A Grouping Genetic Algorithm (SIMOGGAs) simultaneously to solve two grouping problems applied to the cell formation problem with alternative process plans.

Emmanuelle VIN*[1], Pascal FRANCQ[2] and Alain DELCHAMBRE[1]

[1]CAD/CAM Department, [2]Department of Information and Communication Sciences
Université libre de Bruxelles (ULB), Brussels, B-1050, Belgium.
* Corresponding author: Tel.: +(32) 2650-2814; Fax: +(32) 2650-4724; E-mail: Emmanuelle.vin@ulb.ac.be

ABSTRACT

This paper addresses the cell formation problem with alternative process plans and machine capacity constraints. Given alternative process plans, machine capacities and quantities of parts to produce, the problem consists in defining the preferential process and the preferential routing for each part (grouping of operations into machines) optimizing the grouping of machines into manufacturing cells. The problem can be decomposed in two distinct sub-problems: the grouping of operations on resources, yielding flows between the machines, and the grouping of these latter into independent cells. The objective is to optimize both groupings considering several assessment criteria like the minimization of the inter-cellular moves. To solve simultaneously both grouping interdependent problems, we propose a modified multiple objective grouping genetic algorithm (SIMOGGA). In our adapted grouping genetic algorithm, each chromosome is composed of two parts, one part for each problem. According to different application rates, the genetic operators are applied on the first, on the second problem, or on both problems. Finally, the population chromosomes simultaneously evolve in both problems.

1. INTRODUCTION

Cellular production systems are an important application of group technology, which consists in decomposing system into sub-systems and in grouping similar things together. Cellular manufacturing systems are based on the creation and the management of several production cells. These cells are composed by complementary machines placed as close as possible and dedicated to a product family. A principal problem for the implementation of these cells is precisely the cell formation problem.

During the last years, the cell formation problem has been addressed in numerous works. Several methods have been presented and can be classified by different ways. We can use the classification proposed by Joines (Joines and al. 1996) to group the resolution methods into different categories as:

- Part Family identification methods
  - Classification and coding
  - Relational data bases
- Part-Machine grouping methods
  - Array based methods
  - Hierarchical clustering (similarity coefficient)
  - Non-hierarchical clustering
  - Graph theory
  - Mathematical programming
  - Artificial intelligence
  - Search heuristic

Joines presents a complete review of production oriented manufacturing cell formation techniques.

Initially, the cell formation problem was really simplified and the methods used to solve the creation of product families were very simple. With time, the problem has evolved with the complexity of the data. New parameters have been taken into account to solve the cell formation problem. New complex methods have been proposed to solve these problems. These methods can be classified with regard to the complexity of the problem, and with regard to the production parameters taken into account such as sequence operation, cost, alternative process plans…

To take into account all these parameters, we consider not one but two grouping problems. The first one consists to allocate a machine for each operation and this machine must be able to achieve this operation (RP, Resource Planning problem). The second grouping problem tries to make independent cells in assigning a cell for each machine (CF, Cell Formation problem).

With both problems, a question arises: how to solve these two grouping and in which order? The resolution can be sequential, semi-simultaneous or completely simultaneous. The sequential resolution finds a solution for the second problem based on the result found for the first problem or conversely. Looking at the used criteria to evaluate both problems, a good solution for the first problem does not imply a good solution for the second

problem. If an optimization is applied on the first problem, a good solution for the general problem may never be found. To avoid the inconvenient of the sequential resolution, the semi-simultaneous resolution is based on several iterations of the sequential resolution. But in this case, the second solution hardly depends on the first solution. Finally, the simultaneous resolution permits to optimize both problems simultaneously with each iteration.

Section 2 presents a state of the art based on the complexity of the parameters taken into account and on the simultaneous resolution. In the section 3, the problem is described with all used parameters and all used constraints. The proposed method is based on the multiple objectives grouping genetic algorithm MOGGA, explained in section 4. This basic method permits to use a multiple objectives decision aided tool integrated in a GGA. On the basis of this tool, a first semi-simultaneous model is presented in the section 5. Section 6 presents our model to solve simultaneously both grouping problems. The different characteristics of the implementation are explained in section 7. A case study is presented in section 8, before the conclusion in section 9.

2. PREVIOUS WORK - STATE OF THE ART

The extension of the part family problem to part-machine grouping problem is based on the different parameters:

- Alternative process plans
- Cost based methods
- Multiple objectives
- Operation sequences
- Part volumes
- Machine capacity, flexibility
- Labor-related factor (training, balanced work loads, …)

Suresh (Suresh and Slomp, 2001) proposes a classification and a review based on these parameters used for a more complex cell formation problem. Nevertheless, its review does not make any distinction between different alternative process plans.

First, Kusiak (Kusiak, 1987) and Choobineh (Choobineh 1988) introduced the alternative routings. In this case, an operation can be achieved by more than one machine ($Pr = \{m_1, (m_2, m_3), m_4\}$). In this example, the process is composed by three operations where the first one is achieved by the machine m_1, the second one by the machine m_2 or m_3 and the last operation is achieved by the machine m_4. Several machines can be identical like machine m_2 and m_3. The simple problem of cell formation becomes the choice of machine to achieve each operation and the grouping of these machines into cells. But the problem is still limited in the search space because there are not many possibilities to do each product.

