A GENETIC ALGORITHM FOR CELL-LOAD PROBLEM WITH MULTIPLE-OBJECTIVES

by Agus Ristono

Submission date: 03-Sep-2020 06:46PM (UTC+0700)
Submission ID: 1378835971
File name: 1._Agus_Ristono_UPN_Yogya.doc (118K)
Word count: 2872
Character count: 15451

A GENETIC ALGORITHM FOR CELL-LOAD PROBLEM WITH MULTIPLE-OBJECTIVES

Agus Ristono

e-mail: agus_ristono@yahoo.com
Industrial Engineering Department UPN "Veteran" Yogyakarta
Jl. Babarsari 02 Tambakbayan Yogyakarta Indonesia 55281
Phone: + 62 274 485 363, Fax: + 62 274 486 256

Abstract

This paper presents a genetic algorithm (GA) approach to the cell-load problem with multiple objectives: minimizing make-span and the total load variation within cell. Manufacturing cells are formed based on production data, e.g. part routing sequence, production volume and workload. Also, we will discuss the implication of make-span to deal with it. Special genetic operators are developed and an experiment is performed. Finally, the results obtained with the proposed algorithm on the tested problems are compared with those of the same algorithm but only use the second objective function. The result shows that using multiple objective together is better than only use one objective function.

Keywords: Genetic algorithm, cell-load, total load variation, make-span, multiple objective

1. Introduction

Cell loading means assigning component or parts to cells. The cell loading can be performed to minimize make-span, number of tardy jobs, total tardiness, etc. Make-span is the maximum of completion times on cells. Venugopal and Narendran (1992) proposed the minimization of cell load variation in cellular manufacturing systems. Süer (1996) discussed cell loading in labour-intensive manufacturing cells. He stated that the operator assignment to cells influences production rate that each cell can produce. Süer and Dagli (2005) used three-phase methodology to solve product sequencing in a cell and cell load problem. Their objective was to minimize make-span and number of machines. Cesani and Steudel (2005) presented a research concerning labour assignment strategies, and their impact on the cell performance in cellular manufacturing.

Süer and Tummaluri (2008) developed a three-phase hierarchical methodology to solve cell load problem. The first phase is generating alternative operator levels for each product using operation standard times. The second phase is determining cell loads and cell sizes using standard times. The third phase is assigning operators to operations. A mixed integer mathematical model is used in all phases.

Alhawari (2008) solve the cell load problem in labour-intensive manufacturing cells using two assignment approaches: Max-Min and Max. The major concern is to see how these two approaches impact operators' skill levels, make-span and total processing time among cells. The impact is discussed under chaotic environment where sudden changes in product mix with different operation times are applied and under non-chaotic environment where same product mix is run period after period.

This paper use combination of the objective functions from literatures at above. There are two objective functions: (1) minimization of make-span in the cell and (2) minimization of cell load variation.

2. Mathematical Model

2.1. Minimization of Make-span in the cell

The cell loading issue has been studied in cellular manufacturing environments. A mathematical model is used to assign products to cells and the objective function is to minimize the make-span for each cell. Make-span means the maximum completion time among all cells. Equation (1) shows the objective function, minimizing make-span. Equation (2) shows that the total processing time in each cell should be equal or greater the makespan. Equation (3) ensures that each product is assigned to a cell. Equation (4) shows the sign restriction.

Objective function:

$$Min\ Z = MS \quad \ldots\ldots(1)$$

Subject to:

$$MS - \sum_{i=1}^{n} p_{ij} x_{ij} \geq 0 \quad ;1 \leq j \leq c \quad \ldots\ldots(2)$$

$$\sum_{j=1}^{c} x_{ij} = 1 \quad ;1 \leq i \leq n \quad \ldots\ldots(3)$$

$$x_{ij} \in [0,1] \quad \ldots\ldots(4)$$

Where,

x_{ij} = 1 if product *i* is assigned to cell *j*,
0 otherwise

p_{ij} = Processing time of product *i* in cell *j*

c = Number of cells

n = Number of products

Product sequencing usually comes after cell loading in which products are arranged in such a way, that a selected performance measure is completed. In this paper, average flow time is considered as a measure and it is minimized by using the shortest processing time technique (SPT). SPT denotes that jobs are ordered in the increasing order of processing times.

