International Journal of Industrial Engineering Computations 7 (2016) 83–110

Contents lists available at GrowingScience

International Journal of Industrial Engineering Computations

homepage: www.GrowingScience.com/ijiec

A new method for decreasing cell-load variation in dynamic cellular manufacturing systems

Aidin Delgoshaei*, Mohd Khairol Mohd Ariffin, Btht Hang Tuah Bin Baharudin and Zulkiflle Leman

Department of Mechanical and Manufacturing Engineering, University Putra Malaysia, 43400 UPM, Serdang, Kuala Lumpur, Malaysia

C H R O N I C L E

Article history:
Received April 22 2015
Received in Revised Format
July 23 2015
Accepted July 23 2015
Available online
July 30 2015

Keywords:
Facilities planning and design
Cell Scheduling
Cell Load Variation
Part Routing

A B S T R A C T

Cell load variation is considered a significant shortcoming in scheduling of cellular manufacturing systems. In this article, a new method is proposed for scheduling dynamic cellular manufacturing systems in the presence of bottleneck and parallel machines. The aim of this method is to control cell load variation during the process of determining the best trading off values between in-house manufacturing and outsourcing. A genetic algorithm (GA) is developed because of the high potential of trapping in the local optima, and results are compared with the results of LINGO® 12.0 software. The Taguchi method (an L_9 orthogonal optimization) is used to estimate parameters of GA in order to solve experiments derived from literature. An in-depth analysis is conducted on the results in consideration of various factors, and control charts are used on machine-load variation. Our findings indicate that the dynamic condition of product demands affects the routing of product parts and may induce machine-load variations that yield to cell-load diversity. An increase in product uncertainty level causes the loading level of each cell to vary, which in turn results in the development of “complex dummy sub-cells”. The effect of the complex sub-cells is measured using another mathematical index. The results showed that the proposed GA can provide solutions with limited cell-load variations.

© 2016 Growing Science Ltd. All rights reserved

1. Introduction

Facility design is a necessity in manufacturing systems engineering. An estimated $250 billion is spent annually for facility designing, planning, and re-planning in the U.S. (Tompkins et al., 2003). Of the total cost of manufacturing systems, 20% to 50% is attributed to material handling. This source also reports that effective planning can reduce such costs by 10% to 30%. Cellular manufacturing systems (CMS) are considered an effective way of using group technology. The manufacturing process is defined as a hybrid system of cells that apply the advantages of both jobbing (flexibility) and mass (efficient flow and high production rate) production approaches (Papaioannou & Wilson, 2010).

* Corresponding author. Tel/Fax: +(60) 1123091810
E-mail: delgoshaei.aidin@gmail.com (A. Delgoshaei)

© 2016 Growing Science Ltd. All rights reserved.

doi: 10.5267/j.ijiec.2015.7.004

Minimizing material transferring movements may be among the initial reasons for developing CMS. Although shifting from job-shop layouts to CMS can reduce material transferring movements (Agarwal & Sarkis, 1998; Durmusoglu & Nomak, 2005), many challenges are involved in designing or scheduling a system when the optimum part routings have yet to be identified. In this section, we will review some of the most important shortcomings and solutions that are proposed regarding material transferring during the design or scheduling of CMS.

The kinds of material movements in CMS are first described. In CMS studies, two main work-in-process (WIP) movements can be recognized (Fig. 1). Intra-cellular WIP transferring involves transferring materials among machines that are located in a cell. By contrast, in inter-cellular movements, materials are planned to shift between cells to perform some operations. Although intracellular WIP transferring may seem more desirable because of the lesser transfer cost involved than that for intercellular movements, intercellular WIP movements are unavoidable. In the following subsection, some of the most important reasons for the increase in material transferring costs and the solutions provided by scientists will be investigated in detail.

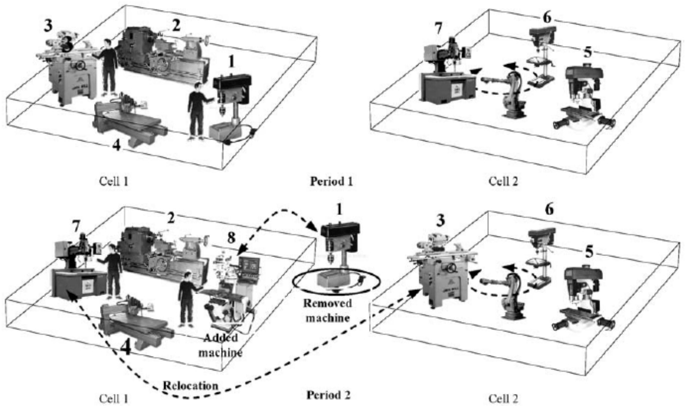


Fig. 1. Intracellular and Intercellular Material Transferring

Determining the best combination of machines that can be used in the consecutive operations of a part (during or after cell generation) is the aim of addressing part routing problems. From another perspective in part routing problems, each part can be completed in more than one way because of the existence of parallel machines. Choosing different permutations of various machines inevitably causes different inter and intracellular movements and entails material transferring costs accordingly. Gupta et al. (1995) focused on minimizing intercellular movements and maximizing machine utilization simultaneously during the process of grouping machines in cells. Gravel et al. (1998) employed a double-loop GA for simultaneous machine grouping and the part assigning problem in which an outer loop assigns machines to cells, and an inner loop determines the optimal part routing. Sofianopoulou (1999) proposed a mathematical method to allocate parallel machines in a multi-routing part allocation. Chen and Cao (2004) used quadratic terms for intercellular material handling to present an integrated model for production planning during cell forming. Yu and Sarker (2006) proposed a quadratic assignment problem model to minimize the total intercellular flows by considering bottleneck parts in which the output of other group formation methods can be used as an input of their method. Goncalves Filho and José Tiberti (2006) proposed a GA to minimize number of inter-cellular material transferring and cell load variations. Nsakanda et al. (2006) presented a multi-process plan problem to determine the best part routing of each process plan. They also used outsourcing as a useful strategy to meet the rest of the part demands. Afterward, Haleh et al. (2009) presented a hybrid Memetic algorithm and revised TOPSIS method to minimize cell load variation and inter-cellular WIP transferring. Safaei and Tavakkoli-Moghaddam (2009) focused on minimizing material transferring movements by considering the pre-determined batch sizes of materials whenever these are planned to be allocated to a cluster.

Existence of multi route plans causes emerging other problems. One of them is number of machine loads which can result machine load variation (Hsu & Su, 1998). Machine load variation has tremendous effects on the cellular systems and can cause the cell load variation which consequently can cause system imbalance (Haleh et al., 2009). In the mentioned condition, the number of machine loads for some machines increases which obviously can cause increasing the depreciation of some machines due to over allocating (Sullivan et al., 2002). Moreover, in such conditions, the possibility of machine broke increases due to pressure on over allocated machines. The queues of in-process parts behind over allocated machines increase while other parallel machines are remained idle (Seifoddini & Djassemi, 1993).

To overcome cell load variation, researchers offered different ways. Seifoddini and Djassemi (1996) used DPA for the re-routing of parts among machines to improve machine utilization. The results show that the proposed DPA improves system performance by reducing the congestion in machines with long waiting lines. Baker and Maropoulos (2000) attempted to minimize material transferring movements by finding the best position of cells and workstations within them in order to find the optimal flow of parts. By contrast, Solimanpur et al. (2004) addressed a QAP model to minimize intercellular WIP transferring in which cell locations were pre-defined. In some studies, scientists attempted to set the controller for the number of intercellular and intracellular part routes. Muruganandam et al. (2005) tried to control inter and intracellular moves by using weight points for these moves (0.3 and 0.7, respectively). This approach can minimize cell load variation to some extent. Gonçalves and Resende (2004) applied a hybrid local search and GA to minimize intercellular movements and maximize the utilization of machines within a cell. One reason for the emerging cell load variation is ignoring the capacity of machines during the scheduling process (Moussa & Kamel, 1998). Ahkioon et al. (2009) presented a CMS where maintenance activities can affect system capacity. To overcome such drawback, they also considered using outsource services and machine relocating. But their model was failed to solve large scale problems.

Boulif and Atif (2006) considered a situation in which cohabitation and non-cohabitation machines exist. In actual practice, such a condition may indeed happen because of safety reasons. They proposed a new approach to minimize WIP transferring between pairwise machines while the maximum number of cells is considered aside from the cell sizes. Ariafar and Ismail (2009) addressed the problem of material transferring while the shape of the machines was assumed equal. A center-based distance calculation between machines was employed.