When the number of identical machines increases, the problem becomes more complex. In this case, the process plan is defined like a sequence of machine type ($Pr = \{tm_1, tm_2, tm_3\}$) where tm_i represents the machine type i. Each type is composed of several machines able to achieve a type of operation. In this case, we can speak about alternative routings because there is one process plan corresponding to several routings (sequence of machines). These processes are used notably by (Askin et al., 1997); (Diallo et al., 1993); (Suresh and Slomp, 2001), (Yin and Yasuda, 2002), (Vin and al., 2003).

Others authors (Kusiak, 1987); (Gupta, 1993); (Logendran et al., 1994); (Caux et al., 2000); (Adenso-Diaz, 2001) use several processes defined like a sequence of machines. ($Pr1 = \{m_1, m_2, m_3\}$, $Pr2 = \{m_3, m_2, m_6,\}$ where the product is defined by two process). So, in this case, we have still alternative routings. The problem is limited to the choice of process to use for each product.

The cell formation becomes more complex when the alternative process plans are used and when each process is defined like a sequence of the machine type ($Pr1 = \{tm_1, tm_2, tm_3\}$, $Pr2 = \{tm_3, tm_2, tm_6,\}$ where the product is defined by two processes). To do each product, we need to choose which process is the best and for each operation, the best machine to manufacture it. The cell formation is resumed by the choice of process, the choice of routing in this process, and the grouping machine into cells. The authors using these real alternative process plans are not frequent (Sofianopoulou, 1999), (Uddin and Shanker, 2001).

With these problems, selection routing and grouping machine into cell, different methods have been proposed based on three strategies as explained in the introduction, sequential, semi-simultaneous and real simultaneous.

Many authors use the first strategy. Gupta (1993) proposed a two-step algorithm to solve this problem. One routing is definitely determined for each part, respecting machine capacity constraints. Next, cell formation is achieved. The drawback of this method is its sequential approach. Routing selection is performed once and the flexibility given by alternative routings is not used to minimize inter-cellular moves.

To evolve the solutions taking into account flexibility in the choice of routing, authors use the second strategy. Nagi et al. (1990) proposed an (semi-simultaneous) iterative method solving the two distinct sub-problems: cell formation, tackled with a heuristic and routing selection, addressed with the Simplex method. The use of the simplex limits the size of the considered problem. Caux et al. (2000) proposed an approach based on simulated annealing and a branch-and-bound algorithm in order to perform routing selection and inter-cellular moves minimization simultaneously. Sofianopoulou (1999) proposed an adapted simulated annealing-based heuristic. A mathematical model assigns randomly part type to a particular process plan (=routing) and finds for this configuration, a machine-to-cell assignment minimizing the intercellular movement. This procedure is iterated until the stopping criteria are satisfied. This method is tested for duplicate machines and/or alternative process plans (sequence of machines). Vin and al. (2003) proposed a semi-simultaneous method based on a genetic algorithm to solve the selection of preferential routing.

Given an operation assignment, another integrated genetic algorithm finds the "best" associated grouping of machines into cell.

The second strategy is often used. However, the methods proposed by these authors are rarely adapted for the real alternative process plans where the search space is larger and the resolution more complex.

In this paper, we solve a cell formation problem with ***real alternative process plans***. We propose a new adapted algorithm (SIMOGGA, Simultaneous resolution by a Multiple Objectives Grouping Genetic Algorithm) based on a multiple objectives grouping genetic algorithm (MO-GGA) ***simultaneously*** to solve both problems:

- The routing selection problem and the allocation of operations on a specific machine, yielding flows between the machines (resource planning problem with several constraints and criteria);
- The grouping of machines into independent cells (cell formation problem).

3. DESCRIPTION OF THE PROBLEM

3.1. NOTATION

3.1.1. Indices

t Machines types index (TM_t=Machine type t). t=1,2,..,nt

m Machines index (M_m=Machine m). m=1,2,..,nm

i Products index (P_i=Product i). i=1,2,..,np

j Process index (Pr_{ij}=Process j of product i). j=1,2,.., npr_i

k Operations index (O_{ijk}=Operation k of process j of product i). k=1,2,.., no_{ij}

c Cells index (C_c=Cell c) c=1,2,.., n_c

3.1.2. Parameters

d_m Availability of machine m.

Q_i Quantity of product i.

T_{ijk} Average operating time of operation O_{ijk}.

T_{ijkm} Operating time if O_{ijk} on machine m.

n_c Maximum number of cells.

nm_c Maximum number of machines in cell c.

The necessary data and the hypotheses are presented hereunder. A machine type has different capabilities in terms of operation types. Each machine m, unique, is characterized by an availability parameter d_m, which is equal to its capacity value times its availability rate. This latter value takes the possible failures into account. Each machine belongs to at least one type and can belong to several types if it is a multi-functional machine.