2.2. Minimization of cell load variation model

This model is based on the model proposed by Venugopal and Narendran (1992). In this model we define m as the number of machines, k as the number of cell, and n as the number of parts. W = $[w_{ij}]$ is an m*n machine component incidence matrix where w_{ij} is the workload on machine *i* induced by part *j*. X = $[x_{ij}]$ is an (m*k) called as cell membership matrix, where

$$x_{ij} = \begin{cases} 1 & \text{if machine is in cell } i \\ 0 & \text{otherwise} \end{cases}$$

M = $[m_{ij}]$ is an (k*n) matrix of average cell load, where

$$m_{jl} = \frac{\sum_{i=1}^{m} x_{il} w_{lj}}{\sum_{i=1}^{m} x_{il}}$$

The total load of cell i induced by part j is given as:

$$\sum_{i=1}^{m} x_{il} w_{lj} \ .$$

The number of machines in cell i is given as:

$$\sum_{i=1}^{m} x_{il}$$

The mathematical programming formulation of the grouping problem is as follows:

$$\text{Minimize } z_1 = \sum_{i=1}^{m}\sum_{i=1}^{k}\sum_{i=1}^{n}(w_{ij} - m_{ij})^2 \quad (5)$$

Subject to:

$$\sum_{l=1}^{k} x_{il} = 1 \quad \forall i \quad (6)$$

$$\sum_{i=1}^{m} x_{il} \geq 1 \quad \forall l \quad (7)$$

The expression $\sum_{i=1}^{m}\sum_{i=1}^{k}\sum_{i=1}^{n}(w_{ij} - m_{ij})^2$ is the objective function. Equation (5) gives the extent of variation of the load on machine i in cell l (induced by all parts) from the mean load of cell i. The objective function z1 adds this quantity for all the machines and cells. Hence this formulation requires a solution for which the total cell load is minimized such that every machine belongs to exactly one cell and no cell is empty. Equation (6) ensures that for particular i, machine i is assigned to one cell only. Equation (7) ensures that no cell is empty.

3. Genetic Algorithm to Solve the Problem

Genetic algorithms were developed by Holland (1975) to mimic the natural selection process in living organisms. The idea is to identify the features that account for successful survival of an individual species over time. In natural selection, species that adapt well to their environment survive. These successful species are described as having a high degree of fitness. Each species can be differentiated by its specific genetic composition embodied in its chromosome, and it is the genetic make-up that accounts for the continual adaptation of a species to its changing environment. The genetic make-up that carries the successful traits for survival and adaptation is acquired and preserved from one generation to the next through three operations: (i) reproduction; (ii) crossover; and (iii) mutation.

Reproduction allows individuals to mate with one another selectively with the intent of passing successful traits along to the next generation. Crossover allows for an exchange of chromosomal materials between two parents to be combined in their offspring, hence synergistically increasing the degree of fitness in these offspring. Mutation allows random variations in the genetic material of the off spring so as to increase its probability of acquiring successful traits that are not passed along from its parents but through background variation. A general structure for a simple genetic algorithm is shown as follows:

Step 1. Let g=0, g is the generation count
Step 2. Initialize a starting population, P(0)
Step 3. Evaluate P(g)
Step 4. Generate P(g+1) using the genetic operators
Step 5. Set P(g) = P(g+*1*)
Until a stopping criterion is met

Genetic algorithm is one of the evolutionary search methods that can provide optimal or near optimal solutions for the combinatorial optimization problems, such as travelling salesman problems, scheduling problems, cell design problems, or process planning problems. Genetic algorithm is difference from conventional optimization and search methods in the following ways (Moon and Kim, 1999):

1. Genetic algorithm works with a coding of solution set, not the solutions themselves.
2. Genetic algorithm searches from a population of solutions, not a single solution point.

3. Genetic algorithm uses probabilistic transition rules, not deterministic rules.
4. Genetic algorithm uses information of fitness function, not derivatives or other auxiliary knowledge.

The key of the algorithm is to make the possible solutions evolve continuously under certain constraints, all the good characteristics must be reserved and developed and finally reach the suboptimal solution. Generally, GA has three steps (Zhao and Wu, 2000).

(1) Coding (initial population is generated randomly).
(2) Reproduction, the population evolves as applying genetic operators (crossover and mutation).
(3) Individual evaluation, give the solution if possible, otherwise go to step (2).

The main issues in developing a genetic algorithm are chromosome representation, initialization of the population, evaluation measure, crossover, mutation, and selection strategy. Also, the genetic parameters such as population size pop_size, number of generation max_ gen, probability of crossover pc, and probability of mutation pm, are determined before execution of genetic algorithm. In the following subsections, these issues and overall procedure are introduced and described for the proposed genetic algorithm.

