A GENETIC ALGORITHM FOR DEVELOP AND IMPROVEMENT VIRTUAL CELL FORMATION SYSTEM PROBLEM

Alongkorn Muangwai[1*] and Thanida Khanongnuch[1]
[1]Faculty of Industrial Technology, Pibulsongkram Rajabhat University, Muang, Phitsanulok, Thailand, 65000
*Corresponding author E-mail: alongkorn@psru.ac.th

วันที่เข้าระบบ 2 พฤศจิกายน 2563
วันที่แก้ไขบทความ 4 ธันวาคม 2563
วันที่ตอบรับบทความ 5 ธันวาคม 2563

Abstract

This paper presents a genetic algorithm (GA) for solving a cell formation problem in virtual cellular manufacturing (VCM) environment which is appropriate for system with various product types and flexible demand changes. Several manufacturing factors are considered, such as machine capacity, inter-cell movement of parts, number of cells and cell size restriction. The objectives of the problem are to maximize number of machine hour, minimize number of shared machines and workers between cells. A preliminary study of the algorithm is performed.

Keywords: Virtual cellular manufacturing system, Manufacturing cell, Genetic algorithm

1. Introduction

The concept of Cellular Manufacturing System (CMS) has received great attention among researchers in the past decade. Applying the notion of Group Technology on the shop floor, several benefits of CMS have been realized such as reduced work-in-process, shorter production lead-time, better worker collaboration, better material handling and increased product quality. The basic notion of CMS is to group set of machines and place them in close proximity in the same area called “cell” which ideally can process set of jobs within these areas. Contrary to job shop manufacturing environment, CMS is appropriate for manufacturing situation where product mix is stable since, in this system, machines have been positioned to gain advantages of particular product mix. Once the product mix has changed, the current cell configuration can no longer provide such aforementioned benefits. Therefore, in the environment with frequent product mix changes, CMS may not an effective concept.

In recent years, increasing global competitiveness has forced manufacturers to react to customer demand more rapidly which results in shorter product life cycle and frequent product mix changes. To cope with such environment, a relatively new alternative was proposed by McLean et al. (McLean *et al.*, 1982) namely Virtual Cellular Manufacturing (VCM) systems. In VCM, machines are not physically repositioned into cells, instead they are grouped logically under production planning and control context. In other words, machines in a cell do not necessarily occupy a contiguous area on the shop floor (Nomden, 2011). Apart from machine location issue, another difference between VCM and CMS is the resource sharing concept. In traditional CMS, parts should be processed entirely with a cell and movements of parts to other cells are considered ineffectiveness. Because inter cell movement are not favored, the duplication of resources is common in CMS which could lead to low utilization of machines. However, in VCM, machines are allowed to be shared amongst more than one cell. Hence the cost of machine duplication and low utilization of machine can be greatly reduced by Ko & Egbelu (2003). As mentioned earlier, there are a great number of works related to VCM in the literature. Nomden *et al.* (2006) provided an excellent review on the subject and also proposed future research issues.

Similar to CMS, one of the steps required for implementing VCM is cell formation which assigns machines to their cells. Because of the machine sharing concept, virtual cell formation problems are more complex than its CMS counterpart. Ko & Egbelu (2003) proposed a cluster-based heuristic for designing virtual manufacturing cell. The algorithm can be applied without the need to obtain the pre-specified number of cells. (Nomden *et al*., 2006) developed a goal programming model for a virtual cell formation problem which not only considering grouping and sharing of machines but also those of labor. They also proposed a two stage approach providing a feasible solution for the model.

The aim of this paper is to develop a genetic algorithm (GA) for a virtual cell formation problem which is partly based on the model proposed by Nomden *et al.* (2006). A preliminary study is then performed with experiment study to configure most suitable GA parameters, such as mutation rate, crossover rate and population sizes.

The organization of this paper is as follows. In the next section, a goal programming based model is presented. The implementing procedure for the proposed GA for the model is then discussed. Three problems are then tested in the preliminary study to evaluating the efficiency of GA. In the last section, finally, a conclusion is followed.

