International Journal of Mechanical, Industrial and Aerospace Sciences
ISSN: 2517-9950
Vol:11, No:3, 2017

A Genetic Algorithm Based Permutation and Non-Permutation Scheduling Heuristics for Finite Capacity Material Requirement Planning Problem

Watchara Songserm, Teeradej Wuttipornpun

***Abstract*—**This paper presents a genetic algorithm based permutation and non-permutation scheduling heuristics (GAPNP) to solve a multi-stage finite capacity material requirement planning (FCMRP) problem in automotive assembly flow shop with unrelated parallel machines. In the algorithm, the sequences of orders are iteratively improved by the GA characteristics, whereas the required operations are scheduled based on the presented permutation and non-permutation heuristics. Finally, a linear programming is applied to minimize the total cost. The presented GAPNP algorithm is evaluated by using real datasets from automotive companies. The required parameters for GAPNP are intently tuned to obtain a common parameter setting for all case studies. The results show that GAPNP significantly outperforms the benchmark algorithm about 30% on average.

***Keywords*—**Finite capacity MRP, genetic algorithm, linear programming, flow shop, unrelated parallel machines, application in industries.

I. Introduction

THE FCMRP is developed to remedy a shortcoming of material requirement planning (MRP) that assumes constant production lead-time. It has become an interesting research topic for a few decades as shown in the literature [1]-[9]. Based on the literature, the metaheuristic algorithm is one of the most popular techniques used to solve this problem since it obtains a near optimal solution within a practical computational time [10]-[14]

This paper presents a GAPNP for the FCMRP problem. It is intently developed to improve the solution obtained from the benchmark algorithm by [15]. Our objective is to minimize the total cost, which is the sum of tardiness, earliness and flow-time costs.

This paper is further organized as follows. Details of the GAPNP algorithm are explained in Section II. The experiments for evaluating the performance of GAPNP and our case studies are explained in Section III. Results and discussions are provided in Section IV. The conclusion and recommendation of this paper are given in the last section.

II. The Proposed GAPNP Algorithm

In the proposed GAPNP, each chromosome consists of y genes, where y is the number of customer orders. The chromosome is constructed by permutation encoding concept. There are 5 main steps of GAPNP as follows: (1) construct initial chromosomes (population), (2) evaluate performance of each chromosome, (3) selection, (4) apply genetic operators (crossover and mutation), and (5) apply the LP model. Note that steps 2-4 are repeatedly performed until the stopping criterion is satisfied, and finally step 5 is performed to minimize the total cost. The overall procedures of GAPNP are shown in Fig. 1 and explained follows.

Watchara Songserm and Teeradej Wuttipornpun are with Department of Industrial Engineering, King Mongkut's University of Technology North Bangkok, Bangkok 10800, Thailand (e-mail: wsongserm@gmail.com, teeradejw@kmutnb.ac.th).

```
overall procedure: GAPNP algorithm
input: orders, BOMs and machines information, GAPNP parameters
output: solution from the GAPNP algorithm
begin
    n←0; //n: generation number
//Step 1: Construct initial population
    create Population(n) consists of Y chromosomes; //Y: population size
    while (not terminating condition of GAPNP) do
//Step 2: Evaluate performance of each chromosome
        Each chromosome is evaluated its performance through the total cost
//Step 3: Selection
        apply elitist method to Population(n);
        select Parent(n) from Population(n) by roulette wheel method;
//Step 4: Apply genetic operators
        crossover Parent(n) to get Offspring(n);
        mutate Offspring(n) to get MOffspring(n); //MOffspring: mutated offspring
        Population(n+1)←a sequence from elitist method and MOffspring(n);
        calculate total cost of Population(n+1);
        n←n+1;
    end;
    select the sequence with minimum total cost from Population(n);
//Step 5: Apply the LP model
    apply LP to the sequence from step 4;
output: solution from the GAPNP algorithm
end;
```

Fig. 1 Pseudo code for GAPNP

A. Construct Initial Population

The initial population consists of Y chromosomes (sequences of orders), where Y is a population size. We construct them by random rule and due date-based dispatching rules called EDD and MST. The reason for selecting EDD and MST is that they obtain favorable results in the benchmark algorithm [15].