Aside from the strategies and solutions discussed above, during the last two decades, the idea of reconfiguring cells by shifting of machines or re-arranging of part families to smooth material transferring within and between cells (or as a result of changing part demands during periods) has become noteworthy. Almost in all cases, the focus is on the modification of cell layouts during the manufacturing process to determine the best part routing. Safaei et al. (2008) investigated the issue of machine time capacity and maximum cell size in a reconfigurable dynamic CMS in which machine relocating was allowed. A few years later, Rafiee et al. (2011) developed a similar mathematical programing method for a reconfigurable manufacturing system in which more aspects of a real system, such as preventive and maintenance activities, finished and unfinished parts inventory, and defective parts replacement costs, were considered. Some studies also investigated other situations in the part routing problem. Xambre and Vilarinho (2003) examined a special condition in which a percentage of each operation can be completed by one machine, and the rest can be rescheduled for another parallel machine. Yu and Sarker (2003) tried to minimize material transforming costs by finding better cell locations by relocating cells. Elmi et al. (2011) developed a situation that was previously proposed by Won and Currie (2007) in which some parts need to visit a machine more than once in a non-consecutive manner (re-entrant parts). In the proposed model, they presented a new method to schedule both bottleneck and re-entrant parts in cells. Paydar et al. (2010) considered operation sequences in a multiple travelling salesman formulation for the cell forming and machine locating problem, with multiple departures and a single destination considered. In most real cases, part demands are different from one planning horizon to another. Such a criterion is known as dynamic part demand. Market changes, changes in product designs, and the manufacture of new products are some of the reasons for the change in part demands through different time periods. These conditions may cause emerging imbalances in part routings and bottleneck machines (Wang et al., 2001). Defersha and Chen (2006) used parallel machines and outsource services to overcome dynamic part demand defects in cell forming process. Jeon and Leep (2006) presented a model for scheduling dynamic cells where machine failures can cause waiting times and reduce system capacity accordingly. Tavakkoli-Moghaddam, Aryanezhad, et al. (2007) considered dynamic part demands and parts mixed for a reconfigurable part routing problem. Tavakkoli-Moghaddam et al. (2007) considered the normal distribution function to estimate the part demands in a stochastic model; minimizing material transferring movements was the main objective of the method. Egilmez et al. (2012) focused on uncertain operation times in D-CMS. The contribution of their model is considering risk level in process of designing cells in dynamic environment. A few years later, Egilmez and Süer (2014) evaluated the impact of risk level in an integrated cell forming and scheduling problem using Monte Carlo Simulation. Ariafar et al. (2014) focused on the impact of dynamic product demand on facility layout problem. The main objective of the proposed model was minimizing material transferring by arranging the machine cells within the shop-floor, and the machines within each of the machine cells. Afterward, Renna and Ambrico (2015) also proposed three models for designing, reconfiguring and scheduling cells in dynamic condition of product demands. In their models, they considered minimizing system costs including intercellular movements, machining and reconfiguring costs as well as maximizing net-profit.

An intensive review of literature reveals that problem scheduling dynamic cellular manufacturing systems (D-CMS) to minimize the cell load variation by finding the best trading off values between the values of in-house manufacturing and of outsourcing when the capacity of machines are taken into account and backorder of products between planning periods is restricted, is less developed.
Therefore, the contribution of the this research can be summarized as investigating the impact of product demand uncertainty on cell load variation and proposing a way to smooth the machine loads which can cause cell load variations.

2. Research methodology

Both inter- and intra-cellular movements are considered in the current study; hence, the model should be sensitive to material transfer within cells according to the product sequence, function, and location of the machines. To explain work-in-progress (WIP) transfer, two consecutive processes labeled as s and $s+1$ are assumed. These processes may cause inter- or intra-cellular movement only if $\sum_{k=1}^{K}\sum_{j=1}^{M} Y_{s+1,i,j,k,t} \geq 1$ (Y is a binary value of 1 if operation $S+1$ of part i is performed using machine j, which is located in cell k during period t). The authors aim to minimize setup costs as a part of the objective functions in the proposed model; hence, the equation was modified to $\sum_{k=1}^{K}\sum_{j=1}^{M} Y_{s+1,i,j,k,t} = 1$, thus ensuring that operation $s+1$ is performed using only one of the machines in the system. As a consequence of eliminating the operations that were planned for service using the same multifunctional machine, equation $Y_{s,i,j,k,t}.\left(\left(\sum_{j=1}^{M} Y_{s+1,i,j,k,t}.X_{s+1,j,k,t}\right) - X_{s+1,i,j,k,t}\right)$ shows the amount of WIP transferred from current machine j to another machine $\sim j$ to serve subsequent operation $s+1$ (Fig. 2). The transfer of all WIP inside the cells for all of the parts operations can be formulated as:

$$\sum_{t=1}^{TH}\sum_{k=1}^{K}\sum_{i=1}^{N}\sum_{s=1}^{S-1}\sum_{j=1}^{M}\left(Y_{s,i,j,k,t}.\left(\left(\sum_{j=1}^{M} Y_{s+1,i,j,k,t}.X_{s+1,j,k,t}\right) - X_{s+1,i,j,k,t}\right)\right) \quad (1)$$

Given the same logic, inter-cellular movements can be detected by determining the consecutive operations that are performed in two different cells (that is, k and $\sim k$) regardless of whether the same type of machine (j) or other machine ($\sim j$) is employed $\left[Y_{s,i,j,k,t}.\left(\left(\sum_{k=1}^{K}\sum_{j=1}^{M} Y_{s+1,i,j,k,t}.X_{s+1,j,k,t}\right)\right)\right]$. Excluding intra-cellular movements from total movement reveals inter-cellular movements (Fig. 2):

$$\left[Y_{s,i,j,k,t}.\left(\left(\sum_{k=1}^{K}\sum_{j=1}^{M} Y_{s+1,i,j,k,t}.X_{s+1,j,k,t}\right)-\sum_{j=1}^{M} X_{s+1,i,j,k,t}\right)\right]. \tag{2}$$

Hence the total amount of inter-cellular transfer for all parts operations in all cells throughout the planning horizon can be formulated as:

$$\sum_{t=1}^{TH}\sum_{k=1}^{K}\sum_{i=1}^{N}\sum_{j=1}^{M}\sum_{s=1}^{S-1}\left(Y_{s,i,j,k,t}.\left(\left(\sum_{k=1}^{K}\sum_{j=1}^{M} Y_{s+1,i,j,k,t}.X_{s+1,j,k,t}\right)-\sum_{j=1}^{M} X_{s+1,i,j,k,t}\right)\right) \tag{2}$$

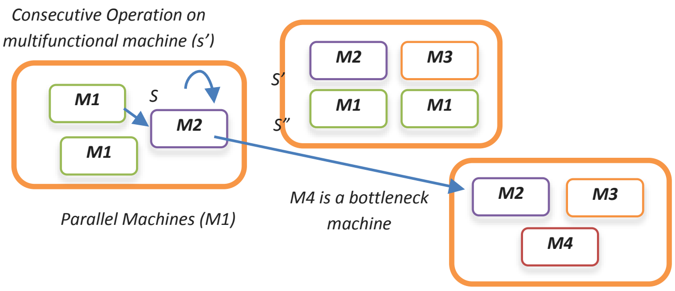

Consecutive Operation on multifunctional machine (s')
M1
S
M2
M1
Parallel Machines (M1)
S'
S''
M2
M3
M1
M1
M4 is a bottleneck machine
M2
M3
M4

Fig. 2. Graphical scheme of inter and intra-cellular movements

3. Problem Formulation

A nonlinear mixed integer programming (NL-MIP) model is presented in this section to determine the ideal tradeoff values between in-house manufacturing and outsourcing over the planning horizon under the condition that backorders are limited and that all product demands are uncertain. These demands may vary from time to time. In formulating the model, all system costs are accounted for, including group setup, operating, machine purchasing, outsourcing, and backorder costs. The goal is to survey production specifications under part demand uncertainty and to demonstrate how system characteristics can influence system performance.

3.1. Assumptions

1) Product types are associated with a number of operations that must be performed according to manufacturing priorities.
2) The lower and upper bounds for each cell are determined in advance.
3) Machine purchasing and relocation are allowed.
4) Product demands in each scheduling period are not fixed and are expressed by a normal distribution function.
5) Machine capacities must be considered in scheduling.
6) Subcontractor services are permitted for some products; however, the capacity of subcontractors is restricted.
7) Backorders are allowed but restricted. The beginning inventory is zero and last-period backorders are not allowed.

3.2 Inputs

i: $number\ of\ products$
m: $number\ of\ operations$
j: $type\ of\ machines$
f: $number\ of\ machines$
k: $number\ of\ cells$
l: $number\ of\ sub-contractors$
t: $planning\ periods\ (time\ slots)$

3.3 Parameters

$d_{i,t}$: demand of part i in period t

$$d_{i,t} \sim N(\mu_{i,t}, \sigma_{i,t}) \tag{3}$$

$S_{\ j}$: Setup cost of machine j
$OP_{\ i,m,j}$: cost of processing part m of product i using machine j
$OS_{\ l}$: cost of performing one product by subcontractor l
BC_i: backorer cost of product i for a period
α_i: intracellular movement cost of part i
β_i: interrcellular movement cost of part i
${K_j}^+$: purchasing cost of mchine j
${K_j}^-$: selling cost of mchine j
$length$: is a lenth of the cell
$width$: is a width of the cells
Cl: is cell size and refers to maximum number of machines can be located in a cell

$$CL = length * width \tag{4}$$

3.4 Input Matrixes

Product demand ($D_{i,t}\ [\ddots]_{i,t}$)
Batch size ($BS_i\ [\ddots]_i$)
Machine Component Incidence Matrix ($MCIM_{i,j}\ [\ddots]_{i,j}$)
Machine capacity ($MA_j\ [\ddots]_j$)
Sub-contractor capability ($SC_l\ [\ddots]_l$)
Allowed backorder ($AB_{i,t}\ [\ddots]_{i,t}$)
Initial number of machines ($NOM_j\ [\ddots]_j$**)**
Operation Cost ($OP_{i,m,j}\ [\ddots]_{i,m,j}$)
Setup Cost ($S_j\ [\ddots]_j$)
Intercellular Cost ($\alpha_i\ [\ddots]_i$)
Intracellular Cost ($\beta_i\ [\ddots]_i$)
Machine purchasing Cost ($K_j\ [\ddots]_j$)
Backorder Cost ($BC_i\ [\ddots]_i$)
Outsourcing Cost ($OS_l\ [\ddots]_l$)

3.5 Variables

$X_{m,i,f,j,k,t}$ number of part m of product *i* which is performing by fth of machine type *j* in cell *k* in period *t* (int.)
$Z_{m,i,f,j,k,t}$ if part m of product *i* performs using fth of machine type *j* in cell *k* in period *t* (bin.)
$Y_{l,t}$ number of products which manufactured by sub-contractor l in period t (int.)
$B_{i,t}$ number of product *i* which is decided to postpone for next period (int.)
${N^+}_{j,k,t}$ number of machine type *j* added to cell *k* during period t (int.)
${N^-}_{j,k,t}$ number of machine type *j* removed from cell *k* during period t (int.)