2. Objective Function

2.1 Maximize number of machine hour

2.2 Minimize number of shared machines and workers between cells

3. Methodology

3.1 Problem Definition

The model for the problem was proposed by Nomden *et al.* (2006). The objective of the problem is to maximize production hours for a particular planning period while minimizing the number of machines and workers that are grouped to be in more than one cell (excess machines and workers). Consider a set of job {I} that need to be operated on the shop floor. Each job has particular routing information which determines machines that are required to process the job. The time required for job I to be operated on machine m is denoted by t_{im}. Each job belongs to its corresponding family f. The set of job belonging to family f is denoted by S_f. The setup time required for a job in S_f on machine m is S_{fm}. The number of machine available on the shop floor is given by θ_m. To operate and setup machines, workers are needed. Parameter Φ_{lm} specifies whether worker l can operate machine m. The notation of parameters and decision variables are presented in Table 1.

Table 1. Notation

Indices:

i = job i = 1; 2; . . . I;

f = family f = 1; 2; 3; . . . F;

l = worker l = 1; 2; 3; . . .L;

m = machine type m = 1; 2; . . .M;

c = virtual cell c = 1; 2; . . .C;

Parameters:

J_f = set of jobs belonging to family f

α_c= effectiveness of cell c to schedule jobs of one family sequentially at a machine (varies between 0 and 1); this factor depends on β and the size of cell c

S_{fm} = major setup time required for family f over machine type m

θ_m= number of type m machines available

t_{im}= processing time for job i on machine type m

R = length of planning period

Φ_{lm}=1 if worker l is able to perform machine type m; =0 otherwise

Π_l, Π_{2m}, Π_{3l} = weight factors

Ω= large number

Variables:

W_{lc} = 1 if worker l is assigned for cell c; = 0 otherwise

X_{ic} = 1 if job i is selected for cell c; = 0 otherwise

Y_{lcim} = 1 if job i is selected for cell c and needs machine of type m and worker l; = 0 otherwise

Z_{fclm} = 1 if family f is processed in cell c, on machine of type m and worker l; = 0 otherwise

nn_{mc} = number of type m machines needed for cell c (integer)

v_m^+ / v_m^- = number of additional/excess machines of type m which are needed overall

w_l^+ / w_l^- = number of additional/fewer cells (than 1) to which worker l has to be assigned

Help variables:

α_c = setup factor indicating the ineffectiveness to reduce the need for setups in cell c (varies between 0 and 1)

LB_{fclm}= the minimal (=lower bound) number of setups needed for jobs of family f, in cell c, performed by worker l on machine m

UB_{fclm} = the maximal (=upper bound) number of setups needed for jobs of family f, in cell c, performed by worker l on machine m

---

$$Max\,Z = \prod_{l}\sum_{i}\sum_{c}\sum_{m}(x_{ic}t_{im}) - \sum_{m}\prod_{2m}\upsilon_m^+ - \sum_{l}\prod_{3l}w_l^+ \quad (1)$$

Subject to:

$$\sum_{c} x_{ic} \le 1 \quad \forall i, \quad (2)$$

$$\sum_{c} nn_{mc} \le \theta_m + v_m^+ - v_m^- \quad \forall m, \quad (3)$$

$$n_{mc} \le nn_{mc} \quad \forall m, c, \quad (4)$$

$$\sum_{c} w_{lc} \le 1 + w_l^+ - w_l^- \quad \forall l, \quad (5)$$

$$t_{im}x_{ic} \le \Omega \sum_{l} y_{iclm} \quad \forall i, c, m, \quad (6)$$

$$\sum_{i} y_{iclm} \le \Omega \phi_{lm} w_{lc} \quad \forall c, l, m, \quad (7)$$

$$\sum_{i\varepsilon f}\sum_{l} y_{iclm} \le \Omega z_{fclm} \quad \forall f, c, l, m, \quad (8)$$

$$\sum_{c}\sum_{m}\left[\sum_{i} t_{im}y_{iclm} + \sum_{f}\left(LB_{fclm} + \alpha_c\left(UB_{fclm} - LB_{fclm}\right)\right)S_{fm}\right] \le R \quad \forall, \quad (9)$$

