Chapter 10
Manufacturing Cell Formation Problem Using Hybrid Cuckoo Search Algorithm

Bouchra Karoum, Bouazza Elbenani, Noussaima El Khattabi, and Abdelhakim A. El Imrani

Abstract Cellular manufacturing, as one of the most important applications of Group Technology, has gained popularity in both academic research and industrial applications. The cell formation problem is considered the first and the foremost issue faced in the designing of cellular manufacturing systems that attempts to minimize the inter-cell movement of the products while maximize the machines utilization. This paper presents an adapted optimization algorithm entitled the cuckoo search algorithm for solving this kind of problems. The proposed method is tested on different benchmark problems; the obtained results are then compared to others available in the literature. The comparison result reveals that on 31 out of 35 problems (88.57%) the results of the introduced method are among the best results.

Keywords Cell formation problem • Lévy flight • Cuckoo search • Metaheuristic • Cellular manufacturing

10.1 Introduction

Cellular manufacturing (CM) is one of the most important applications of the Group Technology that aims to convert a production system into several mutually separable production cells. Where dissimilar machines are aggregated into machine groups (also known as manufacturing cells) and similar parts into part families so that one or more part families can be processed within a single machine group. The main objective is minimizing the intercellular and intracellular movements. Many significant

B. Karoum (✉) • B. Elbenani
Research Computer Science Laboratory (LRI), Faculty of Science, Mohammed V University of Rabat, B.P. 1014 Rabat, Morocco
e-mail: bouchra.karoum@gmail.com; elbenani@fsr.ac.ma

N. El Khattabi • A.A. El Imrani
Conception and Systems Laboratory (LCS), Faculty of Science, Mohammed V University of Rabat, B.P. 1014 Rabat, Morocco
e-mail: khattabi@fsr.ac.ma; elimrani@fsr.ac.ma

L. Amodeo et al. (eds.), *Recent Developments in Metaheuristics*,
Operations Research/Computer Science Interfaces Series 62,
DOI 10.1007/978-3-319-58253-5_10

benefits have been reported for CM, including reducing setup and throughput times, minimizing the material handling costs, improving the quality and the production control, etc.

The cell formation problem (CFP), which is a non-polynomial hard optimization problem [3], is the first issue in the designing of cellular manufacturing systems. It involves the identification of machine cells and part families with the objective of minimizing the intercellular and intracellular part movements.

In the last decades, many solution methods have been proposed to solve this problem. Chandrasekharan and Rajagopalan [1] developed an algorithm entitled ZODIAC (zero-one data: ideal seed algorithm for clustering) for solving the CFP and used a new concept called relative efficiency as a stopping rule for the iterations. Srinivasan and Narendran [14] proposed an efficient non-hierarchical clustering algorithm, based on initial seeds obtained from the assignment method, named GRAFICS for the rearrangement of parts and machines. Elbenani and Ferland [4] suggested an exact method for solving the manufacturing cell formation problem. A good survey paper which review briefly the different methodologies used to solve the problem until 2008 is presented in [10].

Recently, several authors have adopted the use of the metaheuristic algorithms for the cell formation problem, because of its efficiency in solving combinatorial optimization problems. Goncalves and Resende [6] presented a new algorithm that combines a local search heuristic with a genetic algorithm for obtaining machine cells and part families. James et al. [8] employed a standard grouping genetic algorithm with a local search mechanism to form machine-part cells. Diaz et al. [2] suggested a greedy randomized adaptive search procedure (GRASP) heuristic to obtain lower bounds for the optimal solution of the CFP. Sayadi et al. [11] recommended a new solution based on a discrete firefly algorithm for solving the problem. Solimanpur and Elmi [13] presented a nested application of tabu search approach to solve the problem heuristically. Husseinzadeh Kashan et al. [7] introduced a new solution approach based on the particle swarm optimization (PSO) algorithm for solving the CFP. Ying et al. [16] developed a simulated annealing based metaheuristic with variable neighbourhood to form part-machine cells. Elbenani et al. [5] hybrided a genetic algorithm with a local search procedure that applies sequentially an intensification strategy to improve locally a current solution and a diversification strategy destroying more extensively a current solution to recover a new one. A grouping version of league championship algorithm has been proposed by Seyedhosseini et al. [12] for solving the CFP, and so on.