3.6 Mathematical Model

$$\sum_{t}^{T}\sum_{k}^{K}\sum_{j}^{J}\sum_{f}^{F}\sum_{i}^{I}\sum_{m}^{M} S_j . Z_{m,i,f,j,k,t} . (X_{m,i,f,j,k,t}/BS_i) + \sum_{t}^{T}\sum_{k}^{K}\sum_{j}^{J}\sum_{f}^{F}\sum_{i}^{I}\sum_{m}^{M} MCIM_{i,j} . OP_{\ i,m,j} . X_{m,i,f,j,k,t} . Z_{m,i,f,j,k,t} \tag{5}$$

$$+ \sum_{t}^{T}\sum_{k}^{K}\sum_{j}^{J} [(K_j{}^{+} . N_{j,k,t}^{+}) - (K_j{}^{-} . N_{j,k,t}^{-})] \tag{5}$$

$$+ \sum_{t}^{T}\sum_{l}^{L} OS_l . Y_{l,t} + \sum_{t}^{T}\sum_{i}^{I} B_{i,t} . BC_i \tag{6}$$

$$+ \sum_{t=1}^{TH}\sum_{k=1}^{K}\sum_{i=1}^{I}\sum_{m=1}^{m-1}\sum_{j}^{J}\sum_{f=1}^{F} \left(\gamma_i . Z_{m,i,f,j,k,t} . \left(\left(\sum_{j=1}^{J} Z_{m+1,i,f,j,k,t} . X_{m+1,i,f,j,k,t} \right) - X_{m+1,i,f,j,k,t} \right) \right) \tag{7}$$

$$+ \sum_{t=1}^{TH}\sum_{k=1}^{K}\sum_{i=1}^{I}\sum_{m=1}^{m-1}\sum_{j=1}^{J}\sum_{f}^{F} \left(\xi_i . Z_{m,i,f,j,k,t} . \left(\left(\sum_{k=1}^{K}\sum_{j=1}^{M} Z_{m+1,i,f,j,k,t} . X_{m+1,i,f,j,k,t} \right) - \sum_{j=1}^{M} X_{m+1,i,f,j,k,t} \right) \right) \tag{8}$$

subject to

$$\sum_{k}^{K}\sum_{j}^{J}\sum_{f}^{F}\sum_{i}^{I}\sum_{m}^{M} \left[\left({}^{X_{m,i,f,j,k,t}}/_{M} \right) \right] + \sum_{l}^{L} Y_{l,t} + \sum_{l}^{I} B_{i,t} \geq d_{i,t} \ \ \forall\, t, i; \tag{9}$$

$$X_{m,i,f,j,k,t} . \left(1 - MCIM_{i,j}\right) \leq 0 \ \ \forall\, t, k, j, i, m; \tag{10}$$

$$X_{m,i,f,j,k,t} \leq MA_j \ \ \forall\, t, k, j, i, m; \tag{11}$$

$$\sum_{l}^{L} Y_{l,t} \leq SC_l \ \ \forall\, l, t; \tag{12}$$

$$\sum_{i}^{I} B_{i,t} \leq AB_{i,t} \ \ \forall\, t, i; \tag{13}$$

$$\sum_{j}^{J} N_{j,k,t}^{+} - \sum_{j}^{J} N_{j,k,t}^{+} + N_{k,j,t-1} = N_{k,j,t} \ \forall\, t > 1, k; \tag{14}$$

$$N_{j,k,1} = NOM_j \ \ \forall\, k, j; \tag{15}$$

$$LP \leq \textstyle\sum_{j}^{J} N_{j,k,t} \leq CL \ \ \forall\, k, t; \tag{16}$$

$$\sum_{k}^{K}\sum_{f}^{F}\sum_{i}^{I}\sum_{m}^{M} X_{m,i,f,j,k,t} \leq MA_j . \sum_{k}^{K} N_{k,j,t} \ \ \forall\, t, j; \tag{17}$$

$$X_{m,i,f,j,k,t};\ Y_{l,t},;B_{i,t};\ N_{j,k,t}^{+};\ N_{j,k,t}^{-}: integer \tag{18}$$

$$Z_{m,i,f,j,k,t} : binary \tag{19}$$

The first term of the objective function represents the setup cost for each machine that may vary from period to period. The following term denotes the operating cost, including the machinery of each part. The third term corresponds to the purchasing or removing costs of machines. The fourth sentence indicates the outsourcing cost, and the fifth sentence represents backorder costs. The sixth and seventh denote the intra- and inter-cellular material transfer costs, respectively. As observed, all system costs are affected by inflation rate in the production horizon. This condition allows decision-makers to evaluate the system cost entropy caused by different cost uncertainty factors (such as inflation or market change rate) in planning horizon.

The first set of constraints guarantees that the demands for each part can be satisfied by in-house manufacturing through outsource services or postponement to the next period. The second constraint is developed to ensure that parts operations are based on MCIM information. The third constraint guarantees that each machine is allocated on the basis of capacity. Similarly, the fourth constraint assures that sub-contractor services do not exceed the declared capacity. The fifth constraint controls the values of backorders depending on the backorder limits. The sixth set of constraints ensures that the amount of purchased or removed machines remains logical in the manufacturing horizon. The seventh set of constraints represents the initial set of each machine type in the first period. The eighth set of constraints controls the size of cells during the manufacturing process, and the ninth set of operations indicates that the amount of parts produced should not be more than the available amount of that machine type. The next two sets of constraints are used to control the domain of the variables.

4. Solution procedure

Tavakkoli-Moghaddam et al. (2010) reported that CMS scheduling models are non-polynomial problems that are difficult to solve using normal optimizers. On the basis of the review of existing literature, heuristics and metaheuristics have been widely employed by scientists to determine optimum or near-optimum solutions. Genetic Algorithm has been successfully used in similar researches in CMS and therefore can be considered as a trustful method for solving the proposed model (Banerjee & Das, 2012; Dimopoulos & Mort, 2001; Goncalves Filho & José Tiberti, 2006; Paydar et al., 2013; Rogers & Kulkarni, 2005). The risk of trapping in the local optima is high because of the complexity of the proposed model. Hence, in the first step, the authors solved the proposed model for number of small, medium and large scale experiments using LINGO 12.0 and a genetic algorithm (GA). These experiments are selected in line with central limit theorem in statistics; thus, researchers can make confident decisions. The results presented in Table 14 show that although LINGO 12.0 and GA reported similar outcomes for small-scale experiments, the proposed GA can provide improved solutions for all medium- and large-scale experiments. Hence, the next section discusses the use of GA to solve all experiments.

4.1 Genetic Algorithm Design for Solving the Proposed Model

GA was first developed by (Holland, 1975) on the basis of the biological process of genetic inheritance. In this process, population characteristics are improved over generations. The basic concept of GA is to simulate the processes necessary for evolution in a natural system, specifically those that follow the principles first laid down by Charles Darwin's [1] theory of evolution that emphasizes the survival of the fittest.

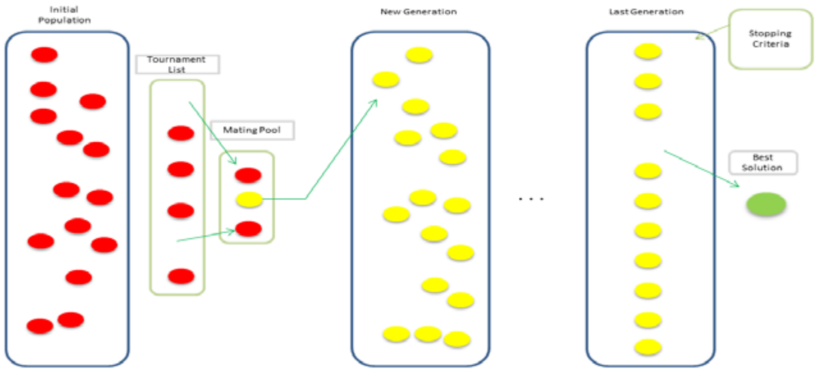


Fig. 3. A scheme of GA performance for improving the characteristics of population members over generations

[1] . (1809-1882 AC)

The favorable characteristics of the members are expected to be retained over generations to optimize a pre-determined fitness function (Fig. 3). GA has been widely used to solve optimization problems in CMS literature.

4.2 Mechanism of the Proposed Genetic Algorithm

The proposed GA is developed to use the best founded part routes of the parts (as chromosomes of each parent) and combine them to generate a new solution string with better quality. For this purpose, a layout is initially presented in the first generation according to a similarity index based on the goal of determining the maximum similarity between the products and required machines. For the other generations, the algorithm calls for a main string and a TOther string (as explained in subsequent sections). The chromosomes of these generations are combined to generate a layout that is supposed to improve total system costs (Fig. 4). The algorithm checks whether or not outsourcing is more economic than in-housing. If outsourcing is cheaper, then the product demands are to be completed by subcontractors in consideration of their capacity.