$$\sum_{l}\left[\sum_{i} t_{im}y_{lcim} + \sum_{f}\left(LB_{fclm} + \alpha_c\left(UB_{fclm} - LB_{fclm}\right)\right)\right] \le n_{mc}R \quad \forall c, m \quad (10)$$

$$LB_{fclm} = z_{fclm} \quad \forall f, c, l, m, \quad (11a)$$

$$LB_{fclm} = \sum_{i\varepsilon f} y_{iclm} \quad \forall f, c, l, m, \quad (11b)$$

w_{lc}, x_{ic}, y_{iclm} *and* z_{fclm} are 0/1 variables $\forall f, c, i, l, m,$ (12)

$nn_{mc}, LB_{fclm}, UB_{fclm}, v_m^+, v_m^-, w_l^+$ *and* w_l^- integer $\forall f, l, m, c,$ (13)

α_c, is a real variable $\forall c.$ (14)

The objective of the model is to maximize production hours and minimizing number of excess machines and workers (resources that are grouped to more than one cell). Note that, in the objective function, each term is multiplied by its corresponding weighting function. Constraints (2) assure that each can be assign to only one cell. Constraints (3) concern the relationship between the number of machine of each type on the shop floor and the number of excess machine. Similar to constraints (3), the relation between the number of workers in cell is governed by constraints (4). Constraints (5), (6) and (7) relate various binary decision variables which govern whether particular job, machine and worker are assigned to a particular cell. Constraints (9), (10), (11a) and (11b) ensure that the time required to perform assigned operations of workers as well as machines is not more than the length of planning period. Finally constraints (12), (13) and (14) define the solution domain.

3.2 Genetic Algorithm for Virtual Cell Formation Problem

GA is a part of evolutionary computing, which is rapidly growing in the area of artificial intelligence (AI). To use GA, the solution must be represented as a genome (or chromosome).

The GA then creates a population of solutions and applies genetic operators such as mutation and crossover to evolve the solutions in order to find the best one. The general outline of GA is summarized below:

Step 1. Generate random population with n chromosomes by using symbolic representation scheme

Step 2. Create a new population by iterating loop highlighted in the following steps until the new population is complete

(i) Select two parent chromosomes from a population
according to from step 2.

(ii) With a preset crossover probability, crossover operation will perform on the selected parents and to form new offsprings (children). If no crossover was performed, offsprings are the exact copy of parents. Here for virtual cellular problem, three-point crossover is used which each part (machine, jobs, worker) in chromosome matched one-point crossover is employed for problem

(iii) With a preset mutation probability, mutation will perform on new offspring at each gene. Chosen genes are swapped to perform mutation process.

(iv) Place new offsprings in the new population.

Step 3. Use new generated population for a further run of the algorithm.

Step 4. Deliver the best solution in the current population. If the end condition is satisfied, stop.

Step 5. Go to step 2.

The general of genetic algorithm based virtual cellular manufacturing tool (Figure 1) consists of chromosome, fitness evaluation, roulette wheel selection and genetic operations (crossover and mutation).This program including graphic user interface and graphical diagrams was coded many line in Visual Basic 2005 programming language.