B. Evaluate Performance of Each Chromosome

Each chromosome is evaluated through the total cost calculated by (1), where Q_i, t_i, e_i and f_i are the order quantity, tardiness, earliness and flow-time of the i^{th} order. T_i, E_i and F_i are the cost per unit of tardiness, earliness and flow-time of the i^{th} order.

$$Total\ \cos t = \sum_{i=1}^{n} T_i Q_i t_i + \sum_{i=1}^{n} E_i Q_i e_i + \sum_{i=1}^{n} F_i Q_i f_i \quad (1)$$

To calculate the total cost, all operations of orders in the sequence must be scheduled to machines in order to calculate the tardiness, earliness and flow-time. It can be done by the following steps: (1) the orders in the sequence are exploded by variable lead-time MRP (VMRP) to obtain details of operations, (2) these operations are scheduled to less tardiness machines by permutation and non-permutation scheduling options, and (3) the tardiness, earliness and flow-time of each order are calculated by the start time, due time and completion time. Note that the permutation and non-permutation schedules are applied separately for the entire steps.

Details of the required operations of the four orders after VMRP explosion shown in Fig. 2 are used to demonstrate this step. Suppose that the population size is 3 sequences of orders (chromosomes), which are O_1 ➪ O_3 ➪ O_4 ➪ O_2 (EDD), O_1 ➪ O_3 ➪ O_2 ➪ O_4 (MST) and O_4 ➪ O_3 ➪ O_2 ➪ O_1 (random). When the permutation option is selected, all operations must be scheduled based on the sequence of orders in a way that the operations of the first order in the sequence are scheduled to less tardiness machines first, and the operations of the second order in the sequence are scheduled next and so on. By this way, it is obvious that all machines have the same sequence of operations that complies with the permutation sequencing concept. When the non-permutation schedule is selected, the operations are scheduled without considering the sequence of orders. It results in different sequence of operations of each machine that complies with the non-permutation sequencing concept. The total costs of these sequences are shown in Table I. Note that the result of non-permutation concept may be the same as that of the permutation concept when there are only a few orders in the sequence as shown in our illustration. However, it does not happen to our case studies.

