International Journal of Computational Intelligence Systems, Vol. 9, No. 4 (2016) 765-777

Cell formation and task scheduling considering multi-functional resource and part movement using hybrid simulated annealing

Chunfeng Liu [1] , Jufeng Wang [2] *

[1] *School of Management, Hangzhou Dianzi University, Economic and Technological Development Zone, Hangzhou, 310018, PR China*

E-mail: lcf_spring@163.com

[2] *Department of Mathematics, China Jiliang University, Economic and Technological Development Zone, Hangzhou, 310018, PR China*

E-mail: wang_jufeng@163.com

Received 23 December 2015

Accepted 15 April 2016

Abstract

This paper designs a non-linear integer mathematical model for the cellular manufacturing system (CMS) with dual-resource constrained setting. The multi-functional machines and the multi-skilled workers need to be grouped and assigned to the cells. Moreover, each operation of the parts has different processing times if processed by different machines or workers. Each part with operation sequence is allowed to move from one machine to another for processing subsequent operation, which might reduce processing time although it will incur additional movement time. In order to solve the simultaneous and intertwined optimization problem, a hybrid simulated annealing (HSA) which embedding priority rule based heuristic algorithm is proposed to minimize the makespan. Computational experiments are conducted to show that the proposed HSA performs well with respect to accuracy and efficiency of solution than the traditional simulated annealing algorithm.

Keywords: Cellular manufacturing system; Cell formation; Group scheduling; Simulated annealing; Operation sequence

1. Introduction

With the intense competitive market and rapid variation of customer needs, companies should adjust their production strategies in time. They need to produce shorter life-cycle products as well as mid-volume and mid-variety product mixes. In such environment, cellular manufacturing system (CMS) is becoming increasingly important for their advantages such as cutting down the costs, improving the quality of the products and strengthening the manufacturing flexibility[1,2]. An important problem of the CMS is cell formation (CF), i.e., the physical division of the facilities, where the machines (workers) are grouped into cells and parts are grouped into part families. Another significant problem is the group scheduling which involves decisions on task dispatching rules as well as task timetabling methods.

For the basic cell formation problem with ma-

* Corresponding author

chine assignment, Mahdavi et al.[3] proposed a model for the CF based on cell utilization concept. The objective is to minimize the exceptional elements and number of voids in cells to achieve the higher performance of cell utilization. Tavakkoli-Moghaddam et al.[4] discussed the CF problem in dynamic condition by using genetic algorithm, simulated annealing and tabu search. Arkat et al.[5] formulated the CF problem as an integer linear programming model. They proposed three types of branch and bound algorithms to minimize the total number of intercellular movements. Paydar et al.[6] modeled the CMS as a multiple departures single destination multiple travelling salesman problem, and proposed a simulated annealing algorithm to solve it. Nouri and Hong[7] took into consideration number of voids in cells and a number of exceptional elements based on the operational time of parts which are required for processing in the machines. A bacteria foraging algorithm was applied to solve the CF problem. Hosseinabadi Farahani and Hosseini[8] developed an ant colony optimization approach for the CF Problem to minimize grouping efficacy. In some articles[9,10,11], mathematical models were designed for the the CF problem problems. The optimization softwares such as LINGO and CPLEX have been used to solve these models.

For the extensional cell formation problem with worker (and machine) assignment, Egilmez et al.[12] designed three stochastic non-linear mathematical models of manpower allocation to manufacturing cells, and developed a four-phased hierarchical methodology to maximize the production rate. They considered normally distributed operation times and customer demand in all models. Süer et al.[13] investigated manpower allocation problem in CMS, and examined three different sharing strategies (no operator sharing allowed, sharing allowed without restrictions, sharing allowed with restrictions). They concluded that the models with little or no restrictions yield a higher production rate than the model with no operator sharing allowed. Sakhaii et al.[14] developed a model which is incorporated with dynamic cell formation, inter-cell layout, machine reliability, operator assignment, alternative process routings and production planning concepts. They adopted a robust optimization approach to minimize the costs of machine breakdown and relocation, operator training and hiring, inter-intra cell part trip, and shortage and inventory. Mahdavi et al.[15] proposed an integrated dynamic CMS model considering multi-period production planning. The authors used LINGO package to minimize the sum of machine and human resource costs, reconfiguration cost, inter-cell material handling cost, inventory holding cost, and backorder cost. Bootaki et al.[16] considered a robust design to configure cells in the dynamic cellular manufacturing system. A new method of percentage multi-choice goal programming is innovated to minimize the inter-cell movements and maximize the worker and machine utilization. Liu et al.[17] proposed a discrete bacteria foraging algorithm for the assignment of workers and machines in the cell formation and task scheduling problem, in order to minimize the inter-cell material handling costs as well as the fixed and operating costs of workers and machines. Liu et al.[18] considered worker assignment and production planning in the dynamic cell formation of fiber connector manufacturing industry. Due to the learning and forgetting effects of workers, the production rate of each workstation will often change, and the bottleneck workstation may transfer to another one in the next period. Reassigning multi-skilled workers may increase the production rate of bottleneck workstation. They suggested a hybrid bacteria foraging algorithm embedding two-phase based heuristic to minimize the backorder cost and holding cost of inventory.

In comparison with the cell formation problem, there are only a few articles discussing group scheduling, which is also called parts scheduling sometimes. Taghavifard[19] studied cellular manufacturing scheduling problem with sequence-dependent setup times, and used ant colony optimization and GA to solve it. Halat and Bashirzadeh[20] discussed scheduling operations of manufacturing cells considering sequence-dependent family setup times and intercellular transportation times, and suggested a GA-based heuristic for this problem. Solimanpur and Elmi[21] investigated group scheduling in buffer-constrained flow shop cells, and proposed a tabu search method to minimize the makespan. Elmi et

al.[22] addressed job shop cell scheduling problem with inter-cell movement and reentrant parts, and suggested a simulated annealing algorithm to minimize the makespan.