Table 1
Pseudo code for the proposed Genetic Algorithm

```
Insert Dataset
    • Input data sets ($D_{i,t}$; $BS_i$; $MCIM_{i,j}$; $MA_j$; $SC_l$; $AB_{i,t}$; $NOM_j$; $OP_i$; $S_j$; $\alpha_i$; $\beta_i$; $OC_j$; $SR_n$; $TB_{j,t}$; $CC_n$; $H_j$; $F_j$; $CS_j$; $W_i$ ; $K_j$; $DR_j$; $PL_{j,t}$; $PST_j$ ; $EMT_j$; $FR_j$; $PMC_j$; $ESC_j$; $BC_i$; $OS_l$; $Ir_t$)
    • Initialize pop.size, k, G, Mu & LEP (pop.size: population size; k= neighbourhood radius; Mu: mutation rate; G: number of generations; LEP=local escape probability)
While n<G
for m=1: pop.size
Step 1)
    •   if n=1
    •   Generate a layout
    •   else
Step 2)
    •   Choose $X^{best}$ as main string
    •   Choose a member from tournament list (tother string)
    •   find $X_{ij}$=0   (find those machines that are not assigned)
    •   $C_{ij} = 0$     (remove idle machines from the main string)
    •   Calculate Slack list (list of idle machines)
    •   Recall the $Candidate.list(X^{best})$
            o   for $p \in Candidate.list({X_k}^{itr})$ (Re-locate the machines in slack list to make best part routes of tother string in the main string)
            o   if $C_{ij}$=0
                ▪   $C_{ij}$=$M_j$  ( $j \in$ Slack list)
Step 2)
    •   $Y: Rand(1)$
    •   if $Y \leq Mu$
            o   A=Rand(1)*width
            o   B=Rand(1)*width
            o   Replace Row A and B in layout of X
Step 3)
    •   for i ∈ $I$
    •   if $OS_l$< $OP_i$
    •   $O_i$=min($D_{i,t}$;Out source capacity l)
    •   calculate Remained $D_{i,t}$
Step 4)
    •   find $MCIM_{i,j}$
    •   Generate  Neighbor list considering neighbor radius ($n^m$)
            o   $for\ k \in n$
                    ▪   $Part.route_k$= $\sum|intercellular\ travel| + |intracellular\ travel|$ (for consecutive machines)
    •   Find the Best Part routes in the X
    •   $x_i$= min($MA_j$, Remained $D_{i,t}$, $BS_i$ )
    •   calculate Remained $D_{i,t}$
Step 5)
    •   for i ∈ $I$
    •   if Remained $D_{i,t}$>0
    •   if Machine Purchasing < Outsourcing or $OS_l = 0$
    •   $NOM_j = NOM_j + 1\ for\ needed\ machine$
    •   Go step 3
Step 6)
    •   else
    •   Set $b_i = \text{Remained}\ D_{i,t}$
Step 7)
    •   Calculate $OFV_i$
    •   if $OFV_i \leq \min_{itr}(OFV)$
            o   $Tournament\ list_i$=$X_i$
            o   $OFV^{best}$ =$OFV_i$ & $X^{best} = {X_k}^{itr}$
            o   save $Candidate.list({X_k}^{itr})$
Step 8)
    •   else
    •   R=rand(1)
    •   if R≤ $LEP$   $then$   $Tournament\ list_i$=$X_i$
Step 9)
    •   Check Stopping Criteria
```

For the remaining product demands (if any), the GA identifies the best part routes according to the minimum distance between the machines and assigns parts to the machines in part routes given the remaining product demands, machine capacity and batch size.

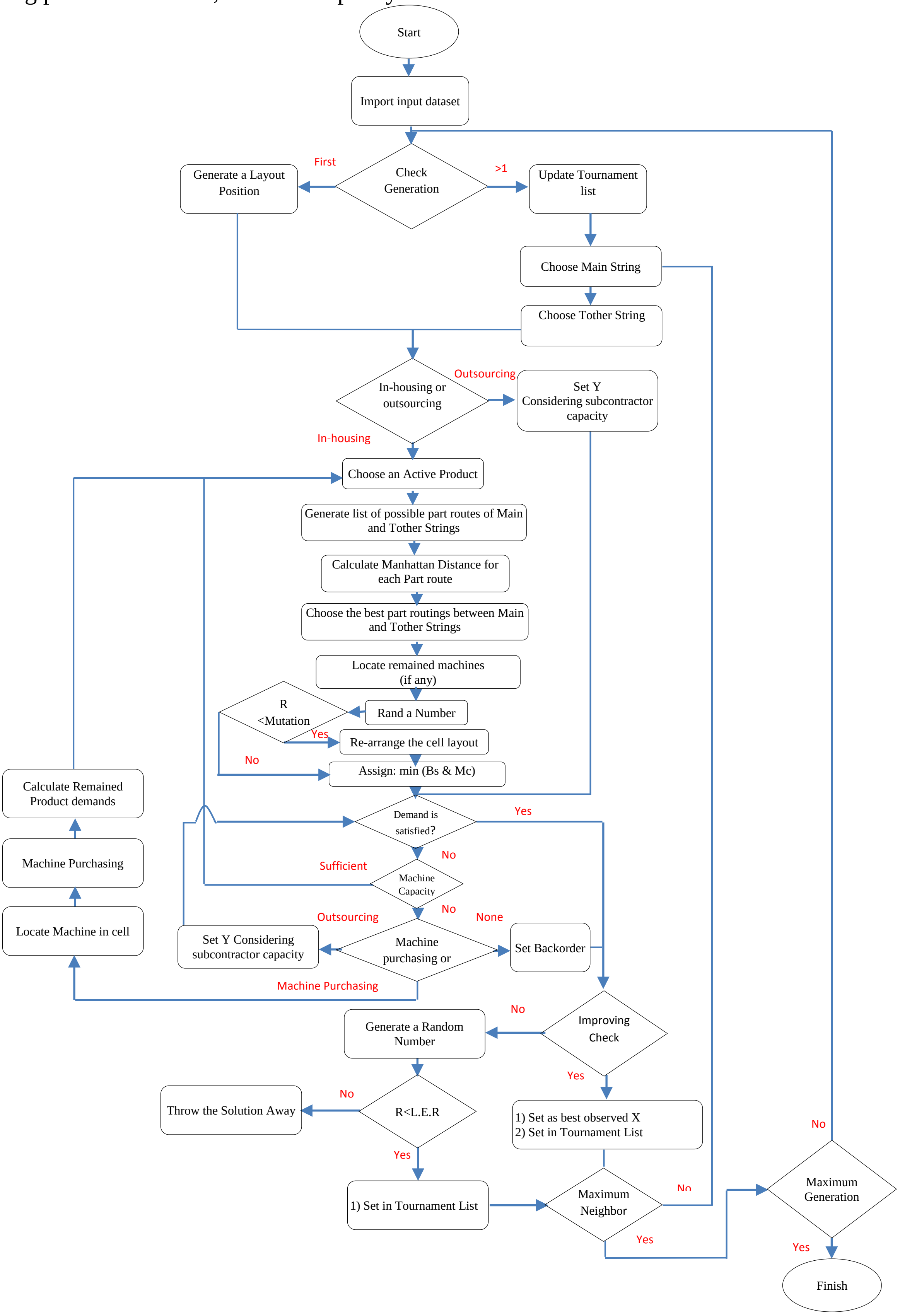


Fig. 4. Flowchart of the proposed genetic algorithm for solving the proposed model

Part routing is repeated until the product demands are fulfilled or when machine capacity in the cells is exhausted. The GA then checks if the purchase of needed machines for manufacturing can help meet product demands. If neither in-house manufacturing nor subcontractor services is feasible, then the remaining product demands are considered to be product backorders (in consideration of the allowed backorder limit). In rare cases, the generated layout is re-arranged by a mutation operator to avoid local optima traps. The optimizing process is repeated until all population members in a generation are fully developed. A tournament list containing elite members is then assembled for future generations. Table 1 shows the pseudo code of the proposed GA.

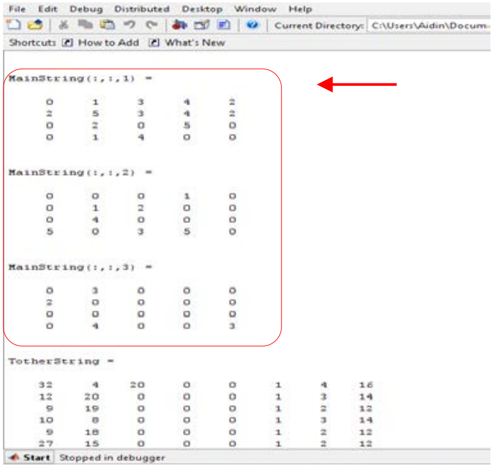
4.3 Parameters, Settings, and Functions

Population Size and Number of Generations

The number of population members in each generation is an important issue in GA. The exact population size cannot be determined; this process depends on the size and complexity of the mathematical models. In this research, we considered three levels (100, 200, and 300 for small-, medium-, and large-scale problems, respectively) for population size and number of generations; however, the ideal set of GA input parameters (generation number, population size, and mutation rate) are determined as well using the Taguchi method presented in Section 4.4 for enhanced accuracy.