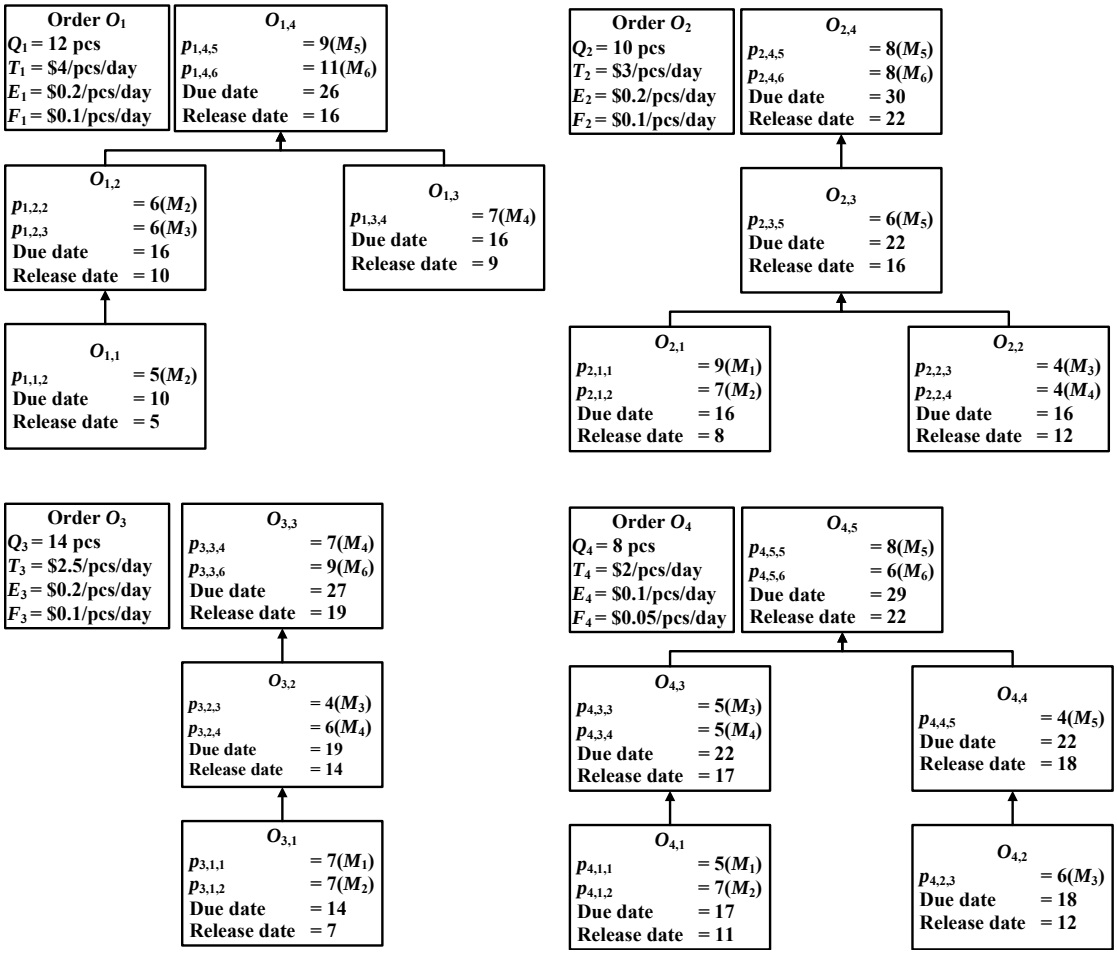

Fig. 2 Details of operations after applying the VMRP explosion

C. Selection

There are two selections called elitist and roulette wheel methods applied in this step. The elitist method is applied to keep the best sequence of orders from each iteration and use it as a new chromosome for a new population. On the other hand, the roulette wheel method selects the parent sequences based on the probability in a way that the sequence with higher probability has higher opportunity to be selected than the lower one. Note that only the permutation option is used in our illustration.

Based on the permutation schedule in Table I, the sequence of orders O_1 ➪ O_3 ➪ O_4 ➪ O_2 is selected by the elitist method

to be a member of new population because it has the minimum total cost. The roulette wheel is applied to select a pair of parent sequences for constructing two offspring in next step. To calculate the probability for roulette wheel, the inverse total cost is first determined and then the probability of each sequence is calculated by (2). The probability of each sequence and its interval are shown in Table II. Suppose that two random numbers are 0.35 and 0.87, the sequences $O_1 \Rightarrow O_3 \Rightarrow O_4 \Rightarrow O_2$ and $O_4 \Rightarrow O_3 \Rightarrow O_2 \Rightarrow O_1$ are then selected to be the parent sequences.

$$Probability(i) = \frac{Inverse\ total\ cost\ of\ sequence\ i}{Sum\ of\ inverse\ total\ cost\ of\ all\ sequence} \quad (2)$$

D. Apply Genetic Operators

This step consecutively applies genetic operators called crossover and mutation to the parent sequences in order to construct the offspring. They are applied based on their probability. In this paper, two well-known crossover operators called PMX and PBX are applied since they obtain a good result for the flow shop scheduling problem [16]. Suppose that PMX is selected, the crossover probability (p_c) is 0.9, the random positions are 2 and 3, and the random number for generating the offspring is 0.46. Since the random number is less than p_c, the crossover process is then allowed and the two offspring sequences generated from PMX are $O_2 \Rightarrow O_3 \Rightarrow O_4 \Rightarrow O_1$ and $O_1 \Rightarrow O_3 \Rightarrow O_2 \Rightarrow O_4$. These two offspring sequences are then sent to the mutation process.