Due to the high complexity of CMS with dual-resource constrained setting, the cell formation and group scheduling issues are normally studied independently. So far, the problem of the CMS simultaneously involving multi-functional machines, multi-skilled workers and operation sequence has not been fully investigated, especially for the objective of minimizing the makespan with the impact of part inter-cell and intra-cell movement on task scheduling. To the best of the authors' knowledge, no related results have been available in the literature, which motivate the present study.

In this paper, we investigate the CMS with two related aspects. On one hand, the machines have multiple functions to process several different tasks, and the workers have multiple skills to operate every machine, but each machine must be operated by only one worker. So the machines and workers need to be grouped and assigned to the cells. On the other hand, each operation has different processing times if processed by different machines or workers. Each part with operation sequence is allowed to move from one machine to another for processing subsequent operation, which might reduce processing time although it will incur additional movement time. In order to solve the simultaneous and intertwined optimization problem, a hybrid simulated annealing (HSA) which embedding priority rule based heuristic algorithm (PRBHA) is proposed to minimize makespan.

The remainder of this paper is organized as follows: The mathematical model integrating cell formation and task scheduling is formulated in Section 2. The PRBHA is developed for high quality initial solution in Section 3. In Section 4, the HSA is proposed for further search to obtain a global optimum. The performance of proposed HSA and the traditional simulated annealing are compared through numerical experiments in Section 5. Finally, the paper closes with a general discussion of the proposed approach as well as a few remarks on future research directions in Section 6.

2. Problem statement and formulation

In this section the problem of cell formation and task scheduling is formulated as a non-linear 0-1 integer programming model. The objective is minimizing the makespan (i.e., maximum completion time of all parts). The main constraints are the multi-functional machines, multi-skilled workers, operation sequence, and part movement. The following assumptions are made for the considered cellular manufacturing system throughout the paper.

1. *Machine and worker assumptions*: The number of machines, the number of worker types, and the number of each worker type are known in advance. The number of all machines are equal to the number of all workers. Each machine can be operated by any one worker.
2. *Task assumption*: The number of parts and the number of operations of each part are known in advance. All parts should be finished in the planning horizon.
3. *Processing assumption*: Each operation of a part can be processed on several different machines. The processing of each operation is not allowed to be interrupted, which implies that each operation of a part is processed by only one machine that have the function to handle it. The processing time of each operation depends on the selected machine and assigned worker type.
4. *Operation sequence assumption*: There exists chain precedence relationship among operations of each part. However, there is no precedence constraint among different parts.
5. *Cell size assumption*: The maximum of the cell size in terms of the number of machines is specified in advance, for too many machines in a cell may generate cluttered flows in a cell due to many routes.
6. *Part movement assumption*: The parts are allowed to move among different machines in the same cell or different cells, and the time spent on movement is known in advance.

Notation

Indices

w	Index for worker type.
c	index for cell.
p	Index for part.
k	Index for operation.
t	Index for time.
m	Index for machine.

Input parameters

W	Number of worker types.
P	Number of parts.
K_p	Number of operations of part p $(p = 1,2,\cdots,P)$.
M	Number of machines.
N_w	Number of worker type w $(w = 1,2,\cdots,W)$.
C	Number of cells.
B_u	Upper bound of cell size (measured in the number of machines) which is assumed to be $\ulcorner M/C \urcorner$.
T_{pkmw}	Processing time of operation k of part p on machine m with worker type w $(k = 1,2,\cdots,K_p; m = 1,2,\cdots,M)$.
Q_{pkm}	1 if operation k of part p can be processed on machine m, and 0 otherwise.
$Y_{pkmcm'c'}$	Movement time of part p whose operation $k-1$ was processed on machine m in cell c and operation k will be processed on machine m' in cell c' $(m,m' = 1,2,\cdots,M;$ $c,c' = 1,2,\cdots,C)$. $Y_{pkmcm'c'} = 0$ when $k = 1$ or $(m = m'$ and $c = c')$.
θ	A sufficient large number which is greater than the makespan definitely.

Decision variables

Z_{mc}	1 if machine m is assigned to cell c , and 0 otherwise.
X_{mw}	1 if machine m is operated by worker type w, and 0 otherwise.
S_{pktm}	1 if operation k of part p starts to be processed on machine m at time point t, and 0 otherwise $(t = 1,2,\cdots,\theta)$.

The proposed problem is formulated as the following non-linear 0-1 integer programming:

$$\text{Min} \quad C_{\max} = \max_{p=1,2\cdots,P} \left\{ \sum_{m=1}^{M} \sum_{t=0}^{\theta} t S_{pK_ptm} + \sum_{w=1}^{W} \sum_{m=1}^{M} \sum_{t=0}^{\theta} S_{pK_ptm} X_{mw} T_{pK_pmw} \right\} \tag{1}$$

$$\text{s.t.} \quad S_{pktm} \leqslant Q_{pkm}, \quad \forall p,k,t,m \tag{2}$$

$$\sum_{c=1}^{C} Z_{mc} = 1, \quad \forall m \tag{3}$$

$$\sum_{w=1}^{W} X_{mw} = 1, \quad \forall m \tag{4}$$

$$\sum_{m=1}^{M} X_{mw} = N_w, \quad \forall w \tag{5}$$

$$\sum_{m=1}^{M} Z_{mc} \leqslant B_u, \quad \forall c \tag{6}$$

$$\sum_{t=0}^{\theta} \sum_{m=1}^{M} S_{pktm} = 1, \quad \forall p,k \tag{7}$$

$$\sum_{p=1}^{P} \sum_{k=1}^{K_p} \sum_{w=1}^{W} \sum_{\tau=t}^{t+T_{pkmw}-1} S_{pk\tau m} X_{mw} \leqslant 1, \quad \forall m,t \tag{8}$$