3.1. Representation and initialization

Genetic algorithm, usually, starts with an initial set of solutions called population and the population at a given time is a generation. Each individual in the population is called a chromosome. The representation of chromosome plays a key role in the genetic algorithm approach. The cell number-based representation scheme is applied in the cell design problem. The chromosome to represent a solution can be represented as follows:

[3 1 1 2 3 1 2]

where the length of the chromosome depends on the number of machines considered in the process of production and each value of gene means the cell number to which that machine has been assigned. We know that, from this, the number of machines and cell numbers considered in this problem were 7 machines and 3 cells, respectively. Machines 2, 3 and 6 are assigned to cell 1; machines 4 and 7 are assigned to cell 2; and machines 1 and 5 are assigned to cell 3.

The second step in genetic algorithm is to initialize the population of chromosomes. The initialization process can be executed with a randomly generated population or a well-adapted population (Gen and Cheng, 2000). The random approach is used widely in many genetic algorithm approaches, but it may yield illegal chromosomes. Thus the checking step should be performed to guarantee that all chromosomes are legal. The random approach is used to initialize the population.

3.2. Evaluation

To improve the solutions, each chromosome is evaluated using some measures of fitness. A fitness value is computed for each chromosome in the population and the objective is to find a chromosome with the maximum fitness value. We use the objective function defined in Section 2 for the fitness function. For the cell design problem, the fitness is simply equal to the value of objective function as follows:

$$\text{eval}(st_i) = Z(st_i) \; ; \; i = 1, 2, 3, \ldots, \text{pop size} \qquad (8)$$

By the way, the genetic algorithm may yield illegal chromosomes because the mathematical model formulated in section 2 has some constraints. To handle the illegal chromosomes, many efficient techniques have been developed. Gen and Cheng (2000) categorized the techniques into four areas: (1) rejecting technique; (2) repairing technique; (3) modifying genetic operators technique; and (4) penalizing technique. Among them, the repairing technique did indeed surpass other techniques in both speed and performance. A new repairing technique for illegal chromosomes in the cell design problem is developed as follows:

Repairing procedure:
Begin
For i =1 to pop_size
Begin
Repeat
check the Uc and Nc in each chromosome;
If (not feasible) **Then**
Begin
select machines having over flow cell number;
replace current cell numbers by new cell numbers generated randomly;
update data;
End
Until (Uc and Nc become feasible)
i=i +1;
End;
End.

3.3. Genetic operators

To create the next generation, new set of chromosomes called offspring is formed by the execution of genetic operators such as crossover and mutation. Crossover acts as the main operator while the mutation acts a background operator. Two-cut-point crossover is used here. For example, the chromosome st_1 and st_2 were selected for crossover as follows, and the positions of the cut-point were randomly selected after second and 5th genes:

st_1 = [3 1 | 1 2 3 | 1 2]
st_2 = [2 3 | 2 1 1 | 2 1]

The offspring by exchanging the left and right parts of their parents would be as follows:

op_1 = [2 3 | 1 2 3 | 2 1]
op_2 = [3 1 | 2 1 1 | 1 2]

The mutation is performed after crossover as random perturbation. It selects a gene randomly and replaces it to a cell number generated within [1, N_c]. If the 5th gene of the chromosome st_1 is selected for mutation and the generated random number was 1, then the chromosome after mutation would be

op_3 = [3 1 1 2 1 1 2]

3.4. Selection

Selection strategy is concerned with the problem of how to select chromosomes from population space. It may create a new population for the next generation based on either all parents and offspring or part of them. A mixed selection strategy based on the roulette wheel and elitist selection is adopted for cell design problem.

3.5. Overall procedure

Let P(t) and C(t) be parents and offspring in current generation t, the overall procedure of the proposed genetic algorithm is described as follows:

Genetic algorithm Procedure:
Begin
t = 0;
initialize P(t);
evaluate P(t);
While (not termination criteria) Do

```
Begin
    apply the genetic operators into the P(t ) to yield C(t );
    evaluate C(t );
    select P(t+ 1) from P(t ) and C(t );
    t = t +1
End;
End.
```

4. Computation analysis

Numerical examples have been provided to verify the approach proposed in previous sections. The example considered has 12 machines and 19 parts. The process plan and production volume for each part is given in Table 1.

Table 1. Process plan and volume (Irani and Huang ,2000).

Part	Sequence	Total batch time (minute)	Part per batch
1	1,4,8,9	96-36-36-72	2
2	1,4,7,4,8,7	36-120-20-120-24-20	3
3	1,2,4,7,8,9	96-48-36-120-36-72	1
4	1,4,7,9	96-36-120-72	3
5	1,6,10,7,9	96-72-200-120-72	2
6	6,10,7,8,9	36-120-60-24-36	1
7	6,4,8,9	72-36-48-48	2
8	3,5,2,6,4,8,9	144-120-48-72-36-48-48	1
9	3,5,6,4,8,9	144-120-72-36-48-48	1
10	4,7,4,8	120-20-120-24	2
11	6	72	3
12	11,7,12	192-150-80	1
13	11,12	192-60	1
14	11,7,10	288-180-360	3
15	1,7,11,10,11,12	15-70-54-45-54-30	1
16	1,7,11,10,11,12	15-70-54-45-54-30	2
17	11,7,12	192-150-80	1
18	6,7,10	108-180-360	3
19	12	60	2

The proposed algorithm written in Pascal and runs on PC with a Pentium 133 MHz. An example were used are presented to compare the result with the same algorithm but only use one objective function (minimization work load variation), or without the first objective function (minimization make span). We applied genetic algorithm to obtain the solution based on both minimization work load variation and minimization make span.