TABLE I
TOTAL COST OF INITIAL SEQUENCES BASED ON PERMUTATION AND NON-PERMUTATION SCHEDULES

Initial sequence	Sequence of operations on machines (Permutation)	Total cost (Permutation)	Sequence of operations on machines (Non-permutation)	Total cost (Non-permutation)
$O_1 \Rightarrow O_3 \Rightarrow O_4 \Rightarrow O_2$ (EDD)	M_1: $O_{3,1} \Rightarrow O_{4,1}$ M_2: $O_{1,1} \Rightarrow O_{1,2} \Rightarrow O_{2,1}$ M_3: $O_{3,2} \Rightarrow O_{4,2} \Rightarrow O_{4,3}$ M_4: $O_{1,3} \Rightarrow O_{3,3} \Rightarrow O_{2,2}$ M_5: $O_{1,4} \Rightarrow O_{4,4} \Rightarrow O_{2,3} \Rightarrow O_{2,4}$ M_6: $O_{4,5}$	62.15	M_1: $O_{3,1} \Rightarrow O_{4,1}$ M_2: $O_{1,1} \Rightarrow O_{1,2} \Rightarrow O_{2,1}$ M_3: $O_{3,2} \Rightarrow O_{4,2} \Rightarrow O_{4,3}$ M_4: $O_{1,3} \Rightarrow O_{3,3} \Rightarrow O_{2,2}$ M_5: $O_{1,4} \Rightarrow O_{4,4} \Rightarrow O_{2,3} \Rightarrow O_{2,4}$ M_6: $O_{4,5}$	62.15
$O_1 \Rightarrow O_3 \Rightarrow O_2 \Rightarrow O_4$ (MST)	M_1: $O_{3,1} \Rightarrow O_{2,1}$ M_2: $O_{1,1} \Rightarrow O_{1,2} \Rightarrow O_{4,1}$ M_3: $O_{3,2} \Rightarrow O_{2,2} \Rightarrow O_{4,2}$ M_4: $O_{1,3} \Rightarrow O_{3,3} \Rightarrow O_{4,3}$ M_5: $O_{1,4} \Rightarrow O_{2,3} \Rightarrow O_{2,4} \Rightarrow O_{4,4}$ M_6: $O_{4,5}$	75.45	M_1: $O_{3,1} \Rightarrow O_{2,1}$ M_2: $O_{1,1} \Rightarrow O_{1,2} \Rightarrow O_{4,1}$ M_3: $O_{3,2} \Rightarrow O_{2,2} \Rightarrow O_{4,2}$ M_4: $O_{1,3} \Rightarrow O_{3,3} \Rightarrow O_{4,3}$ M_5: $O_{1,4} \Rightarrow O_{2,3} \Rightarrow O_{2,4} \Rightarrow O_{4,4}$ M_6: $O_{4,5}$	75.45
$O_4 \Rightarrow O_3 \Rightarrow O_2 \Rightarrow O_1$ (RND)	M_1: $O_{4,1}$ M_2: $O_{3,1} \Rightarrow O_{2,1} \Rightarrow O_{1,1} \Rightarrow O_{1,2}$ M_3: $O_{4,2} \Rightarrow O_{3,2} \Rightarrow O_{2,2}$ M_4: $O_{4,3} \Rightarrow O_{3,3} \Rightarrow O_{1,3}$ M_5: $O_{4,4} \Rightarrow O_{2,3} \Rightarrow O_{2,4}$ M_6: $O_{4,5} \Rightarrow O_{1,4}$	131.45	M_1: $O_{4,1}$ M_2: $O_{3,1} \Rightarrow O_{2,1} \Rightarrow O_{1,1} \Rightarrow O_{1,2}$ M_3: $O_{4,2} \Rightarrow O_{3,2} \Rightarrow O_{2,2}$ M_4: $O_{1,3} \Rightarrow O_{4,3} \Rightarrow O_{3,3}$ M_5: $O_{4,4} \Rightarrow O_{2,3} \Rightarrow O_{2,4}$ M_6: $O_{4,5} \Rightarrow O_{1,4}$	115.05

TABLE II
PROBABILITY FOR ROULETTE WHEEL (PERMUTATION)

Initial sequence	1/Total cost	Probability	Cumulative probability	Random number interval
$O_1 \Rightarrow O_3 \Rightarrow O_4 \Rightarrow O_2$	0.016	0.43	0.43	0 - 0.43
$O_1 \Rightarrow O_3 \Rightarrow O_2 \Rightarrow O_4$	0.013	0.35	0.78	0.43 - 0.78
$O_4 \Rightarrow O_3 \Rightarrow O_2 \Rightarrow O_1$	0.008	0.22	1.00	0.79 - 1.00