In this paper, a recently developed cuckoo search (CS) algorithm is adopted for solving the CFP with the aim of maximizing the grouping efficacy. The CS algorithm is combined with a local search method in order to intensify the search towards promising regions. The experimental results show that the proposed algorithm generates good results in reasonable computational time.

The remainder of this article is organized as follows: Sect. 10.2 describes the CFP; Sect. 10.3 gives an overview of the standard cuckoo search algorithm and introduces the improvement carried out on the algorithm to solve the CFP. The results of numerical experiments on a set of benchmark problems are reported in Sect. 10.4, and concluding remarks are given in Sect. 10.5.

10.2 Problem Formulation

The input parameter of the CF problem is a machine-part incidence matrix A where each row represents a machine and each column represents a part. The objective is to determine a rearrangement so that machine utilization within a cell is maximized and the inter-cellular movement is minimized. Figure 10.1 shows a 5×5 machine part incidence matrix to a problem with five machines and five parts. The second matrix indicates a solution of the problem by partition into 3 different cells illustrated in the gray blocks. The 1 outside the diagonal blocks are called exceptional elements, while the 0 inside the diagonal blocks are called voids.

Different measures have appeared in the literature to compare the efficiency of the methods. The most used is the grouping efficacy (*Eff*) [9] where the closer the grouping efficacy is to 1, the better will be the solution obtained (see Eq. (10.1)).

$$Eff = \frac{(a - a_1^{Out})}{(a + a_0^{In})} \tag{10.1}$$

Where:
a: Total number of entries equal to 1 in the matrix A;
a_1^{Out}: Number of exceptional elements;
a_0^{In} : Number of voids.

To formulate this problem, a mathematical model similar to the one used in [4] is adopted.
Parameters:
i, j, k: Index of machines, parts and cells, respectively.
M, P, C: Number of machines, parts and cells, respectively.
A: Machine-part incidence matrix $A = \lfloor a_{ij} \rfloor$.
a: Total number of entries equal to 1 in the matrix A.
$a_{ij} = 1$ if machine i process part j; 0 otherwise.
$x_{ik} = 1$ if machine i belongs to cell k; 0 otherwise.
$y_{jk} = 1$ if part j belongs to cell k; 0 otherwise.

The objective function of the CFP can be presented as follow:

$$\min ComEff(x, y) = \frac{a + \sum_{k=1}^{C} \sum_{i=1}^{M} \sum_{j=1}^{P} (1 - 2a_{ij}) x_{ik} y_{jk}}{a + \sum_{k=1}^{C} \sum_{i=1}^{M} \sum_{j=1}^{P} (1 - a_{ij}) x_{ik} y_{jk}} \tag{10.2}$$

Parts		1	2	3	4	5
Machines	1	0	1	0	0	1
	2	1	0	1	0	0
	3	1	0	0	1	0
	4	0	1	0	0	0
	5	1	0	0	1	0

Initial incidence matrix

Parts		2	5	1	4	3
Machines	1	1	1	0	0	0
	4	1	0	0	0	0
	3	0	0	1	1	0
	5	0	0	1	1	0
	2	0	0	1	0	1

Matrix solution

Fig. 10.1 Incidence matrix solution

Subject to:

$$\sum_{k=1}^{C} x_{ik} = 1 \quad i = 1, ..., M \tag{10.3}$$

$$\sum_{k=1}^{C} y_{jk} = 1 \quad j = 1, ..., P \tag{10.4}$$

$$\sum_{i=1}^{M} x_{ik} \geq 1 \quad k = 1, ..., C \tag{10.5}$$

$$\sum_{j=1}^{P} y_{jk} \geq 1 \quad k = 1, ..., C \tag{10.6}$$

$$x_{ik} = 0 \; or \; 1 \quad i = 1, ..., M; k = 1, ..., C \tag{10.7}$$

$$y_{jk} = 0 \; or \; 1 \quad j = 1, ..., P; k = 1, ..., C \tag{10.8}$$

Constraint (10.3) ensures that each machine is assigned to exactly one cell. Constraint (10.4) guarantees that each part must be assigned to only one cell. Inequalities (10.5) and (10.6) ensure that each cell includes at least one machine and one part. Finally, constraints (10.7) and (10.8) denote that the decision variables are binary.