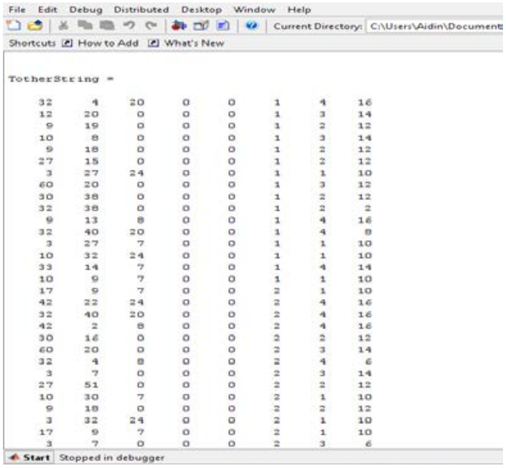
Selection of the Operator and Establishment of the Tournament List

Each member of a generation must carry better original genes than the previous generation did. In the proposed GA, a tournament list is created for each generation that carries the ideal layouts of the previous generation to increase the chances of selecting an improved objective function. The ideal observed layout is considered the base member in the mating bath (main string), and a member of the tournament list is chosen as the other candidate of mating bath (TOther string). The aim of establishing this list is to use the chromosomes (good routes) of the main and TOther strings to improve a layout. Fig. 5 shows the process of selecting the main string in the developed GA. Similarly, Fig. 6 shows the procedure of using the best parts routes of an invited member from the tournament list to improve the part routing of the ideal observed solution thus far.



```
MainString(:,:,1) =

     0     1     3     4     2
     2     5     3     4     2
     0     2     0     5     0
     0     1     4     0     0


MainString(:,:,2) =

     0     0     0     1     0
     0     1     2     0     0
     0     4     0     0     0
     5     0     3     5     0


MainString(:,:,3) =

     0     3     0     0     0
     2     0     0     0     0
     0     0     0     0     0
     0     4     0     0     3


TotherString =

    32     4    20     0     0     1     4    16
    12    20     0     0     0     1     3    14
     9    19     0     0     0     1     2    12
    10     8     0     0     0     1     3    14
     9    18     0     0     0     1     2    12
    27    15     0     0     0     1     2    12
```

Fig. 5. Sample of using best observed layout so far as the main string (right image)



```
TotherString =

    32     4    20     0     0     1     4    16
    12    20     0     0     0     1     3    14
     9    19     0     0     0     1     2    12
    10     8     0     0     0     1     3    14
     9    18     0     0     0     1     2    12
    27    15     0     0     0     1     2    12
     3    27    24     0     0     1     1    10
    60    20     0     0     0     1     3    12
    30    38     0     0     0     1     2    12
    32    38     0     0     0     1     2     2
     9    13     8     0     0     1     4    16
    32    40    20     0     0     1     4     8
     3    27     7     0     0     1     1    10
    10    32    24     0     0     1     1    10
    33    14     7     0     0     1     4    14
    10     9     7     0     0     1     1    10
    17     9     7     0     0     2     1    10
    42    22    24     0     0     2     4    16
    32    40    20     0     0     2     4    16
    42     2     8     0     0     2     4    16
    30    16     0     0     0     2     2    12
    60    20     0     0     0     2     3    14
    32     4     8     0     0     2     4     6
     3     7     0     0     0     2     3    14
    27    51     0     0     0     2     2    12
    10    30     7     0     0     2     1    10
     9    18     0     0     0     2     2    12
     3    32    24     0     0     2     1    10
    17     9     7     0     0     2     1    10
     3     7     0     0     0     2     3     6
```

Fig. 6. Result of Using the best Part routes of an Invited Member from the Tournament List as Tother String (right image)

Mating Pool

Upon selecting a member from the tournament list (TOther string) and determining the best observed layout (main string) through a selecting operator, the chromosomes of candidates are rearranged to generate a new population member that carries the good characteristics of both parents. For this purpose, two main operators are developed: a crossover operator that basically rearranges the selected operators and a mutation operator that is rarely used but is important in avoiding local optima.

Crossover Operator

The main genetic operator is a crossover operator that chooses and combines pieces of information from different individuals in the population. The main purpose of employing this operator in the proposed GA is to facilitate the use of the elite parts routes in the ideal layout observed in the main string layout. Hence the most appropriate crossover operator for the proposed GA is the arithmetic operator. In this operator, arithmetic functions are defined and are applied to rearrange specific parts of the main string with a TOther string to produce new offspring.

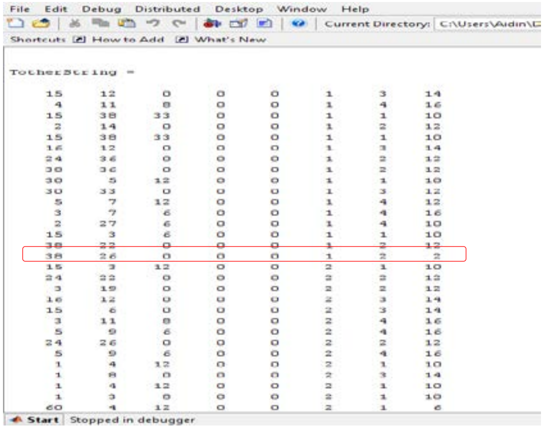
a) Suppose $cell\ lyout_k^m \begin{bmatrix} P_{1,1} & P_{1,2} & \cdots & P_{1,i} \\ \ldots. & \cdots & \cdots & \ldots \\ P_{j,1} & P_{j,2} & \cdots & P_{j,i} \end{bmatrix}$ is the layout of the best observed solution so far.

b) Suppose $M'(m'_1, m'_2, \ldots, m'_k; x'_1, x'_2, \ldots, x'_k)$ is a part route of the tother string which must be replaced by $M(m_1, m_2, \ldots, m_k; x_1, x_2, \ldots, x_k)$ in the $cell\ lyout_k^m$.

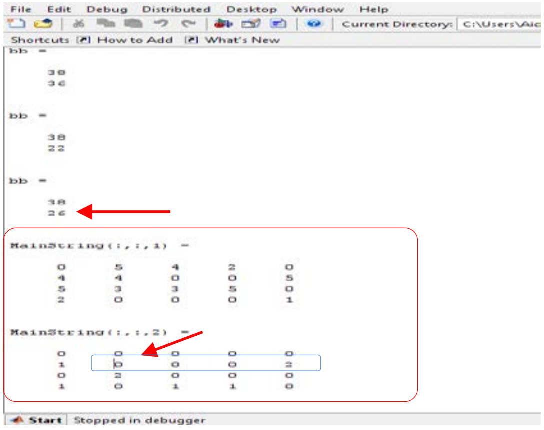
c) If $C_{m'} < C_{m'}$ (20)

d) Then $OFV_{m'} < OFV_{m'}$ (21)

where $C_{m'}$ refers to the distance cost of parts route m' in the TOther string. $OFV_m\ and OFV_{m'}$ represent the objective function value of the main string before and after replacement, respectively. In the left-hand side image of Fig. 8, a certain parts route is checked in the main string. If this route is available in the main string (for example, bb= [38 22] in the right-hand side image of Fig. 8), the GA determines another favorable parts route; however, if the mentioned parts route is not available in the main string (bb = [38 26]), then the needed machines are placed into the main string to generate the parts route (with respect to the total number of available machines).


File Edit Debug Distributed Desktop Window Help
Current Directory: C:\Users\Aidin\D
Shortcuts How to Add What's New
Start
Stopped in debugger

```
TotherString =

    15    12     0     0     0     1     3    14
     4    11     8     0     0     1     4    16
    15    38    33     0     0     1     1    10
     2    14     0     0     0     1     2    12
    15    38    33     0     0     1     1    10
    16    12     0     0     0     1     3    14
    24    36     0     0     0     1     2    12
    30    36     0     0     0     1     2    12
    30     5    12     0     0     1     1    10
    30    33     0     0     0     1     3    12
     5     7    12     0     0     1     4    12
     3     7     6     0     0     1     4    16
     2    27     6     0     0     1     4    10
    15     3     6     0     0     1     1    10
    38    22     0     0     0     1     2    12
    38    26     0     0     0     1     2     2
    15     3    12     0     0     2     1    10
    24    22     0     0     0     2     2    12
     3    19     0     0     0     2     2    12
    16    12     0     0     0     2     3    14
    15     6     0     0     0     2     3    14
     3    11     8     0     0     2     4    16
     5     9     6     0     0     2     4    16
    24    26     0     0     0     2     2    12
     5     9     6     0     0     2     4    16
     1     4    12     0     0     2     1    10
     1     8     0     0     0     2     3    14
     1     4    12     0     0     2     1    10
     1     3     8     0     0     2     1    10
    60     4    12     0     0     2     1     6
```


File Edit Debug Distributed Desktop Window Help
Current Directory: C:\Users\Aid
Shortcuts How to Add What's New
Start
Stopped in debugger

```
bb =

    38
    36


bb =

    38
    22


bb =

    38
    26

MainString(:,:,1) =

     0     5     4     2     0
     4     4     0     0     5
     5     3     3     5     0
     2     0     0     0     1


MainString(:,:,2) =

     0     0     0     0     0
     1     0     0     0     2
     0     2     0     0     0
     1     0     1     1     0
```

Fig. 7. Using Crossover Operator to choose a part route that still has capacity (left image) and replace it in main string (right image).

Using Control Limits during Part Routing Process

To examine machine load variations a formula is developed on the basis of the machine load variations within a cell:

$$\overline{X_\iota} \pm k{S_i}^2, \quad \forall\, i \tag{22}$$

where $\overline{X_\iota}$ is the average of a specific machine-type load in a cell in a planning horizon. ${S_i}^2$ shows the standard deviation that can be calculated based on the observed values for X_i. K is a value between 1 and 3 that can be set by decision makers depending on the dimension of the experiment and the ranges of X_i in each case.