There are two efficient mutation operators for the flow shop problem called INSERT and SWAP [16]. Suppose that SWAP is selected, the mutation probability (p_m) is 0.01, the random positions are 2 and 4, and random numbers for offspring sequences $O_2 \Rightarrow O_3 \Rightarrow O_4 \Rightarrow O_1$ and $O_1 \Rightarrow O_3 \Rightarrow O_2 \Rightarrow O_4$ are 0.47 and 0.006, respectively. Since the random number of the latter offspring sequence is less than p_m, the mutation process is then allowed for only this sequence. The sequence after mutation is $O_1 \Rightarrow O_4 \Rightarrow O_2 \Rightarrow O_3$. There are three sequences obtained after completing this step. The first sequence comes from the elitist method, whereas the last two sequences are obtained from the genetic operators as shown in Table III. These sequences are called a new population. If the stopping criterion is still not satisfied, they are sent to step 3 for performing the next iteration. Otherwise, the sequence with the minimum total cost will be selected and sent to next step.

TABLE III
TOTAL COSTS OF NEW POPULATION (PERMUTATION)

Method	New population	Total cost
Elitist	$O_1 \Rightarrow O_3 \Rightarrow O_4 \Rightarrow O_2$	62.15
Crossover/Mutation	$O_2 \Rightarrow O_3 \Rightarrow O_4 \Rightarrow O_1$	98.30
Crossover/Mutation	$O_1 \Rightarrow O_4 \Rightarrow O_2 \Rightarrow O_3$	76.40

E. Apply the LP Model

From the last step, the operations are scheduled to machines by the heuristics. As a result, the start times of some operations may not be optimal. This step tries to determine the optimal start times of all operations by the LP model in order to minimize the total cost. In our LP model, only the start times of operations are optimized, whereas the sequence of operations on machines obtained from the last step is still maintained. Our LP model is formulated by using indices, parameters and variables explained as follows.

Indices

i = index of order starting from 1 to n
j = index of operation of order i starting from 1 to m
j^* = index of the last operation of order i
k = index of machine starting from 1 to s
j' = index of child operations of order i operation j
q = index of sequence of operations on machine starting from 1 to t. For example, if there are three operations on a machine, when q = 2, it refers to the second operation on this machine.

Sets and Parameters

$p_{i,j}$ = processing time of operation i,j
d_i = due time of order i
CT_i = cost per unit of tardiness of order i
CE_i = cost per unit of earliness of order i
CF_i = cost per unit of flow-time of order i
Q_i = quantity of order i
Seq_k = set of sequence of operation i,j on machine k. For example, if operations $O_{3,1}$ ➪ $O_{4,1}$ are on machine 1, then Seq_1 = {(3,1), (4,1)}.
$CH_{i,j}$ = set of child operations belongs to operation i,j. For example, if $O_{1,4}$ consists of $O_{1,2}$ and $O_{1,3}$, then $CH_{1,4}$ = (2, 3).

Decision variables

$x_{i,j}$ = start time of operation i,j
c_i = completion time of order i
f_i = flow-time of order i
e_i = earliness of order i
t_i = tardiness of order i
Z = total cost

Objective Function

$$Minimize\ Z = \sum_{i=1}^{n} CT_i Q_i t_i + \sum_{i=1}^{n} CE_i Q_i e_i + \sum_{i=1}^{n} CF_i Q_i f_i \quad (3)$$

Constraints

1) Constraint to maintain the sequence of operations on each machine

$$x_{i,j}^{q+1,k} \geq x_{i,j}^{q,k} + p_{i,j} \qquad \forall k, q, \text{ and } (i,j) \in Seq_k \quad (4)$$

2) Constraint to maintain the precedence relationships of operations

$$x_{i,j} \geq c_{i,j'} \qquad \forall i, j, \text{ and } j' \in CH_{i,j} \quad (5)$$