10.3 Improved Cuckoo Search Algorithm

10.3.1 Basic Cuckoo Search

Cuckoo search algorithm is a new nature-inspired metaheuristic algorithm developed by Yang and Deb in 2009 [15]. It was inspired by the special lifestyle and the aggressive brood parasitic behavior of some species of a bird family called cuckoo.

This algorithm is initially designed for solving multimodal functions and can be summarized around the following three ideal rules [15]:

- Each cuckoo lays one egg at a time, and dumps it in a randomly chosen nest.
- The best nests with high quality of eggs (solutions) will carry over to the next generations.
- The number of host nests is fixed, and a host can discover an alien egg with a probability $p_a \in [0,1]$.

The last assumption can be approximated by a fraction p_a of the n nests are replaced by new ones.

In the standard cuckoo search algorithm, the behavior of cuckoo is associated with Lévy flight to search a new nest. Lévy flights, named by the French mathematician Paul Lévy, represent a model of random walks characterized by their step lengths that obey a power-law distribution.

When generating a new solution $x_l^{(t+1)}$ for a cuckoo l, in iteration $t+1$, a Lévy flight is performed using Eq. (10.9).

$$x_l^{(t+1)} = x_l^{(t)} + \alpha \oplus l\acute{e}vy(\beta) \tag{10.9}$$

Where $\alpha > 0$ is the step size parameter that should be chosen according to the scales of the problem. In most cases, $\alpha = 1$. The product $\oplus$ means entry-wise multiplications. The Lévy flight is a random walk where the step length follows a Lévy distribution that has an infinite variance with an infinite mean. It allows the exploration of the search space more efficiently as its step length is much longer in the long run. A simple scheme discussed in detail by Yang [15] can be approximated ($\sim$) by Eq. (10.10):

$$l\acute{e}vy(\beta) \sim 0.01 \frac{\mu}{|\nu|^{1/\beta}} (x_b^{(t)} - x_l^{(t)}),\ 1 < \beta \leq 3 \tag{10.10}$$

Where μ and ν are drawn from normal distribution:

$$\mu \sim N(0, \sigma_\mu^2),\ \nu \sim N(0, \sigma_\nu^2) \tag{10.11}$$

With

$$\sigma_\mu = \left(\frac{\Gamma(1+\beta)\sin(\frac{\Pi\beta}{2})}{\Gamma(\frac{1+\beta}{2})\beta 2^{\frac{(\beta-1)}{2}}} \right)^{\frac{1}{\beta}}, \sigma_\nu = 1 \tag{10.12}$$

Where Γ is the standard Gamma function.

10.3.2 The Proposed Cuckoo Search Algorithm

This section introduces the modification made on the standard CS algorithm in order to solve the cell formation problem which is discrete in nature. The algorithm begins

with an initialization phase, where the number of machines M, the number of parts P, the number of cells C, the population size and the fraction p_a of abandoned nests are defined. Then, an initial population of feasible solutions is created. After initialization, the proposed method iterates until a specific number of loops are met as shown in Algorithm 1.

Algorithm 1 The proposed cuckoo search algorithm

```
Generate an initial population of n host nests x_i (i = 1,...,n)
while (t < MaxGeneration) or (stop criterion) do
    Get a cuckoo l randomly by Lévy flights;
    Evaluate its quality according to its grouping efficacy Eff_l;
    Apply a local search to the generated solution x_l;
    Choose a nest r among n randomly;
    if (Eff_{x_l} > Eff_{x_r}) then
        Replace x_r by the new solution;
    end if
    Abandon a fraction p_a of worse nests and build new ones by Lévy flights
    Rank the solutions and find the current best
end while
```

10.3.2.1 Solution Representation

The encoding adapted in this paper is similar to the one used by Elbenani et al. [5]. Where the problem variable is represented as a vector with length of $P+M$: $(P_1,...,P_P|\ M_1,...,M_M)$. Where:

- P_j is the index of the cell including part j.
- M_i is the index of the cell including machine i.

To understand better, consider the following solution of a problem with 7 parts and 5 machines: $(2,1,3,2,3,1,3|\ 2,1,2,3,1)$. This solution includes three cells: cell 1 contains parts $\{2, 6\}$ and machines $\{2, 5\}$, cell 2 contains parts $\{1, 4\}$ and machines $\{1, 3\}$ and cell 3 contains parts $\{3, 5, 7\}$ and machine $\{4\}$.