Each product is defined by a set process (Process = a sequence of npr_i operations $\{o_{ij1}, o_{ij2},\ldots, o_{ij\ npri}\}$). Each operation is not performed on one given machine, but is defined as an operation type that can be accomplished on one machine type (lathe, grinding machine, etc.). So each operation can be performed on all machines belonging to its type. The duration of each operation can be fixed for the considered machine type (average operating time, T_{ijk}), or particularized to a specific machine (operating time, T_{ijkm}).

With this hypothesis, a product has several potential routings available for a specific process. This concept is illustrated in Figure 1. Four operations and thus four machine types define a product. As can be seen, a given machine m_l may belong to several types (for instance m_1 belongs to tm_1 and tm_3. An example of preferential routing is $\{m_2, m_2, m_6, m_5\}$.

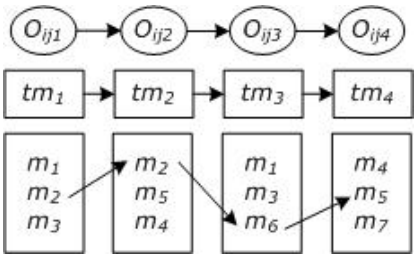

Figure 1. Representation of the process j for the product i.

3.2. FORMULATION

3.2.1. Decision variables

$x_{ij} = 1$ if process j of product i is used (= 0 otherwise).

$y_{ijkm} = 1$ if operation O_{ijk} is achieved on machine m (= 0 otherwise).

$z_{mc} = 1$ if machine m is in cell c (= 0 otherwise).

When the algorithm assigns an operation O_{123} to a specific machine M_5, the variable x_{12} is put at 1 to specify the process j of the product i is used in the solution. This variable implies that all other variables $x_{1j\neq2}$ of the same product (P_1) are put at 0. In this case, all operations belonging to $P_{1j\neq2}$ cannot be used in the grouping solution. To complete this notation, the decision variable y_{1235} is also equal to 1.

The decision variable z_{mc} is used to compute the moves between cells in function of the assignation of machines in each cell.

3.2.2. RP Constraints

$$\sum_{j=1}^{npr_i} x_{ij} = 1 \qquad \forall i \tag{1}$$

$$\sum_{i=1}^{np}\sum_{j=1}^{npr_i}\sum_{k=1}^{no_{ij}} Q_i.T_{ijkm}.y_{ijkm} \le d_m \qquad \forall m \tag{2}$$

$$if\,(y_{ijkm} = 1) => Q_i.T_{ijkm}. > 0 \qquad \forall i,j,k,m \tag{3}$$

The constraint (1) represents the process selection. As explained above, only one process can be chosen by product. The second constraint defines the charge on the machine. This charge cannot exceed the availability of the

machine. Constraint (3) determines that an operation assigned to a specific machine has to have a strictly positive operating time. Indeed, if a machine can not achieve an operation, the operating time T_{ijkm} of the operation O_{ijk} on the machine M_m will be null.

3.2.3. CF Constraints

$$\sum_{c=1}^{nc} z_{mc} = 1 \qquad \forall m \tag{4}$$

$$\sum_{m=1}^{nm} z_{mc} \le n_c \qquad \forall c \tag{5}$$

The constraints (4) and (5) concern the grouping of machine into cells. The first one verifies that all used machines have been grouped. And the second one confirms that the capacity of each cell is not exceeded. The maximum of capacity can be different on each cell.

3.1.6. Cost Function

$$\phi_{mn} = \sum_{i=1}^{np}\left(\sum_{j=i}^{npr_i} x_{ij}\cdot\left(\sum_{k=1}^{no_{ij}-1}\left(y_{ijkm}\cdot y_{ijkn}\right)\cdot\left(Q_i\cdot T_{ij(k+1)n}\right)\right)\right) \tag{6}$$

$$\Phi_{\text{intracell}} = \sum_{c=1}^{nc}\left(\sum_{m=1}^{nm_c}\sum_{n=1}^{nm_c}\left(z_{mc}.z_{nc}\right).\phi_{mn}\right) \tag{7}$$

$$\Phi_{\text{intercell}} = \Phi_{\text{Total}} - \Phi_{\text{intracell}} \tag{8}$$

$$\text{Cost function} : Min\frac{\Phi_{\text{intercell}}}{\Phi_{\text{Total}}} \tag{9}$$

The proposed method is a multicriteria method, but this paper is focused on one criterion: the minimization of inter-cellular moves. Equation (6) represents the move between two machines, *m* and *n*. It is computed on the basis of the sum of operating time to achieve on machine *n* for all products coming from machine *m*. This value can be computed when the first part of the chromosome is completed and the first problem is solved. To compute the equation (7), the second part of the chromosome need to be completed and a valid solution of cell assignment found. The intra-cellular moves into a cell *c* are the sum of moves between all machines assigned to this cell *c*. The total intra-cellular move is the sum of intra-cellular moves for each cell. The total moves can be different in function of choice of process and routing. To compare two solutions and take into account this difference, the criterion to minimize is the relative inter-cellular moves (9). This criterion is the same as the maximization of the total intra-cellular move.

The whole problem is solved with a SIMOGGA whose flowchart is illustrated in Figure 2. This algorithm is an adaptation of the Multiple Objectives Grouping Genetic Algorithm (MOGGA) explained in the next section.