$$\sum_{t=0}^{\theta} \sum_{m=1}^{M} t S_{pktm} - \sum_{t=0}^{\theta} \sum_{m=1}^{M} t S_{p,k-1,t,m} \geqslant \sum_{w=1}^{W} \sum_{t=0}^{\theta} \sum_{m=1}^{M} S_{p,k-1,t,m} X_{mw} T_{pkmw} + \sum_{t'=0}^{\theta} \sum_{m'=1}^{M} \sum_{t=0}^{\theta} \sum_{m=1}^{M} \sum_{c=1}^{C} \sum_{c'=1}^{C} Y_{pkmcm'c'} S_{pktm} S_{p,k-1,t,m} Z_{mc} Z_{m'c'}, \quad \forall p, \quad \forall k = 2,3,\cdots,K_p \tag{9}$$

$$Z_{mc}, X_{mw}, S_{pktm} \in \{0,1\}, \quad \forall m,w,c,p,k,t \tag{10}$$

The objective function (1) is to minimize the makespan $C_{\max}$, where $\sum_{m=1}^{M} \sum_{t=0}^{\theta} t S_{pK_ptm}$ represents the start time of the last operation of part p, and $\sum_{w=1}^{W} \sum_{m=1}^{M} \sum_{t=0}^{\theta} S_{pK_ptm} X_{mw} T_{pK_pmw}$ represents the processing time of the last operation of part p. Constraint (2) guarantees that each operation of a

part is processed on the machine that can process the operation. Constraint (3) ensures that each machine is assigned and only assigned to one cell. Constraints (4) and (5) imply that each machine is operated by one and only one worker. Constraint (6) shows that the number of machines in the same cell can not exceed the upper bound of cell size. Constraint (7) indicates that each operation k of part p must start once. Constraint (8) guarantees that a machine can at most process one operation at a time. Constraint (9) provides the precedence relationship between consecutive operations of part p. Constraint (10) ensures that the decision variables are binary variables.

3. Priority rule based heuristic algorithm

Heuristics have taken an important position in the search of solutions in many combinatory problems, as it is simple, easy to implement and can be embedded in more sophisticated heuristics or metaheuristics for determining initial feasible solutions that can be improved in further stages. We develop a priority rule based heuristic algorithm (PRBHA) consisting of several iterations.

Four task-sets associated with each iteration are defined. The unscheduled operations whose immediate predecessor has been scheduled are in the set H. The unscheduled operations whose immediate predecessor has been completed upon scheduling time point t are in the set $\widehat{D}$. The operations which are available for scheduling on selected machine with respect to precedence constraints but yet unscheduled are in the decision set D. The unscheduled operations of part p are in the set V_p. At each iteration, a prior task is selected according to the earliest finishing time first (EFT) rule and inserted inside a partial schedule on a selected machine. This priority rule method enables the eligible tasks to be processed as soon as possible, which gives rise to a compact schedule.

The variables used for the PRBHA are summarized as follows:

u	the counter of iteration
H	the set of unscheduled operations whose immediate predecessor has been scheduled
t	the scheduling time point
$\widehat{D}$	the set of unscheduled operations whose immediate predecessor has been completed upon scheduling time point t
D	the decision task-set upon scheduling time point t
f_1	the flag variable which is equal to 1 when $\widehat{D}$ is not empty and 0 otherwise
f_2	the flag variable which is equal to 1 when D is not empty and 0 otherwise
V_p	the set of unscheduled operations of part p
I_m	the start idle time of machine m
$\widehat{I}$	the set of start idle times of all machines
ϖ_{pk}	the set of machines that can process the operation k of part p
ω_{pk}	the machine that process the operation k of part p
ε_m	the cell to which the machine m is assigned
W_m	the worker type that operates the machine m
k_p	the operation k of part p (let $k^*_{p^*}$ denote the prior task; p^* and k^* indicate the selected prior part and prior operation, respectively)
FT_{pk}	the finish time of the operation k of part p
ST_{pk}	the start time of the operation k of part p

We give a pseudo-code description of the priority rule based heuristic algorithm (**Algorithm 1**) which consists of N iterations (N is the sum number of operations of all parts, i.e., $N = K_1 + K_2 + \cdots + K_P$).

In Step 1, some variables $u, FT_{p0}, I_m, V_p, \varepsilon_m$ and W_m are initialized. Assuming that each part p has a dummy operation $k = 0$ with 0 unit of processing time at the beginning of the part. So let $FT_{p0} = 0, \forall p$. In Step 2, each job is scheduled at each iteration until $u = N$. In Step $\langle 1 \rangle$, $\widehat{I}$ is computed to represent the set of I_m . In Step ♣1, the scheduling time point t is determined by the minimum start

idle time of machines. Step ♣2 computes the set $\widehat{D}$ which denotes the unscheduled operations whose immediate preceding operation has been completed at the time point t. Step ♡2 computes the set $\{m^*\}$ which denotes the machines that are idle at the time point t. The decision task-set D is computed in the Step ♠1, which shows the eligible operations that can be scheduled on machine m^* upon the time point t. In Step ♢2, the EFT rule is used to determine a prior task processed on machine m^*, i.e., the task in D with minimum finish time is selected as the prior task $k^*_{p^*}$. The start idle time of machine m^* is updated in Step ♢3. The start and finish times of $k^*_{p^*}$ are recorded in Steps ♢4 and ♢5, respectively. The machine $\omega_{p^*k^*}$ that processes $k^*_{p^*}$ is recorded in Step ♢6. In Step ♢8, the makespan is computed and the procedure is terminated if the counter of iteration u is equal to N. The sets V_{p^*} and $\widehat{D}$ are updated in Step ♢9. In Step $\widetilde{\heartsuit}$1, the scheduling time point t is postponed to the next minimum start idle time if $f_1 = 0$ or $f_2 = 0$, because there is no operation which can be scheduled at the time point t.