10.3.2.2 Population Initialization

The initial solutions are generated randomly. Each machine i and part j are assigned randomly to a cell k. each cell must contain at least one machine and one part. To fix infeasibilities that may arise from an empty cell (cell without parts/machines), a repair process is activated. This process involves removing a part/ machine from the cell, including the most to the empty cell, which induces the smallest decrease of the grouping efficacy.

10.3.2.3 Nests

In the CS algorithm, the number of nests is fixed and equal to the population size. A nest is an individual of the population and an abandoned one entails replacing an individual in the population with a new one. Assuming that a cuckoo lays a single egg in one nest; each egg in a nest will be a solution represented by one individual in the population; and an egg of the cuckoo represents a new possible solution.

10.3.2.4 Lévy Flights

Lévy flights represent a model of random walks characterized by their step lengths which obey a power-law distribution as depicted in Eq. (10.10). In this paper, the difference between the two positions $(x_k^{(t)} - x_l^{(t)})$ represent the necessary movements to change from the actual solution given by the second, subtracting term, to the best obtained solution given by the first, subtracted term.

For each of the necessary movements, a random number between 0 and 1 is generated and if its value is less than the correspondent coefficient $Coef = \frac{\mu}{|\nu|^{1/\beta}}$, the movement is applied to the solution. A part movement coded as $(4,1)$ represents changing part 4 to cell 1. Similarly for a machine movement.

To understand better, we take the following example, let:

- $Coef = 0.7$;
- The current solution: $x_l = (3,2,1,2,3,1,3,1|1,2,3,1)$;
- The best solution obtained: $x_b = (2,3,1,1,3,1,3,2|1,2,3,3)$.

The necessary part movements, in this case, are $(1,2),(2,3),(4,1),(8,2)$ while the necessary machine movement is $(4,3)$. Five random values are generated and compared with $Coef$: $0.8, 0.3, 0.9, 0.6$ and 0.4. Therefore, the only part movements that would be applied in the current solution are $(2,3)$ and $(8,2)$, besides the machine movement $(4,3)$. The resulting solution will be: $x_l = (3,3,1,2,3,1,3,2|1,2,3,3)$ If the produced solution is unfeasible, the repair process described above is employed.

10.3.2.5 Local Search

In order to improve the quality of the solutions, the CS algorithm is combined with a search mechanism. The local search approach adopted is based on the one introduced by Goncalves and Resende [6]. Since it is simple, uses the same measure to compare the efficiency of the methods and generates good results.

Based on the initial set of machine groups of the incoming solution, each part is assigned to the cell that maximizes the grouping efficacy calculated by Eq. (10.1). If the modified solution is better than the current solution, the modified solution

replaces the current one. Afterwards, the process will restart by the assignment of the machines. This approach iterates by reassignment of parts, then reassignment of machines until the quality of the new solution does not exceed the quality of the last solution.

10.4 Computational Results

The proposed algorithm is tested on a set of 35 benchmark problems collected from the literature [5]. For each test problem, the number of machines (M), parts (P) and cells (C) are shown in Table 10.1. Besides, the minimum (*Min. Sol.*), the average (*Ave. Sol.*) and the maximum (*Max. Sol.*) solutions obtained, and the Best-Known solution (*Best Known Sol.*). Also, the computational time in seconds (*CPU. Time(s)*) required and the percentage of the gap (*Gap(%)*) are reported in Table 10.1, respectively. Preliminary tests showed that the following values are suitable to find best solutions:

- Population size $n = 50$;
- Portion of abandoned nests $p_a = 0.25$;
- Maximum number of iterations = 200.

For each benchmark problem, 10 independent runs of the algorithm with these parameters are performed. Table 10.1 summarizes the obtained results. The described algorithm is coded using Java, and running on a Personal Computer equipped with Pentium (R) Dual Core CPU T4500 clock at 2.30 GHz and 2 GB of RAM.

As shown in Table 10.1, the proposed method generates very good solutions in a reasonable computational time. Since the proposed method can reach the Best-Known solution of 31 benchmark problems. Among these problems, our method attains the Best-Known solution of 17 test problems throughout the 10 runs of the algorithm (*Min. Sol.* = *Ave. Sol.* = *Max. Sol.*), which prove the performance of the projected method. Regarding the remaining problems, the average solutions obtained are very close to the Best-Known solutions.