4. MULTIPLE OBJECTIVE GROUPING GENETIC ALGORITHM (MO-GGA)

The genetic algorithms (GAs) are an optimization technique inspired by the process of evolution of living organisms (Holland, 1975). The basic idea is to maintain a population of chromosomes, each chromosome being the encoding (a description or genotype) of a solution (phenotype) of the problem being solved. The worth of each chromosome is measured by its fitness, which is often simply the objective function value of the search space point defined by the (decoded) chromosome. Falkenauer (Falkenauer, 1998) pointed out the weaknesses of standard GAs when applied to grouping problems, and introduced the GGA, which is a GA heavily modified to match the structure of grouping problems. Those are the problems where the aim is to group together members of a set (i.e. find a good partition of the set). The GGA operators (crossover, mutation and inversion) are group-oriented, in order to follow the structure of grouping problems.

Applying GAs to solve multiple-objective problems (MOP) has to deal with the twin issues of searching large and complex solution spaces and dealing with multiple, potentially conflicting objectives. This approach proposes to merge the search and multicriteria decisions. The used multiple objectives grouping genetic algorithm (MO-GGA) is presented by Rekiek (Rekiek and al., 2002). Indeed, in order to come out of the MOP stated by the cost function, the author chose the multicriteria decision-aid (MCDA) method called PROMETHEE II (Brans et Mareschal, 1994). The complete description of this method is out of the scope of this paper. It is however important to know that it computes a net flow φ which is a kind of fitness for each solution. This "fitness" yields a ranking, called the PROMETHEE II complete ranking, between the different solutions in the population. The relative importance of the different objectives are set transparently thanks to weights associated to each criterion.

5. SIMOGGA DESCRPITION

5.1. ORIGINS

Vin (Vin and al., 2003) presents an genetic algorithm in two steps. The used algorithm is based on a semi-simultaneous method. First, a population is initialized by a RP heuristic. This population of chromosome corresponds to the first problem (operation/machine): allocation of operations on specific machines respecting capacity constraints. Next, for each solution, a new genetic algorithm is applied to complete the chromosome with a valid solution for the second problem (machine/cell): machine grouping into cells. When each chromosome owns two solution parts operation/machine and machine/cell, genetic operators are applied to make the population evolve until a good solution is found. For each modified chromosome, the second GA is applied to reconstruct the complete solution and find the best associated grouping machine/cell.

This method was good but its application was not easy. First, the method was limited to cases studies with alternative routings but with only one process (sequence of machine types). Furthermore, the computation time was considerably long. Indeed, in the principal GA, the secondary GA is called to initialize the second part (machine/cell) of all chromosomes. So there are as many GA as the population size. Next, when the genetic operators are applied, the valid solution of the machine/cell must be computed again. A secondary GA is called to reconstruct all chromosomes after these operator applications. Finally, to have a good optimization and find the best secondary solution (machine/cell) associated to first solution (operation/machine), it was necessary to take into account the same criteria in the principal GA and in all secondary GAs.

The following question can be asked: why solve two problems with two different GAs, if the structure and the criteria are similar?

5.2. DESCRIPTION OF SIMOGGA

The SIMOGGA (Simultaneous resolution by a Multiple Objective Grouping Genetic Algorithm) is presented in the figure 2.

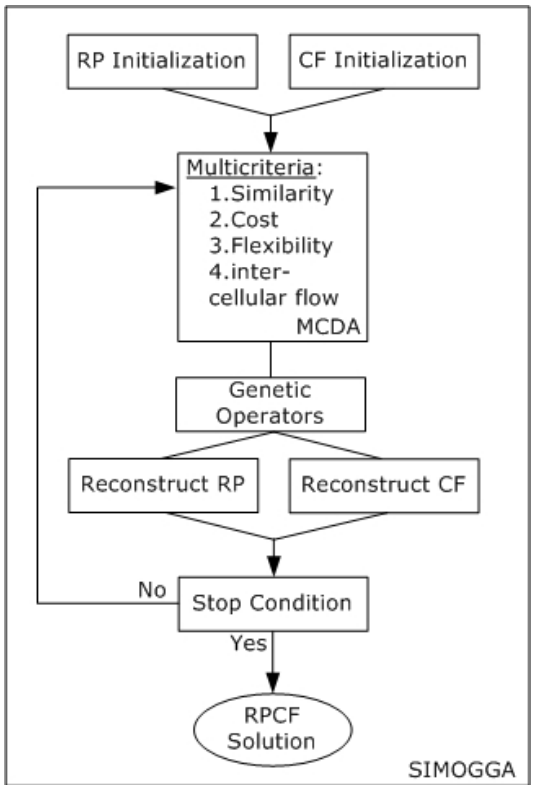

Figure 2. Adapted SIMOGGA applied to the cells formation problem

This algorithm is based on a classical MOGGA. A population of chromosomes is initialized. Each chromosome represents a valid solution to both problems:

- The process selection and the assignment of each operation on a specific machine able to achieve it (**R**esource **P**lanning Problem: operation/machine)
- The grouping of machine into independent cells (**C**ell **F**ormation Problem: machine/cell).