3) Constraints to calculate completion time, tardiness, earliness and flow-time

For completion time:

$$c_i = x_{i,j^*} + p_{i,j^*} \qquad \forall i, j^* \quad (6)$$

$$c_{i,j} = x_{i,j} + p_{i,j} \qquad \forall i, j \quad (7)$$

For tardiness:

$$t_i \geq c_i - d_i \qquad \forall i \quad (8)$$

$$t_i \geq 0 \qquad \forall i \quad (9)$$

For earliness:

$$e_i = d_i - c_i \qquad \forall i \quad (10)$$

$$e_i \geq 0 \qquad \forall i \quad (11)$$

4) Constraint to maintain the precedence relationships of operations

$$x_{i,j} \geq 0 \qquad \forall i, j \quad (12)$$

III. Case Studies and Experiments to Evaluate Performance of GAPNP

A. Detail of Case Studies

There are three case studies from real automotive companies. They all have the same production shop which is the multi-stage flow shop with unrelated parallel machines. The planning horizon is one month with 8 hours a day and no time extended. There is no order splitting or preempting. However, there are some characteristics making our case studies different, for example, a number of products, number of levels in BOM, number of machines, number of operations with parallel machines, and number of machines in the parallel stages. The different characteristics are used to prove that GAPNP can be implemented to various situations.

B. Experiments for Evaluating the Performance of GAPNP

There are six parameters for GAPNP shown in Table IV. The details of these parameters are derived from our screening experiments. For the parameter tuning experiment, these parameters are considered as independent variables for a factorial experiment. The response variable is the total cost before applying the LP model. For the performance evaluation experiment, GAPNP and the benchmark algorithm are considered as independent variables for a one-way ANOVA. The response variable is the total cost after applying the LP model. Both experiments are conducted with five replicates for all case studies. The stopping criterion for each run is 2 hours.

IV. Results and Discussions

In this paper, we intent to determine the best common parameter setting rather than the best setting for an individual case study. To obtain this setting, the relative percentage deviation for each case study (*RPD*) and the average relative percentage deviation of all case studies (*ARPD*) are determined by (12) and (13), where Sol_{Alg} is the total cost obtained from each run, Sol_{Best} is the minimum total cost across all runs, v is the index of case study, and V is the number of case studies. The best common setting is a setting obtained at the minimum *ARPD*, since it guarantees that the total cost from this setting is very close to its best total cost.

$$RPD_v = \frac{Sol_{Alg(v)} - Sol_{Best(v)}}{Sol_{Best(v)}} \text{ x } 100\% \quad (13)$$

$$ARPD = \frac{1}{V}\sum_{v=1}^{V}(RPD)_v \quad (14)$$

Table V shows the best common settings for GAPNP. It can be seen that the permutation and non-permutation options have different settings. Table VI shows the best total cost and near best total of GAPNP. It is obvious that there is only a small gap less than 1% between the near best total cost and its best total cost. This proves that the best common setting works very well. It is also observed that the permutation option slightly outperforms the non-permutation option. Fig. 3 shows the solution improvement characteristic of GAPNP. It clearly shows that the permutation option reaches the steady state faster than the non-permutation option. However, both options can reach their steady state within 2 hours.

TABLE IV
PARAMETERS FOR GAPNP

Algorithm	p_s	Crossover	p_c	Mutation	p_m	Scheduling option
GA	5, 10	PMX, PBX	0.6, 0.8	SWAP, INSERT	0.005, 0.01	Permutation, Non-permutation

TABLE V
BEST COMMON PARAMETER SETTINGS FOR ALL CASE STUDIES

Algorithms	p_s	Crossover	p_c	Mutation	p_m	Min. *ARPD*
GA permutation	10	PBX	0.6	SWAP	0.005	0.78%
GA non-permutation	10	PBX	0.6	INSERT	0.01	0.95%

TABLE VI
BEST TOTAL COSTS, NEAR BEST TOTAL COSTS, AND *RPD* BEFORE APPLYING THE LP

Algorithms	Best total costs ($)/Near best total costs ($)/$RPD_{TC}$			Min. *ARPD*
	Case 1	Case 2	Case 3	
GA permutation	8,203 / 8,275 / 0.88%	13,439 / 13,634 / 1.45%	17,836 / 17,836 / 0.00%	0.78%
GA Non-permutation	8,943 / 9,072 / 1.45%	13,758 / 13,758 / 0.00%	18,501 / 18,760 / 1.40%	0.95%