In order to confirm the effectiveness of the proposed method, the percentage of *gap* is calculated with respect to the Best-Known solution of each test problem, using Eq. (10.13). The results shows that the proposed approach has gaps in 4 problems only with the highest one is in problem $P27$ with 2.22% gap, as demonstrated in Table 10.1.

$$Gap = \frac{(Best\ Known\ Solution - Max.\ Sol.)}{Best\ Known\ Solution} \times 100 \tag{10.13}$$

To exhibit potentials of the proposed method, the obtained results are compared with the results of several algorithms developed in the literature for the CFP. Table 10.2 presents the results reported from these algorithms and the best solutions obtained for each problem.

Table 10.1 The numerical results of the proposed algorithm

No.	M	P	C	$Min.\,Sol.$	$Ave.\,Sol.$	$Max.\,Sol.$	$BestKnown\,Sol.$	$Gap(\%)$	$Time(s)$
P1	5	7	2	82.35	82.35	82.35	82.35	0	0.32
P2	5	7	2	69.57	69.57	69.57	69.57	0	0.33
P3	5	18	2	79.59	79.59	79.59	79.59	0	0.35
P4	6	8	2	76.92	76.92	76.92	76.92	0	0.32
P5	7	11	5	60.87	60.87	60.87	60.87	0	0.57
P6	7	11	4	70.83	70.83	70.83	70.83	0	0.46
P7	8	12	4	69.44	69.44	69.44	69.44	0	0.48
P8	8	20	3	85.25	85.25	85.25	85.25	0	0.52
P9	8	20	2	58.72	58.72	58.72	58.72	0	0.40
P10	10	10	5	75.00	75.00	75.00	75.00	0	0.65
P11	10	15	3	92.00	92.00	92.00	92.00	0	0.51
P12	14	24	7	71.64	72.02	72.06	72.06	0	3.70
P13	14	24	7	69.44	71.51	71.83	71.83	0	4.30
P14	16	24	8	53.26	53.26	53.26	53.26	0	5.79
P15	16	30	6	69.53	67.42	68.92	69.53	0	5.20
P16	16	43	8	56.25	56.80	57.53	57.53	0	14.26
P17	18	24	9	55.67	57.21	57.73	57.73	0	8.45
P18	20	20	5	41.61	42.59	43.45	43.45	0	2.89
P19	20	23	7	48.09	50.25	50.81	50.81	0	6.38
P20	20	35	5	77.91	77.91	77.91	77.91	0	5.04
P21	20	35	5	57.14	57.89	57.98	57.98	0	5.12
P22	24	40	7	100.0	100.0	100.0	100.0	0	14.35
P23	24	40	7	85.11	85.11	85.11	85.11	0	14.19
P24	24	40	7	73.51	73.51	73.51	73.51	0	14.65
P25	24	40	11	52.74	53.13	53.29	53.29	0	38.39
P26	24	40	12	47.95	48.19	48.61	48.95	0.69	44.32
P27	24	40	12	43.84	45.49	46.21	47.26	2.22	43.86
P28	27	27	5	54.82	54.82	54.82	54.82	0	5.23
P29	28	46	10	44.87	46.45	47.06	47.23	0.36	50.68
P30	30	41	14	62.42	62.92	63.31	63.31	0	83.68
P31	30	50	13	58.29	59.31	59.77	60.12	0.58	93.43
P32	30	50	14	49.72	50.46	50.83	50.83	0	85.97
P33	36	90	17	45.66	47.08	47.75	47.75	0	532.59
P34	37	53	3	59.26	59.97	60.64	60.64	0	9.58
P35	40	100	10	84.03	84.03	84.03	84.03	0	248.31

The comparator algorithms are ZODIAC method [1], GRAFICS method [14], evolutionary algorithm (EA) [6], simulated annealing (SA) [16], GRASP heuristic [2], genetic algorithm and large neighbourhood search (GA-LNS) [5].

The approaches: ZODIAC, GRAFICS, EA and GRASP do not allow for singletons (cells having less than two machines or two parts) which may affected the quality of the solutions in comparison with other approaches.