Both problems are interdependent because the groups of the first problem are precisely the objects to group in the second problem. Our objective is to find a "good" solution for both problems (RP, Resource Planning, and CF, Cell Formation).

A RP Heuristic initialized the first part of the chromosome while a random CF Heuristic initialized the second part of each chromosome. These two random heuristics generate only valid solutions respecting all hard constraints defined in section 3.2. After this separated initialization, each chromosome of the population is evaluated on different criteria. On the basis of each evaluation, PROMETHEE II (the multicriteria decision aided tool, MCDA) is applied to sort the population and reorder the chromosomes. The best chromosome is saved. In order to evolve to the best solution, different genetic operators are applied after a specific selection (tournament strategy). These genetic operators are applied on the complete chromosome to make both problems evolve simultaneously. After the application of genetic operators, both parts of each chromosome are reconstructed. And a new generation is begun. The algorithm stops when the stopping condition is reached.

6. IMPLEMENTATION OF THE SIMOGGA

6.1. CODING OF THE CHROMOSOMES

The coding of chromosome is similar to the Grouping Genetic Algorithm (GGA). For the GGA, the standard chromosome of GA is augmented with a group part, encoding the groups on a one gene for one group basis. For example, a chromosome can be encoded as follows:

ADBCEB:ADBCE

with the group part written after the semicolon. Numbering the objects from 0 to 5, the object part of the chromosome can be explicitly written

012345

ADBCEB: ...,

meaning the object 0 is in the group labelled (named) A, 1 in the group D, 2 and 5 in B, 3 in C, 4 in E and 5 in B. The group part of the chromosome represents only the groups. Thus

... :ADBCE

expresses the fact that there are five groups in the solution. This chromosome represents the following solution:

A={0}, B={2,5}, C={3}, D={1} and E={4}.

In fact, the chromosome could also be written in a less visual way as

{0}{2,5}{3}{1}{4}.

In our case, this encoding is doubled. More concretely, let us consider the following chromosome:

ADBCEB:ADBCE: FFGFG:FG

The chromosome encodes the solution for a double grouping problem where the solution of the first problem, composed by 6 objects and 5 groups, can be written as

A={0}, B={2,5}, C={3}, D={1} and E={4}.

The solution of the second problem, composed of the 5 objects (equivalent to the groups for the first problem) and 2 groups, is:

F={0,1,3}, G={2,4}.

The visual encoding can be written as follows:

{0}{2,5}{3}{1}{4}:{0,1,3}{2,4}.

6.2. *INITIALIZATION*

The objective in both problems is to explore the space without limiting the search. It means that we need to test the solution with all groups and with the not used groups. For example, the best solution can be the one with two non used machines producing the minimal inter-cellular moves.

In our problem, the heuristic can not be a first-fit heuristic like in the bin packing problem (Falkenauer, 1998), because this heuristic implies a minimum number of used groups. The equal pile heuristic (Falkenauer, 1998) tries to equilibrate the charge with a constant number of used machines.

Our heuristic is based on a first fit with a first random part. The different objects are treated in random order. The heuristic is decomposed in several steps:

Step 1. **Verify** if the object can be used in order to respect the hard constraints with the actual solution in construction (One used process by product (RP), machine not empty to be grouped (CF)).

Step 2. **Create a new group** with a variable probability equal to

$$p = \frac{MaxGroups - NbGroups}{MaxGroups} \tag{10}$$

where: *MaxGroups* = maximum of allowed groups for the problem, and *NbGroups* = actual number of used groups.

Step 3. **Find the first group** able to accept the object in order to respect the hard constraints about capacity or compatibility (Capacity not exceeded (RP and CF), machine able to achieve the operation (RP)). If a group is found, the object is inserted. Otherwise, a new group is created to accept the object.

The algorithm stops when all objects have been tested and inserted if they had to be used.

Thanks to the used probability *p*, the probability to create a new group before the assignment of the object in a group decreases with the number of created groups. The number of used groups will not necessary be minimal. Randomly, solutions will not contain the same number of groups except if the capacity requires it.

6.3. *GENETIC OPERATORS*

The important point is that the genetic operators will work with the group part of the chromosomes, the standard object part of the chromosomes serving to identify which objects actually form which group. Note in particular that this implies that the operators will have to handle chromosomes of variable length with genes representing the groups.

6.3.1. ***Selection***

The tournament strategy is chosen to select the chromosome for the application of operators. The idea is to create an ordered list of the individuals with the best solution always at the top, and the others ordered according to a specific method described below. The upper part of the list will further be used when “good” chromosomes are needed, while the lower part for “bad” chromosomes. An initial set of all the individual identifiers is established. Two identifiers of this set are chosen randomly, and their fitness values are compared. The best of the two - the one corresponding to the individual with the best fitness value - is reinserted into the set, while the other is pushed into the list. This method is repeated until all the identifiers in the set are in the list; this process leads to the construction of an ordered list of individuals, with the best one at the top of the list. The operator can then presume that the chromosomes ranked in the top half will be used as parents for the crossovers and the resulting children will replace the chromosomes in the bottom half.