4. Hybrid simulated annealing algorithm

Simulated annealing (SA) is a stochastic neighborhood search technique which was introduced by Kirkpatrick et al.[23]. It is derived from the similarity between the process of annealing of solids and the method of solving combinatorial optimization problems.

The algorithm starts with a random trial solution as current solution. A neighborhood of current solution is then searched, by some neighborhood search structures. If the neighborhood is better than the current one, it is accepted as the new current solution. Otherwise, it is accepted by a certain probability. The principal advantage of SA is to escape from local optima by accepting some nonimproving solutions at each temperature level to reach a global optimum. In the beginning, the probability of accepting nonimproving solutions is high, but as the temperature drops, the probability of accepting nonimproving solutions decreases.

Because the proposed cell formation and task scheduling problem is much complicated, a basic SA may not perform well to solve it. Therefore, we suggest a hybrid simulated annealing (HSA) which combines the PRBHA approach and three neighborhood strategies with traditional simulated annealing.

4.1. Initial solution

The PRBHA can generate feasible solution represented by the variables ε_m, W_m, ω_{pk}, ST_{pk} and makespan $C_{\max}$. ε_m and W_m correspond to decision variables Z_{mc} and X_{mw}, respectively. ω_{pk} and ST_{pk} correspond to decision variable S_{pktm}. The HSA employs the feasible solution as initial solution for further search.

4.2. Neighborhood generation strategy

Well-designed solution mutation(SM) operators are significant to the success of HSA. In this research, three different mutation strategies are developed as follows:

1. Machine-cell mutation(SM1):
   Randomly select a machine m_1 in cell c_1 and move it to different cell c_2 if the number of machines in cell c_2 does not reach the upper cell size limit, otherwise randomly select a machine m_2 in cell c_2 and then exchange machines m_1 and m_2 between cells c_1 and c_2.

2. Machine-worker mutation(SM2):
   Randomly select two machines operated by two different types of workers, and then exchange their workers.

3. part-operation-machine mutation(SM3):
   Randomly select an operation of a part, and then reassign it to other machine that can process the operation.

The objective function values (i.e., makespan $C_{\max}$) of the neighborhood solutions can be computed by the revised backward recursion algorithm (**Algorithm 2**).

Algorithm 1: Priority Rule Based Heuristic Algorithm (PRBHA)

1. Initialize: $u=0;\quad FT_{p0}=0,\forall p;\quad I_m=0,\forall m;\quad V_p=\{1,2,\ldots,K_p\},\forall p;$
   Randomly generate ε_m and $W_m,\forall m$
2. **WHILE** (1) **DO**
   $\langle 1\rangle$. $\widehat{I}=\{I_m|m=1,2,\ldots,M\}$
   $\langle 2\rangle$. $f_1=0,f_2=0$
   $\langle 3\rangle$. $H=\{k_p|p\in\{p|V_p\neq\emptyset,p=1,2\cdots P\},k_p=\min\{k_p|k_p\in V_p\}\}$
   $\langle 4\rangle$. **WHILE** (1) **DO**
   ♣1. $t=\min\{\tau|\tau\in\widehat{I}\}$
   ♣2. $\widehat{D}=\{k_p|FT_{p,k-1}\leqslant t,k_p\in H\}$
   ♣3. **IF** $(\widehat{D}\neq\emptyset)$ **THEN**
   ♡1. $f_1=1$
   ♡2. $\{m^*\}:I_{m^*}\leqslant t$
   ♡3. **FOR** each m^* from $\{m^*\}$:
   ♠1. $D=\left\{k_p|k_p\in\widehat{D},m^*\in\varpi_{pk}\right\}$
   ♠2. **IF** $(D\neq\emptyset)$**THEN**
   ♢1. $f_2=1$
   ♢2. Let $\alpha_{pk}=\max\{t,FT_{p,k-1}+Y_{pk}\omega_{p,k-1}C_{\omega_{p,k-1}m^*}C_{m^*}\}$,
   $k^*_{p^*}:ft_{p^*k^*}=\min\left\{ft_{pk}|ft_{pk}=\alpha_{pk}+T_{pkm^*W_{m^*}},\forall k_p\in D\right\}$
   ♢3. $I_{m^*}=\max\{t,FT_{p^*,k^*-1}+Y_{p^*k^*}\omega_{p^*,k-1}C_{\omega_{p^*,k-1}m^*}C_{m^*}\}+T_{p^*k^*m^*W_{m^*}}$
   ♢4. $ST_{p^*k^*}=\max\{t,FT_{p^*,k^*-1}+Y_{p^*k^*}\omega_{p^*,k-1}C_{\omega_{p^*,k-1}m^*}C_{m^*}\}$
   ♢5. $FT_{p^*k^*}=I_{m^*}$
   ♢6. $\omega_{p^*k^*}=m^*$
   ♢7. $u:=u+1$
   ♢8. **IF** $(u=N)$ **THEN**
   *1. Compute $C_{\max}=\max\{FT_{pK_p},p\in 1,2,\cdots,P\}$
   *2. **RETURN**
   ♢9. $V_{p^*}=V_{p^*}\backslash\{k^*_{p^*}\}$, $\widehat{D}=\widehat{D}\backslash\{k^*_{p^*}\}$
   ♠3. **IF** $(\widehat{D}=\emptyset)$ **THEN BREAK**
   ♣4. **IF**$(f_1=0$ or $f_2=0)$ **THEN**
   $\widetilde{\heartsuit}$1. $\widehat{I}:=\widehat{I}\backslash\{t\}$
   $\widetilde{\heartsuit}$2. clear $\widehat{D}$
   ♣. **ELSE**
   $\overline{\heartsuit}$1. Clear $\widehat{D},H,\widehat{I}$
   $\overline{\heartsuit}$2. **BREAK**