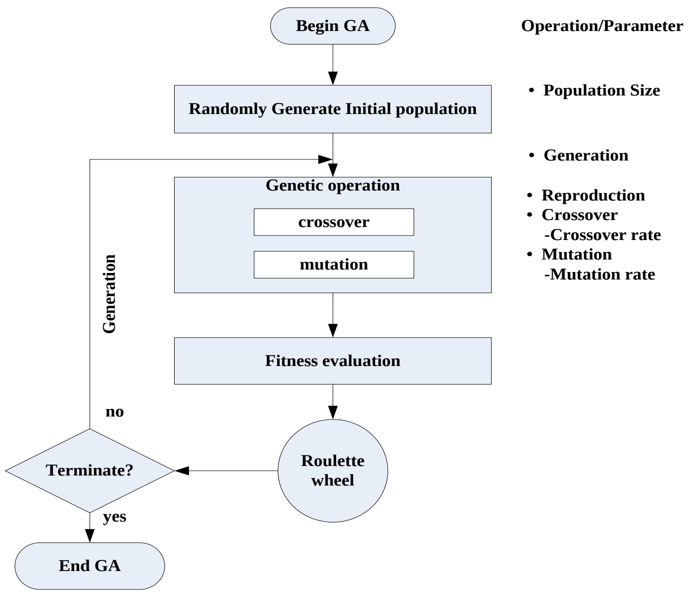

Figure 1. A general framework for GA.

1) Chromosome representation and initialization

In order to encode information of resources, each chromosome is divided into three sections, namely machine, job and worker. For the machine section, the positioning of a gene in the chromosome represents machine number and cell number and each gene contains binary value, which would be 1 if the corresponding machine is to be grouped to the corresponding cell, 0 otherwise. Whereas in the jobs section, the position of a gene represents job number and the value of the gene defines the cell number to which the job will be assigned. For the worker section, the position of a gene describes worker number and cell number and the value of the gene is binary value which determines the assignment of the worker to the cell; 1 if the worker to be in the cell, 0 otherwise.

An example of chromosome structure is presented in Figure 2. In this example, we assume that the number of cells is two and there are four jobs. The number of machines and workers are n. The first section of the chromosome is the machine section. The first gene shows that machine 1 is assigned to cell 1 and the second gene indicates that machine 1 is also to be grouped to cell 2 and so forth. In the job section, job 1, 2 ,3 and 4 are appointed to cell 1, 2, 2 and 1 respectively. The last section shows that worker 1 is neither assigned to cell 1 nor cell 2 and worker 2 is appointed to cell 1. As a result, each individual chromosome is a vector composed of (M*C)+J+(W*C) members as pictorially described in Figure 2 where the notation M denotes the number of machines, J represents the number of jobs, W is the number of workers and C is the number of cell.

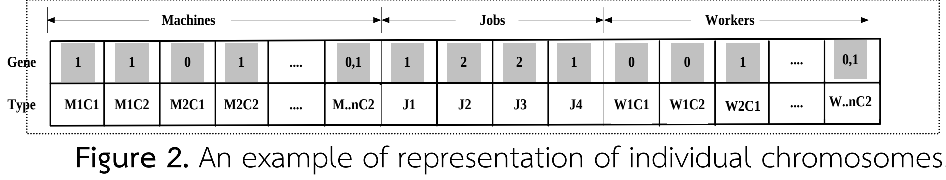

Figure 2. An example of representation of individual chromosomes.

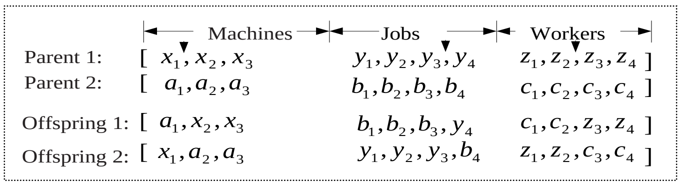
Machines
Jobs
Workers
Parent 1: [x_1, x_2, x_3 y_1, y_2, y_3, y_4 z_1, z_2, z_3, z_4]
Parent 2: [a_1, a_2, a_3 b_1, b_2, b_3, b_4 c_1, c_2, c_3, c_4]
Offspring 1: [a_1, x_2, x_3 b_1, b_2, b_3, y_4 c_1, c_2, z_3, z_4]
Offspring 2: [x_1, a_2, a_3 y_1, y_2, y_3, b_4 z_1, z_2, c_3, c_4]

Figure 3. The representation of crossover in GA for solving the VCM

The next step is to initialize the population which can be achieved by randomly generating chromosome or apply heuristic methodology to create chromosomes. In this work, the initial population is randomly generated.

2) Crossover and mutation operator

In this work, one point crossover is applied for each section of the chromosome as shown in Figure 3. The crossing point is chosen randomly. The crossover is carried out with the probability of crossover rate.