6.3.1. ***Crossover***

The crossover will fit the following pattern:

1. A crossing site is selected randomly in each parent. These sites can be situated integrally in the first or second part or overlapping on both parts.
2. The groups selected by the crossing site of one parent are inserted at the crossing site of the second parent. At this stage, some objects may appear in more than one group.
3. The objects cannot appear twice in one solution. The new injected objects have the priority. So, the existing groups containing objects that are already in the inserted groups are eliminated. After these eliminations, some objects are no longer in a group. If some groups are empty, they are also removed from the solution.
4. The validity of the solution is verified according to the hard constraints relative to the cell formation problem. The used process can be differed in two parents. Two process of a same product cannot coexist in the solution. Moreover, a specific machine cannot appear twice. Compatibility is tested between inserted groups and existing groups. If the existing groups contain operations belonging to another process or the groups corresponding to an inserted machine, these groups are also eliminated.

5. The objects left aside are reinserted into the solution. It is the reconstruction phases.

6.3.3. ***Mutation***

The role of a mutation operator is to insert new characteristics into a population to enhance the search space of the Genetic Algorithm. In our case, the mutation is based on two principal aspects:

1. The first idea is to randomly choose a few groups and to remove them from the solution.
2. The second idea is specialized for the cell formation problem. It is to force the use of another process for a few products. All operations of selected process are eliminated

The objects attached to these groups are then reinserted into the solution. The method used for the insertion of the objects is generally the same as the one used for the crossover operator.

6.3.4. ***Inversion***

The role of the inversion operator is to propose the same solution to the Grouping Genetic Algorithm, but differently. As pointed out in 6.1, a single solution may have different presentations, and because crossovers work through crossing sites, the way in which a solution is presented influences the crossover operator's results. The first group appearing in the group element of a chromosome has likely less probability to be chosen than the other groups. It is therefore important to include this operator in the Grouping Genetic Algorithms.

In function of the encoding of the SIMOGGA including two parts of the chromosome, it is more important to correctly apply this inversion. Indeed, randomly, we can have a crossing site overlapping on both parts. In these cases, the concerned groups are always the last of the first part and the first groups of the second part of the chromosome. Thanks to the inversion operator, all groups can be implied in a crossover applied simultaneously on both parts.

6.4. RECONSTRUCTION

After operators of crossover and mutation, it is necessary to reinsert all the objects not assigned. Therefore, the part RP is reconstructed with the same random heuristic used for the initialization respecting all the hard constraints. The part CF is reconstructed with a heuristic based on the computed move (flow) between the machines. The different objects are treated in random order. The flows between the object (machine) to insert and each group (cell) are computed. The object is inserted in the group with the maximum flow. If it is not possible because of the capacity constraint, the object is inserted in the group with the next maximum flow. If it is not yet possible, a new group is created.

6.5. COST FUNCTION

As explained in section 4, the algorithm uses the multicriteria decision aided tool PROMETHEE II to order all the chromosomes. Once the chromosomes are ordered, this ranking is transformed into a fitness function as presented by Francq (2003). The net flow used to compute the complete ranking is based on the population of analyzed chromosomes. A solution will no have the same net flow in two different populations. For this reason, Francq proposed the specific fitness function to use after the ranking given by PROMETHEE II. Let us suppose that at generation g the chromosomes were ranked as s_1, $\ldots s_{|A|+1}$ where s_1 was the best ranked chromosome and $s_{|A|+1}$ the worst ranked one. Let us also assign to each chromosome a cost function, $C_g(C_c)$ defined as follows:

$$C_g\left(C_c\right)=\begin{cases} g+1.1 & C_c = s_1 \text{ and } s_1 \neq C_{Best} \\ C_{Best,g-1} & C_c = s_1 \text{ and } s_1 = C_{Best} \\ 1 & C_c = s_2 \text{ and } s_1 \neq C_{Best} \\ \frac{|A|+1-r}{|A|} & C_c = s_r \neq C_{Best} \text{ and } c > 2 \end{cases} . \quad (11)$$

The value of the cost function of the highest ranked chromosome is set to *g+1.1* if it is not the best chromosome ever computed, otherwise the cost function remains identical. The value of the cost function for the second highest ranked chromosome is set to 1. The values of the cost function for the other chromosomes are proportional to the corresponding ranking.

The complete ranking can be computed on the basis of several criteria as flexibility, cost, load balancing … as presented in (Vin and al., 2005). As explained in section 3.2, the algorithm is presented with the flow criterion: minimization of the inter-cellular moves.

6.6. STOPPING CONDITION

A stop condition must determine when a GA stops. When the problem to be optimized is well defined, a particular value of the cost function can be used as exit condition. But such a value is unknown for our grouping problem. Another exit criterion must therefore be defined. The trivial idea used is to run the SIMOGGA a fixed number of generations. Another possible idea could be to let the SIMOGGA running until the age[1] of the best solution ever computed is greater than a given threshold.

The number of generations influences the quality of the solution. In fact, the number of generations has a double effect on a Genetic Algorithm (GA):

- When the number of generations increases, the quality of the GA should also increase.
- When the number of generations decreases, the computational time needed also decreases.