4.3. Cooling schedule

There are some parameters existing in the cooling schedule, which include initial temperature, final temperature, cool rate, and Markov chain length. They are determined through pretest after referring to some literature about SA.

1. Cooling rate:
   The temperature $\hbar$ should be gradually decreased from initial temperature $\hbar_0=200$ in order to reach slow cooling. The temperature

Algorithm 2: Revised Backward Recursion Algorithm (RBRA)

1. Initialize: $FT_{p0}=0;\quad I_m=0;\quad H=\{1_p|p=1,2\cdots P\};\quad Y_{p1mcm'c'}=0$
2. **FOR** $g=1$ to N
   - 2.1. Randomly select k_p from H
   - 2.2. $ST_{pk}=\max\left\{I_{\omega_{pk}},FT_{p,k-1}+Y_{pk\omega_{p,k-1}C_{\omega_{p,k-1}}\omega_{pk}C_{\omega_{pk}}}\right\}$
   - 2.3. $FT_{pk}=ST_{pk}+T_{pk\omega_{pk}W_{\omega_{pk}}}$
   - 2.4. $I_{\omega_{pk}}=FT_{pk}$, and update H
3. $C_{\max}=\max\{FT_{pK_p}|p=1,2\cdots P\}$

is decreased by using the common equation $\hbar:=\alpha\hbar$, where α is cooling rate and its value is set to 0.95.

2. Markov chain length(MCL):
   MCL is chosen in such a way that a thermal equilibrium can be attained for each temperature level. MCL is denoted by $L_{\max}=200$.

3. Termination condition:
   Let x_{best} denote the best solution up to now. The HSA algorithm is stopped if x_{best} does not change after three consecutive temperature levels or the final temperature ($\hbar_f=0.5$) is reached.

4.4. The pseudo-code of the HSA

The pseudo-code of the HSA is given as **Algorithm 3**. The main idea of optimization is that: If $C_{\max}$ of neighborhood solution x_l is less than $C_{\max}$ of current solution x_c, accept x_l as current solution, otherwise accept x_l as current solution by certain probability, which can escape from local optima to reach a global optimum.

5. Computational experiments

The following numerical experiments are conducted to evaluate the performance of the HSA compared to the traditional SA which randomly generates initial solution. The performance is evaluated by use of several impact factors including the number of parts (P), the number of machines (M), the number of worker types (W), the upper cell size limit (B_u), the processing time (T_{pkmw}), and the number of machines that can process the operation (β). The value range of these parameters are determined to ensure that they can reflect small-sized, middle-sized and large-sized instances.

To test the effects of varying P, M, W and B_u, four different values of P are used, including 25, 50, 75 and 100, four different values of M are used, including 10, 20, 30 and 40, and four different values of W and B_u are used, including 2, 4, 6 and 8. Moreover, to determine whether the range of T_{pkmw} and β may have any impact on the performance of the HSA, four different distributions of T_{pkmw} are used, including $T_{pkmw}\sim$ DU[5, 10], $T_{pkmw}\sim$ DU[5, 20], $T_{pkmw}\sim$ DU[5, 30] and $T_{pkmw}\sim$ DU[5, 40], and four different distributions of β are used, including $\beta\sim$ DU[1, 5], $\beta\sim$ DU[1, 6], $\beta\sim$ DU[1, 7] and $\beta\sim$ DU[1, 8], where DU[a, b] represents a discrete uniform distribution with an integer range from a to b.

Six sets of numerical experiments are conducted. In the first set, P is allowed to vary, given $M=20$, $W=5$, $B_u=2$, $T_{pkmw}\sim$ DU[5,20] and $\beta\sim$ DU[1,4]. In the second set, M is allowed to vary, given $P=20$, $W=5$, $B_u=2$, $T_{pkmw}\sim$ DU[5,20] and $\beta\sim$ DU[1,4]. In the third set, W is allowed to vary, given $M=20$, $P=20$, $B_u=2$, $T_{pkmw}\sim$ DU[5,20] and $\beta\sim$ DU[1,4]. In the fourth set, B_u is allowed to vary, given $M=30$, $P=30$, $W=5$, $T_{pkmw}\sim$ DU[5,20] and $\beta\sim$ DU[1,4]. In the fifth set, the distribution of T_{pkmw} is allowed to vary, given $M=20$, $P=30$, $W=5$, $B_u=8$ and $\beta\sim$ DU[1,4]. In the sixth set, the distribution of β is allowed to vary, given $M=20$, $P=30$, $W=5$, $B_u=8$ and $T_{pkmw}\sim$ DU[5,40]. Given $K_p\sim$ DU[20, 40],

Algorithm 3: Hybrid Simulated Annealing (HSA)

