHBSA can find better solutions than NB-COIN in the very beginning generations. It can find competitive solutions with less number of function evaluations. Unfortunately, NHBSA was trapped in some pitfalls whereas it cannot combine the solutions with higher profits such that satisfy the orders due dates and capacity utilizations. The generated test problems were design such that the lowest profitable product utilize the most balanced capacity. On the other hand, the most profitable product causes the most capacity leftover. The greedier profit maximization would results in the worse capacity utilization.

It is not necessary for the manufactures to utilize all of the OT capacity in order to get the best profits, especially in mix-model multi-process production lines, which are not easy to re-balance. Consequently, it is necessary to choose the series of orders that are not only profitable but also balance in capacity utilization.

The unique characteristic of NB-COIN is that it not only learns from the good solutions, but also learns from the poor solutions. This characteristic enables NB-COIN to find not only the good quality solutions, but also the diverse of solutions [12], which is the fundamental characteristic to solve multimodal and multi-objective problems. The incremental learning method enables NB-COIN to maintain the high potential substructure to be composed. NB-COIN simply estimated the sequence of the accepted orders in which the good sequences of orders may be conflict with each other.

IV. CONCLUSION

In case of unpredictable capacity-lacking, extending the current production capacity by using OT can help manufacturers accept more orders and keep the good relationship with customers. This paper presented the new idea to accept more orders, in which the objective is not only to gain high profit, but also to utilize capacity usage effectively and keep overall production lines well-balanced. A new extended mathematical OA model is proposed in order to enable OT capacity expansion. This work uses node based EDAs to demonstrate the trading off between over and under capacity utilization. The gross profit maximization, surplus and overtime cost minimization were set as objectives. From the empirical study, NB-COIN can seek for sequences of orders that can make use of OT to gain more profits. Besides the capacity balancing objective not only keep the production line balanced, but also effectively improve the profits significantly.

TABLE I
PERFORMANCE OF GA, NHBSA AND NB-COIN IN ORDER ACCEPTANCE WITH AND WITHOUT OVERTIME CAPACITY

problem size (orders)	GA+LS			NHBSA			NB-COIN		
	profit (baht)	% under capacity	% over capacity	profit (baht)	% under capacity	% over capacity	profit (baht)	% under capacity	% over capacity
50	134370	45.6	0	148047	33.9	0	185605	25.3	0
50 + OT	143303	12.8	0	154378	9.56	0	188586	0	3.49
75	152562	42.5	0	150553	31.7	0	194074	19.3	0
75 + OT	153556	12.2	0	163649	8.43	0	195075	0	5.67
100	161686	38.4	0	174005	29.1	0	210095	14.5	0
100 + OT	164657	11.4	0	178939	7.34	0	213732	0	6.78

REFERENCES

[1] S.A. Slotnick ,2011. “Order acceptance and scheduling: A taxonomy and review”, *European Journal of Operational Research 210(3), pp.527-536,2011*

[2] C.S. Chen, S. Mestry, P. Damodaran and C. Wang “The capacity planning problem in make-to-order enterprises” *Mathematical and computer modelling*, 50(9), pp. 1461-1473. 2009

[3] S. Mestry, P. Damodaran and C.S. Chen “A branch and price solution approach for order acceptance and capacity planning in make-to-order operations” *European Journal of Operational Research* 211(3), pp. 480-495. 2011

[4] F.H. de B. Harris and J.P. Pinder “A revenue management approach to demand management and order booking in assemble-to-order manufacturing” , *European Journal of Operations Management 13(4), pp. 299–309, 1995*

[5] A.J. Kleywegt, J.D.Papastavrou, 2001. “The dynamic and stochastic knapsack problem with random sized items.” *Operations Research 49 (1), pp.26–41.,2001*

[6] X.L. Zhong, J.W. Ou, G.Q. Wang “Order acceptance and scheduling with machine availability constraints” *European Journal of Operational Research 232, pp.435–441,2014*

[7] S. Tsutsui, “Node histogram vs. edge histogram: A comparison of probabilitstic model-building genetic algorithms in Permutation Domains,” in *Proc. of the IEEE Congress on Evolutionary Computation (IEEE CEC 2006)*, pp. 1939-46, 2006.

[8] O. Srimongkolkul, and P. Chongstitvatana. "Application of Node Based Coincidence algorithm for flow shop scheduling problems." In *Computer Science and Software Engineering (JCSSE), 2013 10th International Joint Conference on*, pp. 49-52. IEEE, 2013.

[9] W .Wattanapornprom, T.K. Li, W.Wattanapornprom ,P.Chongstitvatana “Application of Estimation of Distribution Algorithms for Solving Order Acceptance with Weighted Tardiness Problems ” *Industrial Engineering and Engineering Management (IEEM), 2011 IEEE International Conference on*. IEEE, 2011.

[10] G. Syswerda, “Schedule Optimization Using Genetic Algorithms” in *A Handbook of Genetic Algorithms*, 1991.

[11] W. Wattanapornprom, P. Olanviwitchai, P. Chutima, and P. Chongstitvatana, “Multiobjective combinatorial optimisation with coincidence algorithm,” in *Proc. of IEEE Congress on Evolutionary Computation (IEEE CEC 2009)*, pp. 1675-82, 2009.

[12] K Waiyapara, P Chongstitvatana, "Solving Multimodal Problems by Coincidence Algorithm." in *Proc. of International Joint Conference on Computer Science and Software Engineering (JCSSE 2012)*, 2012.