Figure 1 shows the solution achieved by the GA and the performance of the GA without the first objective-function. The solution iteration for the ten replications by GA was 7 on the same result. We can see that the GA performed better than the same without the first objective-function. Figure 1 further shows how fast the GA solutions improved during the search for a typical replication. We can see that the GA search achieved a lower objective function value faster and thus may be a better choice when computational time is limited.

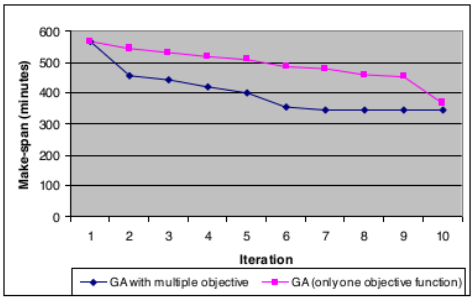

Figure 1. Best solution improvement over solution time

5. Conclusion

An approach is proposed using a genetic algorithm to group parts/component into cells to minimizing cell load variation and considering minimization make-span. Generally, the component/part-grouping problem is a combination problem. This study develops specific genetic operators and makes the problem solving easy. During the cell formation process, the routing sequence of parts, production volume, workload balance, and the constraints of cell number and cell-size are carefully considered. Although taking one objective function does decrease the result during GA searching, an example in section 4 denotes that the result's GA which using multiple objective for solving is better than the same algorithm but only use one objective function. It is completely acceptable and feasible tomedium-scale tasks. Thus, this approach would be suitable for the machine-component/part grouping problem with complicated cell-load requirements.

Reference

Alhawari, O. I. (2008). Operator Assignment Decisions in a Highly Dynamic Cellular Environment. *Thesis Report*, Department of Industrial and Systems Engineering, faculty of the Russ College of Engineering and Technology, Ohio State University, USA.

Cesani , V. I., and Steudel, H. J. (2005). A study of labor assignment flexibility in cellular manufacturing systems. *Computers & Industrial Engineering*, *48*(3), 571-591.

Gen, M. and Cheng, R. (2000). *Genetic algorithms & engineering optimization*. New York: Wiley.

Holland, J. H. (1975). *Adaptation in Natural and Artificial Systems*, Michigan: The University of Michigan Press, USA.

Irani, S. A. dan Huang, H. (2000). Custom design of facility layouts for multi-product facilities using layout modules. *IEEE Transactions on Robotics and Automation*, 16(3) 259-267.

Moon, C. and Kim, J. (1999). Genetic algorithm for maximizing the parts flow within cells in manufacturing cell design. *Computers & Industrial Engineering*, 36, 379-389.

Süer, G. A. (1996). Optimal operator assignment and cell loading in labor-intensive manufacturing cells. *Computers & Industrial Engineering*, *31*(1-2), 155-158.

Süer, G. A., and Dagli, C. (2005). Intra-cell manpower transfers and cell loading in labor-intensive manufacturing cells. *Computers & Industrial Engineering*, *48*(3), 643-655.

Süer, G. A. and Tummaluri, R. (2008). Multi period operator assignment considering skills, learning and forgetting in labour-intensive cells. *International Journal of Production Research*, *2*, 1-25.

Venugopal, V. and Narendran, T. T. (1992). A genetic algorithm approach to the machining grouping problem with multiple objectives. *Computers and Industrial Engineering*, 22 (4), 469-480.

Zhao, C. and Wu, Z. (2000). A genetic algorithm for manufacturing cell formation with multiple routes and multiple objectives. *International Journal of Production Research*, 38(2), 385-395.

A GENETIC ALGORITHM FOR CELL-LOAD PROBLEM WITH MULTIPLE-OBJECTIVES

ORIGINALITY REPORT

18%	16%	13%	%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

MATCH ALL SOURCES (ONLY SELECTED SOURCE PRINTED)

7%

★ Godfrey C. Onwubolu, Victor Songore. "A tabu search approach to cellular manufacturing systems", Production Planning & Control, 2010

Publication

Exclude quotes	On	Exclude matches	< 2%
Exclude bibliography	On		