1. Initialize parameters:
   1.1. $\hbar_0 = 200, \hbar_f = 0.5, \alpha = 0.95, Total = 0, Change = 0, Unchange = 0.$
   1.2. Generate initial solution x_0 and objective function value $C_{\max}(x_0)$ by **Algorithm 1**;
   Set $x_c = x_0$ and $C_{\max}(x_c) = C_{\max}(x_0)$
2. **WHILE** $(\hbar > \hbar_f)$ **DO**
   2.1. **WHILE** $(Total < L_{\max})$ **DO**
   2.1.1. Randomly generate a number $r_1 \sim$ U[0,1]
   (U[0,1] represents an uniform distribution real interval [0,1])
   IF $(r_1 > 0.5)$
   Randomly generate a number $r_2 \sim$ U[0,1]
   IF $(r_2 > 0.5)$
   Generate neighborhood x_l by SM1
   Compute $C_{\max}(x_l)$ by **Algorithm 2**
   ELSE
   Generate neighborhood x_l by SM2
   Compute $C_{\max}(x_l)$ by **Algorithm 2**
   ELSE
   Generate neighborhood x_l by SM3
   Compute $C_{\max}(x_l)$ by **Algorithm 2**
   2.1.2. **IF** $(C_{\max}(x_l) \leqslant C_{\max}(x_c))$
   $x_c = x_l$
   IF $(C_{\max}(x_c) \leqslant C_{\max}(x_{best}))$
   $x_{best} = x_c, Change := Change + 1, Unchange = 0$
   ELSE
   Randomly generate a number $r_3 \sim$ U[0,1]
   Let $\Delta = C_{\max}(x_l) - C_{\max}(x_c)$
   IF $(e^{-\Delta/\hbar} > r_3)$
   $x_c = x_l$
   2.1.3. $Total := Total + 1$
   2.2. **IF** $(Change = 0)$
   $Unchange := Unchange + 1$
   2.3. **IF** $(Unchange = 3)$
   RETURN $C_{\max}(x_{best})$
   2.4. $\hbar := \alpha \times \hbar, Total = 0, Change = 0$
3. **RETURN** $C_{\max}(x_{best})$

$Y_{pkmcm'c'} = 2$ if $c = c'$ and $m \neq m'$, and $Y_{pkmcm'c'} = 20$ if $c \neq c'$ and $m \neq m'$ in these experiments.

The experiments have been performed on a Pentium-based Lenovo-compatible personal computer with 2.3 GHz clock-pulse and 4 GB RAM. The HSA and SA have been coded in C++, compiled with the Microsoft Visual C++ 9.0 compiler, and tested under Microsoft Windows 7 operating system.

The experimental results are presented in Tables 1∼6. Let HSA-$C_{\max}$ and SA-$C_{\max}$ denote the objective function values of the problem using the HSA and SA, respectively. Each table entry represents

the minimum, maximum and average of its associated 10 instances. For example, 10 random instances are generated when $P = 25$ in Table 1, and another 10 random instances are generated when $P = 50$. Let $\Delta C_{\max}$ denote the declining percentage of average HSA-$C_{\max}$ over average SA-$C_{\max}$. Let HSA-CPU and SA-CPU denote the mean CPU times of the HSA and SA without including input and output times, respectively. Let ΔCPU denote the declining percentage of HSA-CPU over SA-CPU.

For example, when $P = 25$ in Table 1, we generate 10 random instances. The average objective function values using the HSA and SA are 1241.3 and 2186.2, respectively. The HSA is better than the SA by 43.22% in terms of the average objective function value. The mean CPU times of the HSA and SA are 52.00s and 97.14s, respectively. The HSA is faster than the SA by 46.47% in terms of the mean CPU time.

From the obtained results, we can see that $\Delta C_{\max}$ reaches 33.60%$\sim$55.83% and ΔCPU reaches 32.51%$\sim$56.96% . That is to say, the HSA can get higher quality solution within less CPU time than the SA regardless of the variation of six impact factors P, M, W, B_u, T_{pkmw} and β. The reason lies in that the PRBHA algorithm plays an important role in generating good initial solution of the HSA to avoid the blind search of SA at the beginning while exploring the solution space.

6. Conclusions

This paper investigates a novel optimization model of cellular manufacturing system (CMS) under dual-resource constrained setting along with a hybrid simulated annealing (HSA) embedding priority rule based heuristic algorithm (PRBHA). The advantage of the proposed model is simultaneously integrating cell formation and task scheduling by assuming multi-functional resource and inter-intra cell part trip. Other features are operation sequence and maximal cell size limit. Considering part trip might reduce processing time although it will incur additional movement time, the objective of the problem is minimizing the makespan. The PRBHA schedules a prior task on a prior machine operated by a prior worker according to the priority rule at each iteration, which helps to generate a high quality initial solution for further search. Experimental results show that the quality of solutions obtained by the HSA is better than the SA in terms of accuracy and efficiency no matter what variation of some important parameters.

The major contribution of this paper includes two aspects: (1) There is a trade-off among no movement, intra-cell movement and inter-cell movement for parts in the proposed problem. The intertwined issue as well as multi-functional machines and multi-skilled workers are incorporated into the nonlinear 0-1 integer programming model for optimization. (2) Operation sequence may hinder some immediate successors to start in time even if there are machines available in cells. However, the PRBHA is suggested to generate a compact schedule. The mechanism is that at each iteration, a prior task is selected according to the EFT rule and inserted inside a partial schedule, which enables the eligible tasks to be processed as soon as possible. The PRBHA helps the HSA to improve the solution quality.

An important research direction that may be pursued in the future is to extend chain precedence constraints of operations to arbitrary precedence constraints of parts and operations. The other potential interest is to develop more efficient priority rules combined in the heuristics, embed them in some population based evolutionary algorithms, and compare the performance of these metaheuristics.