As can be seen, the proposed method achieves the best known solutions in 31 test problems out of the existing 35 test problems being 88.57%. The remaining problems in which the method performs a little bit worse than its comparator are 26,

Table 10.2 Performance of the proposed method compared to other approaches

No.	ZODIAC	GRAFICS	EA	SA	GRASP	GA-LNS	Proposed method	Bestknown Sol.
P1	73.68	73.68	73.68	**82.35**	73.68	**82.35**	**82.35**	**82.35**
P2	56.22	60.87	62.50	**69.57**	62.50	**69.57**	**69.57**	**69.57**
P3	–	–	**79.59**	**79.59**	**79.59**	**79.59**	**79.59**	**79.59**
P4	–	–	**76.92**	**76.92**	**76.92**	**76.92**	**76.92**	**76.92**
P5	39.13	53.12	53.13	**60.87**	53.13	**60.87**	**60.87**	**60.87**
P6	–	–	70.37	**70.83**	70.37	**70.83**	**70.83**	**70.83**
P7	68.30	68.30	68.29	**69.44**	**69.44**	**69.44**	**69.44**	**69.44**
P8	85.24	85.24	**85.25**	**85.25**	**85.25**	**85.25**	**85.25**	**85.25**
P9	58.33	58.33	**58.72**	**58.72**	**58.72**	**58.72**	**58.72**	**58.72**
P10	70.59	70.59	70.59	**75.00**	70.59	**75.00**	**75.00**	**75.00**
P11	**92.00**	**92.00**	**92.00**	**92.00**	**92.00**	**92.00**	**92.00**	**92.00**
P12	64.36	64.36	69.86	**72.06**	69.86	**72.06**	**72.06**	**72.06**
P13	65.55	65.55	69.33	**71.83**	69.33	**71.83**	**71.83**	**71.83**
P14	32.09	45.52	52.58	**53.26**	51.96	**53.26**	**53.26**	**53.26**
P15	67.83	67.83	67.83	68.99	67.83	**69.53**	**69.53**	**69.53**
P16	53.76	54.39	54.86	**57.53**	56.52	**57.53**	**57.53**	**57.53**
P17	41.84	48.91	54.46	**57.73**	54.46	**57.73**	**57.73**	**57.73**
P18	21.63	38.26	42.94	**43.45**	42.96	**43.45**	**43.45**	**43.45**
P19	38.96	49.36	49.65	**50.81**	49.65	**50.81**	**50.81**	**50.81**
P20	75.14	75.14	76.22	**77.91**	76.54	**77.91**	**77.91**	**77.91**
P21	–	–	58.07	**57.98**	58.15	**57.98**	**57.98**	**57.98**
P22	**100.0**	**100.0**	**100.0**	**100.0**	**100.0**	**100.0**	**100.0**	**100.0**
P23	**85.11**	**85.11**	**85.11**	**85.11**	**85.11**	**85.11**	**85.11**	**85.11**
P24	37.85	**73.51**	**73.51**	**73.51**	**73.51**	**73.51**	**73.51**	**73.51**
P25	20.42	43.27	51.88	**53.29**	51.97	**53.29**	**53.29**	**53.29**
P26	18.23	44.51	46.69	**48.95**	47.37	**48.95**	48.61	**48.95**
P27	17.61	41.67	44.75	**47.26**	44.87	46.58	46.21	**47.26**
P28	52.14	47.37	54.27	**54.82**	54.27	**54.82**	**54.82**	**54.82**
P29	33.01	32.86	44.37	**47.23**	46.06	47.06	47.06	**47.23**
P30	33.46	55.43	58.11	**63.31**	59.52	63.12	**63.31**	**63.31**
P31	46.06	56.32	59.21	59.77	60.00	**60.12**	59.77	**60.12**
P32	21.11	47.96	50.48	**50.83**	50.51	**50.83**	**50.83**	**50.83**
P33	32.73	39.41	42.12	47.14	45.93	**47.75**	**47.75**	**47.75**
P34	52.21	52.21	56.42	**60.64**	59.85	60.63	**60.64**	**60.64**
P35	83.92	83.92	**84.03**	**84.03**	**84.03**	**84.03**	**84.03**	**84.03**

The bold values indicate the solutions equal to the best-known solutions.

27, 29 and 31. For problem 21, data reported in EA and GRASP are inconsistent with the original data in the literature.