Mutation is aim to prevent premature convergence and explore new solution space. Unlike crossover, the mutation operator alters one or more genes within a single chromosome. The conventional exchange mutation operator is used here and the mutation is performed base on mutation rate.

3) Fitness function

In this stage, objective function from Equation (1) is computed for each chromosome and since our problem is a maximization problem, the objective function value can naturally represent the fitness value of the chromosome which measures the quality of the solution of the chromosome.

4) Roulette wheel selection

After each chromosome has been evaluated for its fitness function, GA selects a number of chromosomes for the next generation. The roulette wheel [6] approach was used for chromosome selection with a random number generator over the range 0-1. The probability of survival and the number of replicates of a chromosome in the next generation are determined by its fitness value divided by the sum of fitness value of all chromosome created. The GA process is then repeated until the termination criteria are satisfied which is when the number of population is reached

4. Results and Discussion

In this work, three problems (shown in Table 1) were established to test the efficiency of the algorithm. The first (small) problem consists of 5 machines, 6 jobs, 3 workers and 3 cells. Whereas in the medium problem, the number of machines, jobs, workers and cells are 10, 40, 6, 4 respectively. The third problem is relatively large. It involves 30 machines, 40 jobs, 12 workers, 5 cells. For all the problem, we assume α_C =0.1 (which indicate high effectiveness in reducing setup time in all cells) and Π_l = 0.8 , Π_{2m} = 0.6, Π_{3l} = 0.4 (weight factors). The length of planning period is set to 1000, 10000, 20000 minutes for small, medium and large problems respectively. The detail of the problems is shown in Table 2.

Table 2. Virtual cell formation test problems

Characteristics of virtual problem									
Number of Problems	Size of Problems	Number of Machines	Number of Cells	Type of Family	Number of Jobs	Number of Workers	Set up times	α_C	R (min.)
1	Small	5	3	3	6	3	29	0.1	1000
2	Medium	10	4	4	20	6	40	0.1	10000
3	Large	30	5	4	40	12	60	0.1	20000

An experimental study was then performed on the medium sized problem to configure GA parameters, namely population size/the number of generations, crossover rate and mutation rate. The results has shown that the best results obtained from the setting of P/G and the probabilities of crossover (%C) and mutation (%M) at 100/25, 0.1 and 0.5, respectively.

Table 3 indicates the performance of GA to solve small, medium and large problems which the best parameter setting form previous experiment were used. This table shows objective function value, means value, standard deviation and execution time in each problem.

Table 3. The results of GA with three problems.

Problem sizes	Objective function value	Means Values	Standard deviation	Time (second)
Small	102	82.6	11.80254	108
Medium	826	625.2	144.7988	161
Large	3943	3270.2	668.0069	745

5. Conclusion

This work proposes a GA approach for solving a complex virtual cell formation problem. Apart from considering machine grouping under VCM environment, the problem also includes labor grouping consideration. The goal programming model is then used to describe the problem. The preliminary study of the algorithm is then performed. Analysis of experiments indicates that the best parameter setting of GA which are P/G and the probabilities of %C and %M at 100/25, 0.1 and 0.5, respectively. Three problems are then tested and can achieve to the solution within acceptable time.

6. Reference

Goldberg, D. E. (1989). **Genetic algorithms in search, optimization and machine learning**. MA: Addison-Wesley.

Ko, K. C., & Egbelu., P. (2003). Virtual cell formation. **International Journal of Operations Research**, 41(11), 2365-2389.

McLean, C. R., Bloom, H. M., & Hopp, T.H. (1982). **The virtual manufacturing cell**. Proceedings of Fourth IFAC/IFIP Conference on Information Control Problems in Manufacturing Technology, 207-215.

Nomden., G. (2011). **Virtual cellular manufacturing**. Relevance and development of heuristics for family based dispatching Groningen. University of Groningen, SOM research school.

Nomden, G., Slomp, J., & Suresh, N.C. (2006). Virtual manufacturing cell: A taxonomy of past research and identification of future research issuse. **International Journal of Flexible Manufacturing Systems**, 17, 71-92.