Acknowledgment

This research was supported by the Zhejiang Provincial Natural Science Foundation of China (Grant No. LY14G020014), Humanities and Social Sciences Youth Foundation of the Ministry of Education (Grant No. 14YJC630089), Zhejiang Provincial Key Research Base of Humanities and Social Sciences in Hangzhou Dianzi University (Grant No. ZD05-201601), and the Research Center of Information Technology & Economic and Social Development (Key Research Institute of Philosophy and Social Sciences of Zhejiang Province). The authors are grateful for the financial supports.

Table 1: Comparison between HSA and SA with different number of parts (P)
$(M = 20,\ W = 5,\ B_u = 2,\ T_{pkmw} \sim \mathrm{DU}[5,20],\ \beta \sim \mathrm{DU}[1,4])$

P	HSA-$C_{\max}$			SA-$C_{\max}$			$\Delta C_{\max}$ (%)	HSA-CPU (s)	SA-CPU (s)	ΔCPU (%)
	MIN	MAX	AVE	MIN	MAX	AVE				
25	1176	1319	1241.3	2141	2207	2186.2	43.22	52.00	97.14	46.47
50	1570	1803	1685.7	3253	3573	3428.6	50.83	151.57	344.60	56.02
75	2173	2307	2228.1	4000	4183	4140.5	46.19	242.93	512.88	52.63
100	2763	2899	2841.1	4912	5159	5081.3	44.09	424.58	737.42	42.42

Table 2: Comparison between HSA and SA with different number of machines (M)
$(P = 20,\ W = 5,\ B_u = 2,\ T_{pkmw} \sim \mathrm{DU}[5,20],\ \beta \sim \mathrm{DU}[1,4])$

M	HSA-$C_{\max}$			SA-$C_{\max}$			$\Delta C_{\max}$ (%)	HSA-CPU (s)	SA-CPU (s)	ΔCPU (%)
	MIN	MAX	AVE	MIN	MAX	AVE				
10	1162	1426	1279.3	2247	2468	2349.4	45.55	37.22	81.97	54.59
20	1055	1181	1130.9	1786	1939	1895.9	40.35	34.22	79.51	56.96
30	1219	1332	1277.3	1957	2075	2004.9	36.29	42.68	75.76	43.66
40	1261	1314	1283.0	1860	1988	1932.1	33.60	49.60	92.74	46.52

Table 3: Comparison between HSA and SA with different number of worker types (W)
$(M = 20,\ P = 20,\ B_u = 2,\ T_{pkmw} \sim \mathrm{DU}[5,20],\ \beta \sim \mathrm{DU}[1,4])$

W	HSA-$C_{\max}$			SA-$C_{\max}$			$\Delta C_{\max}$ (%)	HSA-CPU (s)	SA-CPU (s)	ΔCPU (%)
	MIN	MAX	AVE	MIN	MAX	AVE				
2	1113	1204	1160.5	1955	2076	2026.0	42.72	40.91	88.94	54.00
4	1168	1274	1213.5	1891	2073	2010.1	39.63	38.63	73.02	47.10
6	1156	1264	1206.1	2113	2196	2167.3	44.35	43.84	96.87	54.74
8	1119	1212	1165.9	1949	2078	1984.4	41.25	39.34	90.41	56.49

Table 4: Comparison between HSA and SA with different upper cell size limit (B_u)
$(M = 30,\ P = 30,\ W = 5,\ T_{pkmw} \sim \mathrm{DU}[5,20],\ \beta \sim \mathrm{DU}[1,4])$

B_u	HSA-$C_{\max}$			SA-$C_{\max}$			$\Delta C_{\max}$ (%)	HSA-CPU (s)	SA-CPU (s)	ΔCPU (%)
	MIN	MAX	AVE	MIN	MAX	AVE				
2	1263	1366	1312.7	2206	2355	2299.6	42.92	80.00	170.46	53.07
4	1250	1376	1296.8	2259	2428	2342.2	44.63	80.83	119.77	32.51
6	1084	1242	1161.7	2006	2101	2057.6	43.54	73.81	130.01	43.23
8	1086	1179	1134.1	1954	2091	2035.9	44.29	75.41	138.98	45.74

Table 5: Comparison between HSA and SA with different processing time (T_{pkmw})
($M = 20,\ P = 30,\ W = 5, B_u = 8,\ \beta \sim \mathrm{DU}[1,4]$)

T_{pkmw}	HSA-$C_{\max}$			SA-$C_{\max}$			$\Delta C_{\max}$ (%)	HSA-CPU (s)	SA-CPU (s)	ΔCPU (%)
	MIN	MAX	AVE	MIN	MAX	AVE				
DU[5, 10]	785	939	864.3	1624	1714	1686.6	48.75	71.43	155.02	53.92
DU[5, 20]	1132	1260	1180.9	2300	2447	2357.5	49.91	76.09	155.57	51.09
DU[5, 30]	1321	1492	1428.9	2616	2840	2723.6	47.54	71.66	129.59	44.70
DU[5, 40]	1520	1768	1679.6	3379	3632	3564.2	52.88	78.18	142.34	45.08

Table 6: Comparison between HSA and SA with different number of machines that can process the operation (β)
($M = 20,\ P = 30,\ W = 5,\ B_u = 8,\ T_{pkmw} \sim DU[5,40]$)

β	HSA-$C_{\max}$			SA-$C_{\max}$			$\Delta C_{\max}$ (%)	HSA-CPU (s)	SA-CPU (s)	ΔCPU (%)
	MIN	MAX	AVE	MIN	MAX	AVE				
DU[1, 5]	1494	1646	1565.1	3220	3309	3264.00	52.05	69.33	142.18	51.24
DU[1, 6]	1571	1873	1657.8	3391	3604	3479.8	52.36	74.02	150.46	50.80
DU[1, 7]	1434	1725	1590.5	3238	3432	3315.3	52.03	70.96	154.22	53.99
DU[1, 8]	1442	1622	1494.4	3270	3488	3383.6	55.83	66.48	135.37	50.89