The maximum number of generations must always be chosen by balancing these two effects one against the other.

[1] The age of a solution is the number of generations during which the solution is not modified.

7. APPLICATION, CASE STUDY

The algorithm has been tested with different case studies found in the literature. In this article, we will present the four case studies used by Sofianopoulou. The advantage of these four cases is that they represent a set of the different processes presented in the section 2. The implementation of the SIMOGGA algorithm for these problems is coded in C++ and run on a Bi-Xeon 3.60Ghz Hyper Threading with 1 Go RAM.

For each problem, the solution is presented in a table where the cell composition is determined by the machines and the products assigned to each cell. When the product is written in parentheses in two cells, there are so many moves in each cell. In this case, the product can be assigned independently to each cell. Furthermore, the selected process plan is defined in parentheses for each product with alternative process plans.

7.1. PROBLEM 1

The first problem (P1) is an adaptation from Kusiak (1990) to take into account the processing sequence of each part. We consider 5 products and 4 machines. The particularity of this case is the use of alternative routings (alternative machines sequences). Each type of machine is composed by only one machine. The different routings have not necessary the same number of operation as shown by the fifth product.

The algorithm solves this problem for a maximum cell size nm_c=2.

The solution is presented in table 1. It is the same solution as the one presented by Sofianopoulou and Kusiak with a number of inter-cellular moves equal to 0.

Table 1: Solution of Problem 1

Cell	Machines	Products
1	1, 3	2, 4, 5
2	2, 4	1, 3
Selected process for each product		
1(2), 2(2), 3(2), 4(2), 5(2)		

7.2. PROBLEM 2

The second problem has the same particularity as the first problem except the size. The algorithm tries to group 20 products and 12 machines into 3 cells. The maximum cell size is equal to 5. The solution to this problem is presented in table 2. The algorithm was run for 500 generations and the corresponding runtime was about 3 seconds. The solution contains the same number of inter-cellular moves (29) as the Sofianopoulou's solution. However, the selected process plans for each product is not the same and the machines are not allocated into the same cell. But the principal difference with Sofianopoulou's solution is the type of moves. In our algorithm, the cell formation heuristic is adapted to produce preferentially unidirectional moves. To simplify the moves between cells, it is better to visit cell 1, cell 2 and cell 3 than to visit cell 1, cell 2 and to come back to cell 1. The proposed solution has 13 go-returns and 16 simple moves between cells. Sofianopoulou's solution has the opposite, 16 go-returns, and 13 simple moves.

Table 2: Solution of Problem 2

Cell	Machines	Products
1	1, 4	3, 15
2	2, 6, 7, 9, 10	2, 4, 5, 8, 10, 12, 14, 16, 19,20 (6, 17)
3	3, 5, 8, 11, 12	1, 7, 9, 11, 18 (13)
Selected process for each product		
2(2), 5(1), 12(2), 14(2), 17(2)		

7.3. PROBLEM 3

The problem 3 is composed by 20 products and 14 machines (12 machine types). Two machines are duplicated. Each product is characterized by an unique process sequence. The utilization of duplicate machines implies one or several routings (machine sequence) for each product. As for the problem 2, the cell size is limited to 5. The algorithm was run for 500 generations in 6 seconds. The solution is presented in table 3. The same observations as for the problem 2 can be done. The selected process plans are not equivalent but in this case, the types of moves are equivalents.

Table 3: Solution of Problem 3

Cell	Machines	Products
1	1, 4, 7, 10, 13	3, 4, 8, 10, 14, 15, 20, (9, 13, 19)
2	2, 3, 5, 6, 11	1, 2, 5, 11, 16, 17 (6, 9, 13)
3	8, 9, 12, 14	7, 12, 18, (19)
Selected process for each product		
1(1), 2(1), 3(2), 6(3), 8(2), 9(2), 10(2), 11(3), 12(2), 13(2), 15(2), 16(1), 17(1), 18(2), 19(2), 20(2)		

7.4. PROBLEM 4

The problem 4 is an application of the alternative process plans. Each product is defined by several processes (machine type sequence) and each machine type can contain one or two machines. So the problem will consist to define the preferential process and the preferential routing for each product. This problem is composed by 30 products and 18 machines regrouped into 16 machine types. For this problem, the cell size is limited to 7 machines by cell. The solution presented in table 4 is found with 500 generations in 12 seconds. The best total number of inter-cellular moves found by the algorithm is 32, less than Sofianopoulou's solution (34).

Sofianopoulou needed 100 seconds to solve these both last problems. The presented SIMOGGA is fast and efficient with all types of data in term of alternative process plans and alternative routings.