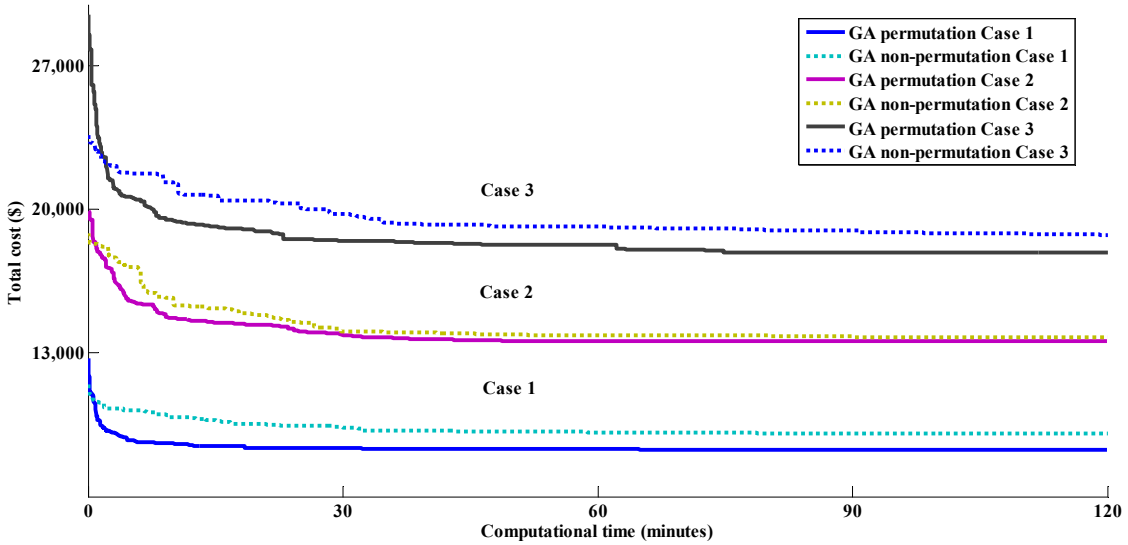

Fig. 3 Solution development characteristics of GAPNP

TABLE VII
TOTAL COSTS AND *RPI* AFTER APPLYING LP

Algorithms	Total costs ($)/*RPI*			Average
	Case 1	Case 2	Case 3	
FCMRP (2014)	9,424 (Benchmark)	16,016 (Benchmark)	19,280 (Benchmark)	14,906.58 (Benchmark)
GA permutation	7,300 / 22.54% (2)	11,960 / 25.33% (2)	13,707 / 28.91% (2)	10,988.99 / 25.59% (2)
GA Non-permutation	7,016 / 25.55% (1)	10,265 / 35.91% (1)	12,814 / 33.54% (1)	10,031.70 / 31.66% (1)

Table VII shows the total cost after applying the LP model. The numbers in parentheses are the rankings obtained from Tukey multiple comparison test. It is obvious that the near best total cost in Table VI is substantially improved when the LP model is applied. This proves that our LP model works as expected. It is observed that the non-permutation option is significantly better than the permutation option. To compare the total cost between GAPNP and the benchmark algorithm,

the relative percentage improvement (*RPI*) over the best total cost of the benchmark algorithm is applied. It is calculated from (14), where Sol_{Alg} is the total cost of each algorithm, Sol_{BM} is the best total cost from the benchmark algorithm. Based on the *RPI* value, it observes that GAPNP improves the total cost obtained from the benchmark algorithms in a range of 25.59%-31.66%.

$$RPI = \frac{|Sol_{Alg} - Sol_{BM}|}{Sol_{BM}} \text{ x } 100\% \quad (15)$$

V. Conclusion

In this study, the genetic algorithm based permutation and non-population scheduling heuristics (GAPNP) is presented for solving the FCMRP problem. It is intently developed for the flow shop with unrelated parallel machines. The objectives are to improve the total cost obtained from the benchmark algorithm by [15] and to offer a practical alternative approach for the planner. Based on our case studies, it proves that GAPNP works very well for many industrial situations. The total cost obtained from the benchmark algorithm is dramatically reduced when the GAPNP algorithm is applied.