The numerical results indicate that the proposed method generates good results for the 35 problems, and that the performance of this method is rather constant on all types of problems ranging from small to large.

10.5 Conclusion

This paper presents in detail how a recently developed heuristic entitled cuckoo search algorithm is operating for solving the manufacturing cell formation problem. A remarkable contribution of the paper is the novelty of applying CS algorithm to this type of combinatorial optimization problems which require adapting its elements appropriately. The proposed method is combined with a local search mechanism in order to intensify the search and improve the quality of the solutions.

The results obtained in the computational experiments carried out show that on 31 out of 35 problems (88.57%) the results of the introduced method are among the best results. These results justify the superior performance of the method compared to other methods collected from the literature.

For future research, the extension of this approach could be applied to other variants of the cell formation problem having additional factors such as the real time manufacturing data, number of routings for each part, etc. Moreover, the proposed problem can be used in various similar clustering problems that involve the assignment of distinct objects in indistinguishable group.

References

1. M.P. Chandrasekharan, R. Rajagopalon, ZODIAC-an algorithm for concurrent formation of part-families and machine-cells. Int. J. Prod. Res. **25**, 835–850 (1987)
2. J.A. Diaz, D. Luna, R. Luna, A GRASP heuristic for the manufacturing cell. Top. **20**, 679–706 (2012)
3. C. Dimopoulos, A.M.S. Zalzala, Recent developments in evolutionary computation for manufacturing optimization: problems, solutions, and comparisons. IEEE Trans. Evol. Comput. **4**, 93–113 (2000)
4. B. Elbenani, J.A. Ferland, *CIRRELT: Cell Formation Problem Solved Exactly with the Dinkelbach Algorithm* (CIRRELT, Montreal, 2012), pp. 1–14
5. B. Elbenani, J.A. Ferland, J. Bellemare, Genetic algorithm and large neighbourhood search to solve the cell formation problem. Expert Syst. Appl. **39**, 2408–2414 (2012)
6. J.F. Goncalves, M.G.C. Resende, An evolutionary algorithm for manufacturing cell formation. Comput. Ind. Eng. **47**, 247–273 (2004)
7. A. Husseinzadeh Kashan, B. Karimi, A. Noktehdan, A novel discrete particle swarm optimization algorithm for the manufacturing cell formation problem. Int. J. Adv. Manuf. Tech. **73**, 1543–1556 (2014)
8. T.L. James, E.C. Brown, K.B. Keeling, A hybrid grouping genetic algorithm for the cell formation problem. Comput. Oper. Res. **34**, 2059–2079 (2007)
9. C.S. Kumar, M.P. Chandrasekharan, Grouping efficacy: a quantitative criterion for goodness of block diagonal forms of binary matrices in group technology. Int. J. Protein Res. **28**, 233–243 (1990)
10. G. Papaioannou, J.M. Wilson, The evolution of cell formation problem methodologies based on recent studies (1997–2008): review and directions for future research. Eur. J. Oper. Res. **206**, 509–521 (2010)
11. M.K. Sayadi, A. Hafezalkotob, S.G.J. Naini, Firefly-inspired algorithm for discrete optimization problems: an application to manufacturing cell formation. J. Manuf. Syst. **32**, 78–84 (2013)

12. S.M. Seyedhosseini, H. Badkoobehi, A. Noktehdan, Machine-part cell formation problem using a group based league championship algorithm. J. Promot. Manag. **21**, 55–63 (2015)
13. M. Solimanpur, A. Elmi, A tabu search approach for cell scheduling problem with makespan criterion. Int. J. Prod. Econ. **141**, 639–645 (2013)
14. G. Srinivasan, T.T. Narendran, GRAFICS- A nonhierarchical clustering algorithm for group technology. Int. J. Prod. Res. **29**, 463–478 (1991)
15. X.S. Yang, S. Deb, Cuckoo search via Lévy flights, in *IEEE NaBIC 2009: Proceedings of the World Congress on Nature and Biologically Inspired Computing*, Coimbatore (2009), pp. 210–214
16. K.C. Ying, S.W. Lin, C.C. Lu, Cell formation using a simulated annealing algorithm with variable neighbourhood. Eur. J. Ind. Eng. **5**, 22–42 (2011)