References

1. Z. Zhang, Complexity of cellular manufacturing systems based on entropy models, *Journal of Mechanical Science and Technology*, 24 (11) (2010) 2275–2280.
2. V. R. Ghezavati, S. J. Sadjadi, M. Dehghan Nayeri, Integrating strategic and tactical decisions to robust deigning of cellular manufacturing under uncertainty: Fixed suppliers in supply chain, *International Journal of Computational Intelligence Systems*, 4 (5) (2011) 837–854.
3. I. Mahdavi, B. Javadi, K. Fallah-Alipour, J. Slomp, Designing a new mathematical model for cellular manufacturing system based on cell utilization, *Applied Mathematics and Computation*, 190 (1) (2007) 662–670.
4. R. Tavakkoli-Moghaddam, M. B. Aryanezhad, N. Safaei, A. Azaron, Solving a dynamic cell formation problem using metaheuristics, *Applied Mathematics and Computation*, 170 (2) (2005) 761–780.
5. J. Arkat, H. Abdollahzadeh, H. Ghahve, A new branch and bound algorithm for cell formation problem, *Applied Mathematical Modelling*, 36 (10) (2012) 5091–5100.
6. M. M. Paydar, I. Mahdavi, I. Sharafuddin, M. Solimanpur, Applying simulated annealing for designing cellular manufacturing systems using MDmTSP, *Computers & Industrial Engineering*, 59 (4) (2010) 929–936.
7. H. Nouri, T. S. Hong, A bacteria foraging algorithm based cell formation considering operation time, *Journal of Manufacturing Systems*, 31 (3) (2012) 326–336.
8. M. Hosseinabadi Farahani, L. Hosseini, An ant colony optimization approach for the machineCpart cell formation problem, *International Journal of Computational Intelligence Systems*, 4 (4) (2011)486–496.
9. P. Renna, M. Ambrico, Design and reconfiguration models for dynamic cellular manufacturing to handle market changes, *International Journal of Computer Integrated Manufacturing*, 28 (2) (2015) 170–186.
10. S. A. kioon, A. A. Bulgak, T. Bektas, Integrated cellular manufacturing systems design with production planning and dynamic system reconfiguration, *European Journal of Operational Research*, 192 (2) (2009) 414–428.
11. H. Rafiei, M. Rabbani, B. Nazaridoust, S. S. Ramiyani, Multi-objective cell formation problem considering work-in-process minimization, *The International Journal of Advanced Manufacturing Technology*, 76 (9–12) (2015) 1947–1955.
12. G. Egilmez, B. Erenay, G. A. Süer, Stochastic skill-based manpower allocation in a cellular manufacturing system, *Journal of Manufacturing Systems*, 33 (4) (2014) 578–588.
13. G. A. Süer, K. Kamat, E. Mese, J. Huang, Minimiz-

ing total tardiness subject to manpower restriction in labor-intensive manufacturing cells, *Mathematical and Computer Modelling*, 57 (3-4) (2013) 741–753.
14. M. Sakhaii, R. Tavakkoli-Moghaddam, M. Bagheri, B. Vatani, A robust optimization approach for an integrated dynamic cellular manufacturing system and production planning with unreliable machines, *Applied Mathematical Modelling*, 40 (2015) 169–191.
15. I. Mahdavi, A. Aalaei, M. M. Paydar, M. Solimanpur, Designing a mathematical model for dynamic cellular manufacturing systems considering production planning and worker assignment, *Computers and Mathematics with Applications*, 60 (4) (2010) 1014–1025.
16. B. Bootaki, I. Mahdavi, M. M. Paydar, New bi-objective robust design-based utilisation towards dynamic cell formation problem with fuzzy random demands, *International Journal of Computer Integrated Manufacturing*, 28 (6) (2015) 577–592.
17. C. Liu, J. Wang, J. Y.-T. Leung, K. Li, Solving cell formation and task scheduling in cellular manufacturing system by discrete bacteria foraging algorithm, *International Journal of Production Research*, 54 3 (2016) 923–944.
18. C. Liu, J. Wang, J. Y.-T. Leung, Worker assignment and production planning with learning and forgetting in manufacturing cells by hybrid bacteria foraging algorithm, *Computers & Industrial Engineering*, 96 (2016) 162–179.
19. M. T. Taghavifard, Scheduling cellular manufacturing systems using ACO and GA, *International Journal of Applied Metaheuristic Computing*, 3 (1) (2012) 48–64.
20. K. Halat, R. Bashirzadeh, Concurrent scheduling of manufacturing cells considering sequence-dependent family setup times and intercellular transportation times, *The International Journal of Advanced Manufacturing Technology*, 77 (9–12) (2015) 1907–1915.
21. M. Solimanpur, A. Elmi, A tabu search approach for group scheduling in buffer-constrained flow shop cells, *International Journal of Computer Integrated Manufacturing*, 24 (3) (2011) 257–268.
22. A. Elmi, M. Solimanpur, S. Topaloglu, A. Elmi, A simulated annealing algorithm for the job shop cell scheduling problem with intercellular moves and reentrant parts, *Computers & Industrial Engineering*, 61 (1) (2011) 171–178.
23. S. Kirkpatrick, C. D. Gelatt, M. P. Vecchi, Optimization by simulated annealing, *Science*, 220 (4598) (1983) 671–680.