The formula above determines the machines that allocate more than the average numbers of parallel machines in a cell and if so avoids them to consider as a member of a part route. The number of loads depends on many factors, such as remaining product demand and capacity. Machines must be separated at a safe distance, as indicated by the symbol $k{S_i}^2$. Each machine that exceeds the safety distance calculated by Equation 22 should not be considered as a candidate for next part routings.

Fitness Function

For this research, the objective function of each proposed model is applied as a fitness function operator in the proposed mathematical models:

$$\begin{aligned} &\sum Group\ Setup\ Cost &&+ \sum Operating\ Cost + \\ &\sum Outsourcing\ Cost &&+ \sum Backorder\ Cost + \\ &\sum Machine\ Purchasing\ Cost &&+ \sum Intercellular\ Transferring\ Cost + \\ &\sum Intracellular\ Transferring\ Costs && \end{aligned} \tag{23}$$

Each solution is evaluated to determine whether or not it can improve the minimum total system costs thus far. If so, then the GA considers the solution in question to be a member of the tournament list.

a) Suppose $X_k^{itr}(x_1, x_2, \dots, x_i; o_1, o_2, \dots, o_l; b_1, b_2, \dots, b_i)$ is the k^{th} solution in itr^{th} iteration.

b) If $F\left(X_k^{itr}\right) \le min\left(F\left(X_1^{itr}\right), F\left(X_2^{itr}\right), \dots, F\left(X_{k-1}^{itr}\right); F^{best}\left(X_k^{itr-1}\right)\right)$; $\forall\, k \in i$ (24)

c) Then, $Tournament.\, list_{m+1} = X_k^{itr}$ (suppose the tournament list already has m members) (25)

Mutation Operator

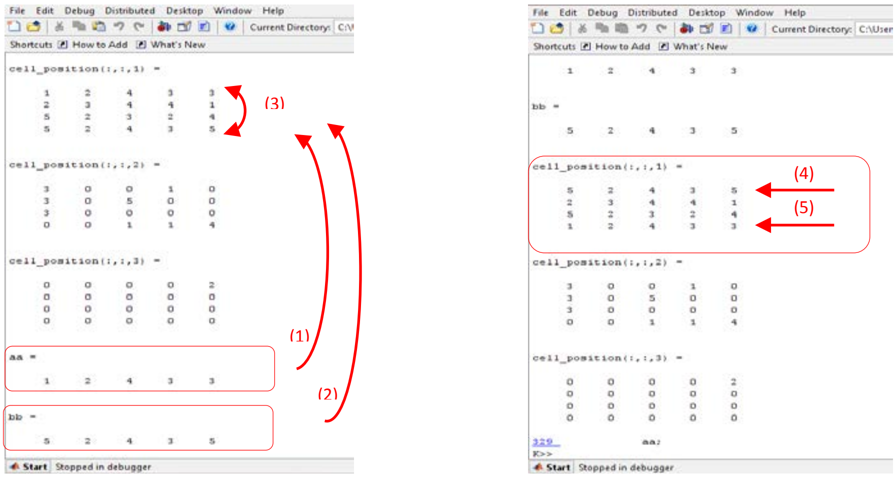
The concept of using a mutation operator is to avoid involvement with the local solution space when other operators cannot generate an improved solution. In such cases, the mutation operator evades the undesirable situation but rearranges either a part of the solution structure or the whole (Fig. 8). Although the use of a mutation operator may temporarily provide faulty solutions, the GA generates enhanced solutions eventually.

a) Suppose $cell\ lyout_k^m \begin{bmatrix} P_{1,1} & P_{1,2} & \cdots & P_{1,i} \\ \dots. & \cdots & \cdots & \cdots \\ P_{j,1} & P_{j,2} & \cdots & P_{j,i} \end{bmatrix}$ is the layout of k^{th} population member in m^{th} generation. (26)

b) If $Y \le MU$ and C=1 and D=j (27)

c) Then, $cell\ lyout_k^m = \begin{bmatrix} P_{j,1} & P_{j,2} & \cdots & P_{j,i} \\ \dots. & \cdots & \cdots & \cdots \\ P_{1,1} & P_{1,2} & \cdots & P_{1,i} \end{bmatrix}$

where *Y* is a random number between 1 and 0; *MU* is the pre-determined value for the mutation rate; and *c* and *d* are the rows that must be replaced.


File Edit Debug Distributed Desktop Window Help
Current Directory: C:\
Shortcuts How to Add What's New
(3)
(1)
(2)
Start Stopped in debugger
File Edit Debug Distributed Desktop Window Help
Current Directory: C:\User
Shortcuts How to Add What's New
(4)
(5)
Start Stopped in debugger

```
cell_position(:,:,1) =

     1     2     4     3     3
     2     3     4     4     1
     5     2     3     2     4
     5     2     4     3     5


cell_position(:,:,2) =

     3     0     0     1     0
     3     0     5     0     0
     3     0     0     0     0
     0     0     1     1     4


cell_position(:,:,3) =

     0     0     0     0     2
     0     0     0     0     0
     0     0     0     0     0
     0     0     0     0     0

aa =

     1     2     4     3     3

bb =

     5     2     4     3     5
```

```
     1     2     4     3     3


bb =

     5     2     4     3     5


cell_position(:,:,1) =

     5     2     4     3     5
     2     3     4     4     1
     5     2     3     2     4
     1     2     4     3     3

cell_position(:,:,2) =

     3     0     0     1     0
     3     0     5     0     0
     3     0     0     0     0
     0     0     1     1     4


cell_position(:,:,3) =

     0     0     0     0     2
     0     0     0     0     0
     0     0     0     0     0
     0     0     0     0     0

329               aa;
K>>
```

Fig. 8. Choosing two rows for replacement (left image)/ replacing the selected rows in the cell layout (right image)

Local Optimum Escaping Operator

After calculating the fitness function for the developed solution in the iterations (for example, X_k^{itr}), the GA algorithm checks the obtained value with respect to the best value observed thus far ($F^{best;itr-1}$). If the fitness function value (F_k^{itr}) is less than $F^{best;itr-1}$, then GA replaces $X_k^{itr}$with $X^{best;itr-1}$. By contrast, if the value of F_k^{itr} is higher than $F^{best;itr}$, then X_k^{itr} is withdrawn immediately. In the proposed GA, the algorithm provides a base to maintain the value with a small probability even after attaining the worst fitness function. This strategy enables the GA to keep searching for improved solutions (Fig. 9). A local escaping operator is added to the algorithm using the function described below:

$$F^{best;itr} = \begin{cases} F_k^{itr}; & if\ F_k^{itr} \leq \min\left\{F_i^{itr}, F^{best;itr-1}\right\} \forall\ i \in itr \\ F_k^{itr}; & if\ R \leq LEP \\ F^{best;itr-1}; & otherwise \end{cases} \tag{28}$$

where *R* is a normal random number between 0 and 1, and *LEP* is a local escaping parameter that is defined by a decision maker. The exact amount of *LEP* cannot be determined, and the value may differ from case to case; however, *LEP* can be approximated on the basis of the experimental design to be explained in Section 3.6.

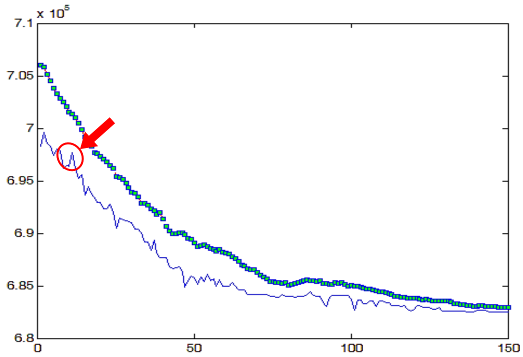


Fig. 9. A sample of escaping from local optimum traps by using local optimum escaping operator

Stopping Criteria

The stopping criteria in the proposed GA are as follows:
1) The maximum number of pre-defined iterations is reached.
2) If no choices are generated in the tournament list during iteration, thus implying that none of the solutions in the iteration can improve the fitness function, then the algorithm is not improved.

a) Suppose $X_k^{itr}(x_1, x_2, \dots, x_i; o_1, o_2, \dots, o_l; b_1, b_2, \dots, b_i)$ is the k^{th} solution in itr^{th} iteration.

b) If $F(X_k^{itr}) > min\left(F(X_1^{itr}), F(X_2^{itr}), \dots, F(X_{k-1}^{itr}); F^*(X_k^{itr-1})\right) \;\; ; \forall\, k \in i$ (29)

Then $Tournament.\,list^{itr} = \emptyset$ for next iteration (30)

4.4 Taguchi Method for Determining the Appropriate Input Sets for the Proposed GA

As mentioned previously, the experimental design helps estimate the effect of each setting parameter and determine the relationship between these parameters. Therefore, the Taguchi method is used using Minitab® 17 while an L_9 (3^4) orthogonal optimization for CdIII removal is considered (Table 2).