Table 4 Solution of Problem 4

Cell	Machines	Products
1	1, 2, 7, 11, 13, 14, 15	2, 4, 9, 11, 12, 15, 25, 30 (1, 17)
2	5, 8, 18	8, 24, 29 (5)
3	3, 4, 6, 9, 10, 12, 16	3, 6, 7, 10, 13, 14, 16, 18-23, 26-28 (1, 17)
Selected process for each product		
1(1), 2(1), 3(3), 4(1), 6(3), 8(3), 10(1), 11(1), 12(3), 13(1), 16(1), 21(1), 23(1), 24(2), 26(1), 29(6)		

8. SUMMARY

In the present paper, an adapted multi objective grouping genetic algorithm has been developed to solve a double grouping problem. The algorithm SIMOGGA can solve simultaneously two grouping problems. It is particularized to the cell formation problem for manufacturing systems. We took into account the three most important production parameters in cell design:

- The process sequence allows to define the flows between machines;
- The production volume allows to compute the real material moves between machines and between cells;
- The use of real alternative process plans allows optimizing the cell formation.

In this paper, the algorithm is applied with one flow criterion to four case studied used by Sofianopoulou. These four cases are a good variety of the different processes utilization. The algorithm is as efficient with low flexibility cases as high flexibility cases.

REFERENCES

Adenso-Diaz, B, Lozano, S., Racero, J., Guerrero, F., 2001. Machine cell formation in generalized group technology. Computers and industrial engineering, Vol. 41, No. 2, pp. 227-240.

Askin, R., Selim, H., Vakharia, A., 1997. A methodology for designing flexible cellular manufacturing systems. IIE transactions, Vol. 29, No. 7, pp. 599-610.

Brans, J.-P., Mareschal, B., 1994. The PROMCALC & GAIA decision support system for multicriteria decision aid. Decision support systems, Vol. 12, pp. 297-310.

Caux, C., 2000. Cell formation with alternative process plans and machine capacity constraints: A new combined approach. International Journal of Economics, Vol. 64, pp. 279–284.

Choobineh, F., 1998. A Framework for the Design of Cellular Manufacturing Systems. International Journal of Production Research, Vol. 26, No. 7, pp. 1161-1172.

Diallo, M., Pierreval, H., Quilliot, A., 1993, Manufacturing cells design with flexible routing capability in presence of unreliable machines. International Journal of Production Research. Vol. 74, No. 1, pp.175-182.

Falkenauer, E., 1998. Genetic algorithm and grouping problem. Wiley & Sons.

Francq, P., 2003. Structured and Collaborative Search: An integrated approach to share documents among users. Université Libre de Bruxelles (ULB), PhD Thesis.

Gupta, T., 1993. Design of manufacturing cells for flexible environment considering alternative routing. International Journal of Production Research, Vol. 31, No. 6, pp. 1259-1273.

Holland, J. H., 1975. Adaptation in Natural and Artificial Systems. Ann Arbor: University of Michigan Press.

Joines, J. Culbreth, C., King, R., 1996. A Comprehensive Review of Production Oriented Cell Formation Techniques. International Journal of Factory Automation and Information Management, Vol. 3, No. 3-4, pp. 225-265.

Kusiak, A., 1987. The Generalised Group Technology Concept. International Journal of Production Research, Vol. 25, No. 3, pp. 561-569.

Lee, R., Ramakrishna P., Srikandarajah, C., 1994. Tabu search-based heuristics for cellular manufaturing systems in the presence of alternative process plans. International Journal of Production Research, Vol. 32, No. 2, pp. 273-297.

Logendran, R., Ramakrishna P., Srikandarajah, C., 1994. Tabu search-based heuristics for cellular manufaturing systems in the presence of alternative process plans. International Journal of Production Research, Vol. 32, No. 2, pp. 273-297.

Nagi, R., Harhalakis, G., Proth, J.-M., 1990. Multiple routings and capacity considerations in group technology applications. International Journal of Production Research, Vol. 28, No. 12, pp. 2243-2257.

Rekiek, B., De Lit, P., Delchambre, A., 2002. Hybrid assembly line design and user's preference. International Journal of Production Research, Vol. 40, No. 5, pp. 1095-1111.

Sofianopoulou, S., 1999. Manufacturing cell design with alterantive process plans and/or replicate machines. International Journal of Production research, Vol. 37, No. 3, pp. 707-720.

Suresh, N.C., Slomp, J., 2001. A multi-objective procedure for labour assignments and grouping in capacitated cell formation problems. International Journal of Production research, Vol. 39, No. 18, pp. 4103-4131.

Uddin, M.K., Shanker, K., 2002. Grouping of parts and machines in presence of alternative process routes by genetic algorithm. International journal of production economics, Vol. 76, No. 3, pp. 219-228

Vin, E., De Lit, P., Delchambre, A., 2003. An integrated approach to solve the cell formation problem with alternative routings. Une approche intégrée pour résoudre le problème de formation de cellules de production avec des routage alternatifs. Proceedings of 4e Conférence Francophone de MOdélisation et SIMulation, Vol. 1, pp. 43-50, Toulouse, France.

Vin, E., De Lit, P., Delchambre, A., 2005. A Multiple Objective Grouping Genetic Algorithm for the Cell Formation Problem with Alternative Routings. Journal of Intelligent Manufacturing, Vol 16, No. 2, pp. 189-206.

Yin, Y. and Yasuda, K., 2005. Manufacturing cells' design in consideration of various production factors. International journal of production research, Vol. 40, No. 4, pp. 885-906.