The GAPNP algorithm has some limitations. The lot sizing policy is only lot-for-lot. Other lot-sizing techniques should be studied. Since it is especially designed for the flow shop, the performance of the proposed algorithm in other manufacturing shops such as job shop, open shop should be investigated. The production time extended (overtime) and order preemption are also of interest because they can significantly reduce the total cost. Therefore, a new algorithm should be developed to address these problems.

Acknowledgment

This research has been supported by Faculty of Engineering, King Mongkut's University of Technology North Bangkok.

References

[1] W. H. M. Zijm, and R. Buitenhek, "Capacity planning and lead time management," *International Journal of Production Economics*. vol. 46, pp. 165-179, 1996.

[2] M. Taal, and J. C. Wortmann, "Integrating MRP and finite capacity planning," *Production Planning & Control,* vol. 8, no. 3, pp. 245-251, 1997.

[3] P. C. Pandey, P. Yenradee, S. Archariyapruek, "A finite capacity material requirement planning system," *Production Planning & Control*, vol. 11, no. 2, pp.113–121, 2000.

[4] P. B., Nagendra, and S. K. Das, "Finite capacity scheduling method for MRP with lot size restrictions," *International Journal of Production Research*, vol. 39 no. 8, pp. 1603-1623, 2001.

[5] T. Wuttipornpun, and P. Yenradee, "Development of finite capacity material requirement planning system for assembly operations," *Production Planning & Control*, vol. 15, no. 4, pp. 534-549, 2004.

[6] J. Mula, R. Poler, J. P. Garcia, "MRP with flexible constraints: A fuzzy mathematical programming approach." *Fuzzy Sets and Systems,* vol. 157, no. 1, pp. 74-97, 2006.

[7] M. Vanhoucke, and D. Debels, A finite-capacity production scheduling procedure for a Belgian steel company. *International Journal of Production Research*, vol. 47 no. 3, pp. 561–584, 2009.

[8] C. Öztürk, A. M. Örnek "Capacitated lot sizing with linked lots for general product structures in job shops," *Computers & Industrial Engineering* vol. 58, no. 1, pp. 151–164, 2010.

[9] K. Thörnblad, A-B. Strömberg, M. Patriksson, and T. Almgren, "Scheduling optimisation of a real flexible job shop including fixture availability and preventive maintenance," *European Journal of Industrial Engineering*, vol. 9, no. 1, pp. 126–145, 2015.

[10] C. Moon, Y. Seo, Y. Yum, and M. Gen, "Adaptive genetic algorithm for advanced planning in manufacturing supply chain," *Journal of Intelligent Manufacturing*, vol. 17, no. 4, pp. 509-522, 2006.

[11] H. Kim, H. I. Jeong, and J. Park, "Integrated model for production planning and scheduling in a supply chain using benchmarked genetic algorithm," *International Journal of Advanced Manufacturing Technology*, vol. 39, no. 11, pp. 1207–1226, 2008.

[12] M. H. F. Rahman, R. Sarkerm and D. Essam, "A real-time order acceptance and scheduling approach for permutation flow shop problems," *European Journal of Operational Research*, vol. 247, no. 2, pp. 488-503, 2015.

[13] M. Zandieh, and N. Karimi, "An adaptive multi-population genetic algorithm to solve the multi-objective group scheduling problem in hybrid flexible flowshop with sequence-dependent setup times," *Journal of Intelligent Manufacturing*, vol. 22, no. 6, pp. 979-989, 2011.

[14] P-C. Chang, W-H. Huang, J-L. Wu, and T. C. E. Cheng, "A block mining and re-combination enhanced genetic algorithm for the permutation flowshop scheduling problem," *International Journal of Production Economics*, vol. 141, no. 1, pp. 45-55, 2013.

[15] T. Wuttipornpun, and P. Yenradee, "Finite capacity material requirement planning system for assembly flow shop with alternative work centres," *International Journal of Industrial and Systems Engineering,* vol. 18, no. 1, pp. 95–124, 2014.

[16] R. Vanchipur, and R. Sridharan, "Development and analysis of hybrid genetic algorithms for flow shop scheduling with sequence dependent setup time" *International Journal of Services and Operations Management*, vol. 17, no. 2, 2014.