Table 2
The level of factors used for L9 orthogonal designing method

Factor	Level 1	Level 2	Level 3
Number of Generations	50	100	200
Pop-size	100	200	300
Mutation	0.05	0.1	0.2
Local Escaping Rate	0.1	0.2	0.3

Table 3 shows the experiments designed by Taguchi method. The value R shows the minimum OFV observed while using the suggested parameters in the proposed GA. The formula which is used for calculating the signal ratio is chosen in a way that lower OFV values are desired more by the software:

$$S.N\ Ratio = -10 \times \log_{10}\left(\frac{\sum OFV^2}{n}\right) \tag{31}$$

Table 3
Results of implemeting the experiments for Taguchi method

Experiment Number	Factor				R	S.N RA
	# of Generations	Pop-size	Mutation	Local Escaping Rate		
	(A)	(B)	(C)	(D)		
1	1	1	1	1	54630	-94.749
2	1	2	2	2	53530	-94.572
3	1	3	3	3	52450	-94.395
4	2	1	2	3	52880	-94.466
5	2	2	3	1	51580	-94.249
6	2	3	1	2	51380	-94.216
7	3	1	3	2	51020	-94.155
8	3	2	1	3	49620	-93.913
9	3	3	2	1	48820	-93.772

Table 4 provides details of analysing the results by Minitab. As seen, all factors A (the number of generations), B (population size), C (mutation rate) and D (local escaping rate) can improve OFV but with different severity levels. Upon implementing the experiments designed for the Taguchi method, the obtained results show that the algorithm is sensitive to the number of generations, population size and to interaction between number of generations and population size (Fig. 10). The high slope of the curve related to the number of generations and population size suggests that these factors significantly affect the objective function (Fig. 11). Mutation rate and local escaping rate have moderate effects on the solving process; however, the positive results indicate that these factors should not be disregarded. Therefore, maintaining a value of 0.1 for mutation- and local escaping rate are appropriate for all experiments.

Table 4
Analyzing results of implementing the designed experiments (Taguchi Analysis: R versus A, B, C, D)

Response Table for Signal to Noise Ratios Smaller is better					Response Table for Means					Response Table for Standard Deviations				
Level	A	B	C	D	Level	A	B	C	D	Level	A	B	C	D
1	-94.57	-94.46	-94.29	-94.26	1	53537	52843	51877	51677	1	*	*	*	*
2	-94.31	-94.24	-94.27	-94.31	2	51947	51577	51743	51977	2	*	*	*	*
3	-93.95	-94.13	-94.27	-94.26	3	49820	50883	51683	51650	3	*	*	*	*
Delta	0.63	0.33	0.03	0.06	Delta	3717	1960	193	327	Delta	*	*	*	*
Rank	1	2	4	3	Rank	1	2	4	3	Rank	2.5	2.5	2.5	2.5

The interaction between factors can be monitored using the regression equation in uncoded units that estimated based on actual values:

$$R = 50767.5 - 1862.5\,A - 1453.5\,B - 662.5\,A * B \tag{32}$$

As seen, in this equation, the interactions between factors A & B influence on the Y (the expected OFV based on actual values) in a constructive manner. The model did not recognize any other interactions between other factors. Therefore, for the larger scale experiments the number of generations and the number of population increase simultaneously. Hopefully there is no negative interaction between factors in the proposed GA. However, such negative interactions (if exist) must be considered carefully while setting the parameters. Hence, the appropriate ranges are 50–100 for small scale problems, 100–150 for medium-scale problems, and of 200–250 for large-scale problems. Any value beyond the specified range increases the volume of computations without noticeable improvements.

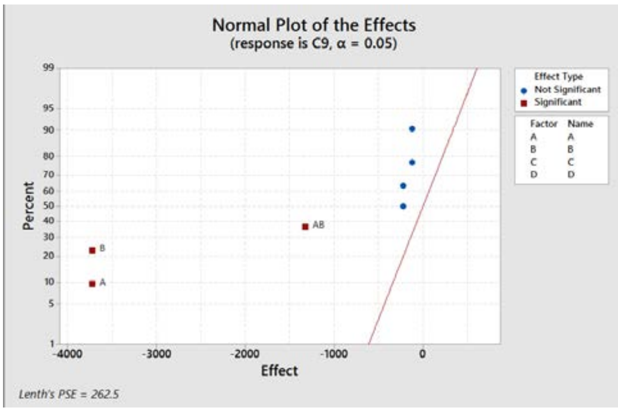


Fig. 10. The normal plot of the effects between input parameters for the proposed hybrid method

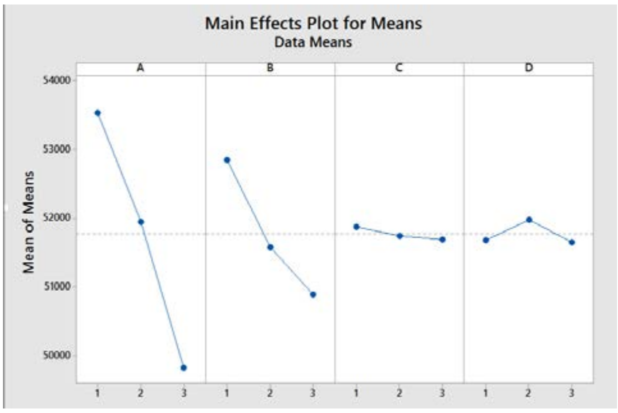


Fig. 11. The main effects plot for showing the impact of levels of input parameters

5. Computational Experience

To determine and to verify the performance of the solving algorithms for the proposed method, a small-scale experiment (sample number 5 in Table 6) is described in detail to determine its dataset, to evaluate computations, and to analyze outcomes. The experimental results obtained are analyzed thoroughly. Seventeen experiments derived from literature are solved in small-, medium-, and large-scale problems. In these experiments are conducted under different conditions (Table 14). The results of other problems are similar and are thus outlined in Table 2. All examples are solved using MATLAB R2009a® software on a Core™ i7 personal laptop with 2.40 GHz CPU and 8 GB RAM.

5.1 Experiment 1

In this example, we assume that four products must be processed in a manufacturing system containing three different types of machines and two cells. We also consider two planning periods. Other related information is presented in Table 5:

Table 5
Product dataset for experiment number 1

Product	Demand (N~(μ,σ²))	Supplier	Outsource Capacity	Batch size	Allowed Back orders	Outsourcing cost (per product)	Backorder cost (per product)	Intracellular cost	Intercellular cost
P1	N~(100,12)	L1	100	15	[20,0]	50	4	3	4
P2	N~(60,8)	L2	50	10	[30,0]	14	3.5	5	8
P3	N~(75,14)	L3	60	8	[45,0]	20	3	4	6
P4	N~(200,4)	L4	120	15	[60,0]	18	2.4	7	8

The information of various machine types must also be incorporated into the model (Table 6):

Table 6
Machine dataset for experiment number 1

Machine	Initial Number	Capacity	Setup Cost	Machine Purchasing cost
M1	5	100	100	8000
M2	7	150	140	12000
M3	4	75	200	7500

Table 7 depicts the operation process for completing each product. For example, product 1 needs to visit machine types 1, 2, 3, and 5.

Table 7
MCIM matrix for experiment number 1

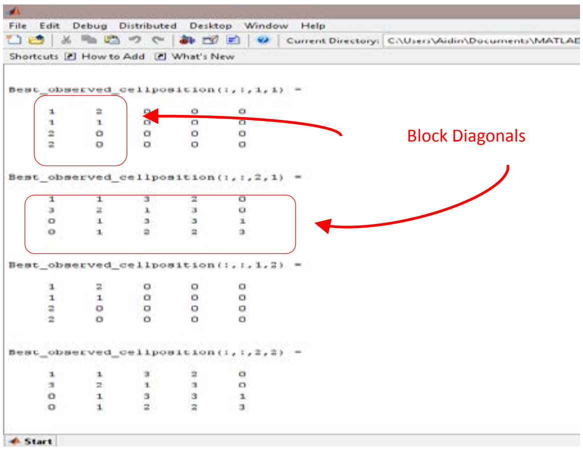
MCIM	M1	M2	M3
P1	1	1	1
P2	0	0	1
P3	0	1	1
P4	1	0	0

The cost of each operation on various machine types is also provided in Table 8. For example, the required costs of performing sequential operations for product 1 are $7 for the operation on machine type 1, $12 for the operation using machine type 2, and $4 for the operation on machine type 3.

Table 8
Operational cost of using each machine type

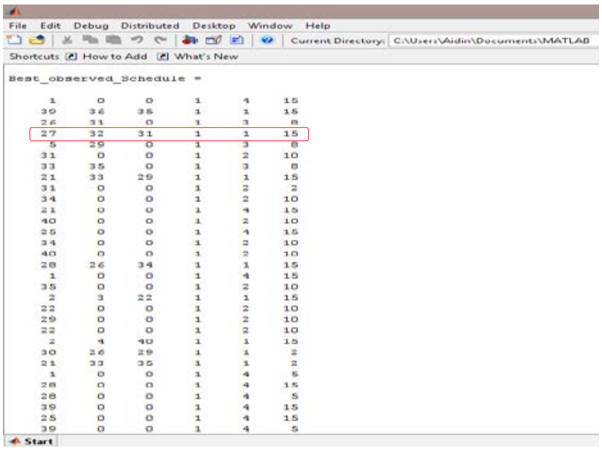
OP	M1	M2	M3
P1	7	12	4
P2	0	0	5
P3	0	9	4.6
P4	8	0	0

For comprehensiveness, all of the results are presented using the result view of MATLAB. The resultant best cell layout is displayed in Fig. 12. The figure indicates that the GA generated two block diagonals through agglomeration. These blocks are maintained until the end of the searching process. Machine re-arrangement is possible; however, this rearrangement is unnecessary in this experiment as per the GA because of low rate (as explained in the next section). Such agglomerative blocks are generated in this example primarily because all products are similar in terms of their use of machine services. Hence the algorithm considers only one center point and attempts to locate other machine groups on the basis of their similarity scores around the center point. In the process, material transfer is as limited as possible. However, if a product uses a different machine type that none of the other product needs (for instance, product 5 needs machine types 4 and 5), then the algorithm sets another center point for this product that facilitates the production of these two different block diagonals. Upon forming cells and locating the machines inside, the next priority is identifying the best parts routes for allocating the operation sequence of products. Parts routes represent a set of required machines (based on MCIM priorities) that are chosen in consecutive order to complete a product. For this purpose, a new matrix is developed that determines the ideal number of parts routes for production. This matrix is presented in Fig. 13.


File Edit Debug Distributed Desktop Window Help
Current Directory: C:\Users\Aidin\Documents\MATLAE
Shortcuts How to Add What's New
Block Diagonals
Start

```
Best_observed_cellposition(:,:,1,1) =

     1     2     0     0     0
     1     1     0     0     0
     2     0     0     0     0
     2     0     0     0     0

Best_observed_cellposition(:,:,2,1) =

     1     1     3     2     0
     3     2     1     3     0
     0     1     3     3     1
     0     1     2     2     3

Best_observed_cellposition(:,:,1,2) =

     1     2     0     0     0
     1     1     0     0     0
     2     0     0     0     0
     2     0     0     0     0


Best_observed_cellposition(:,:,2,2) =

     1     1     3     2     0
     3     2     1     3     0
     0     1     3     3     1
     0     1     2     2     3
```

Fig. 12. The generated block diagonals obtained from the experiment 1

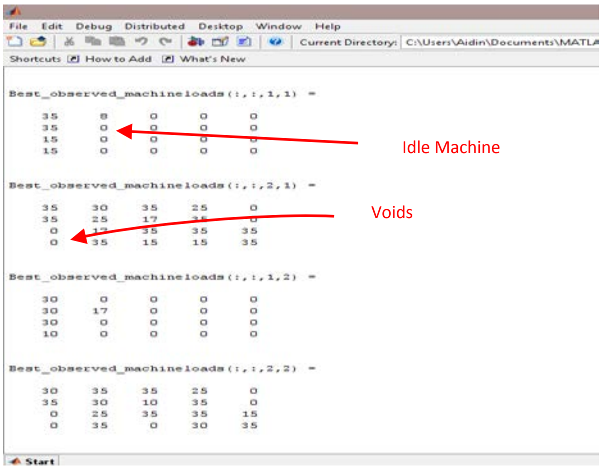

File Edit Debug Distributed Desktop Window Help
Current Directory: C:\Users\Aidin\Documents\MATLAB
Shortcuts How to Add What's New
Start

```
Best_observed_Schedule =

     1     0     0     1     4    15
    39    36    35     1     1    15
    26    31     0     1     3     8
    27    32    31     1     1    15
     5    29     0     1     3     8
    31     0     0     1     2    10
    33    35     0     1     3     8
    21    33    29     1     1    15
    31     0     0     1     2     2
    34     0     0     1     2    10
    21     0     0     1     4    15
    40     0     0     1     2    10
    25     0     0     1     4    15
    34     0     0     1     2    10
    40     0     0     1     2    10
    28    26    34     1     1    15
     1     0     0     1     4    15
    35     0     0     1     2    10
     2     3    22     1     1    15
    22     0     0     1     2    10
    29     0     0     1     2    10
    22     0     0     1     2    10
     2     4    40     1     1    15
    30    26    29     1     1     2
    21    33    35     1     1     2
     1     0     0     1     4     5
    28     0     0     1     4    15
    28     0     0     1     4     5
    39     0     0     1     4    15
    25     0     0     1     4    15
    39     0     0     1     4     5
```

Fig. 13. Part-routes observed for the experiment 1

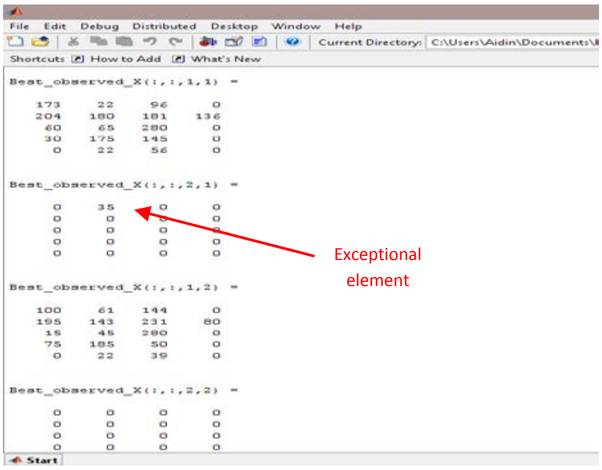
The fourth row in Fig. 13 shows that machines that are located in positions 27, 32, and 31 in the second cell are employed to output 15 units of product 1 in the first period of planning. Generating one unit of product type 1 requires machine types 1, 2, and 3, consecutively. The GA is developed such that it can search the closet set of the mentioned machines in the proposed cell layout to minimize material transfer costs. Therefore, machines that are located in positions 27, 32, and 31 are chosen as to be a machine in the closet set that can complete product 1 at the lowest intracellular WIP transfer costs (see Fig. 12). Scheduling 15 units of product type 1 facilitates the calculation of the minimum value in the remaining product demands, as well as the capacity of the remaining machines and the batch size for product 1 (15). For the remaining parts routes, the same strategy is used in each step.

An important task to be performed after generating layouts and scheduling products is surveying the number of machine loads because this number represents the machine and cell load variations. This calculation can assist decision makers in determining the efficiency of the generated layout and parts routes in practice.


File Edit Debug Distributed Desktop Window Help
Current Directory: C:\Users\Aidin\Documents\MATLA
Shortcuts How to Add What's New
Idle Machine
Voids
Start

```
Best_observed_machineloads(:,:,1,1) =

    35     8     0     0     0
    35     0     0     0     0
    15     0     0     0     0
    15     0     0     0     0


Best_observed_machineloads(:,:,2,1) =

    35    30    35    25     0
    35    25    17    35     0
     0    17    35    35    35
     0    35    15    15    35


Best_observed_machineloads(:,:,1,2) =

    30     0     0     0     0
    30    17     0     0     0
    30     0     0     0     0
    10     0     0     0     0


Best_observed_machineloads(:,:,2,2) =

    30    35    35    25     0
    35    30    10    35     0
     0    25    35    35    15
     0    35     0    30    35
```

Fig. 14. Machine load and voids observed for the experiment 1

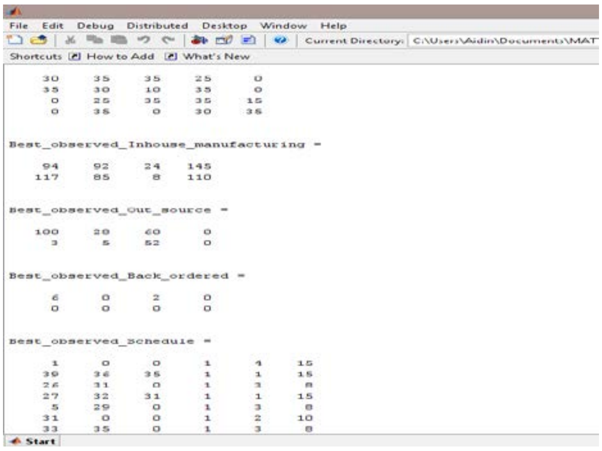

File Edit Debug Distributed Desktop Window Help
Current Directory: C:\Users\Aidin\Documents\
Shortcuts How to Add What's New
Exceptional
element
Start

```
Best_observed_X(:,:,1,1) =

   173    22    96     0
   204   180   181   136
    60    65   280     0
    30   175   145     0
     0    22    56     0


Best_observed_X(:,:,2,1) =

     0    35     0     0
     0     0     0     0
     0     0     0     0
     0     0     0     0
     0     0     0     0


Best_observed_X(:,:,1,2) =

   100    61   144     0
   195   143   231    80
    15    45   280     0
    75   185    50     0
     0    22    39     0


Best_observed_X(:,:,2,2) =

     0     0     0     0
     0     0     0     0
     0     0     0     0
     0     0     0     0
```

Fig. 15. Emerging an exceptional element in final layout

The "*best-observed-machineloads*" matrix in Fig. 15 indicates the number of machine loads in each cell. Machines with less or more load than other parallel machines can thus be recognized. For example, the machine located in the second column of the first row in the first cell of Fig. 14 is allocated only eight times, whereas similar machine types in the same cell are allocated more often (similar machine types can be identified using Fig. 6). Hence this event can be considered is an example of machine load variation that may lead to cell load variation as well. Similarly, if a machine is scheduled more often than other parallel machines in a cell, then this scenario is regarded as an example of machine load variation. The second machine in second row of the first cell is filled by a machine type 1; however, this machine is not used during scheduling in the first period. Hence this machine is an example of an idle machine. Exceptional elements also play a key role in generating efficient cells. Hence, the number of such elements can be minimized as a metric for evaluating the performance of the proposed method. Less exceptional elements emerge in the final layout, and the layout is strengthened. Fig. 15 shows a sample cell layout that contains an exceptional element in cell 2.

The total number of in-house manufacturing processes in each period is shown in the "*best-observed-inhouse manufacturing*" matrix depicted in Fig. 16. For example, the amount of product 1 that must be manufactured in the first period is 97 units in this matrix. That of product 4 in the second unit is 110 units. The capacity of machines in cells is inadequate; therefore, the algorithm set some units to be completed through subcontractor services. The *"best-observed-out-source"* matrix indicates the number of units that are scheduled for outsourcing. For example, 28 units of product 2 are scheduled to be completed through subcontractor services in the first period. As mentioned previously, the remaining product demands that cannot be completed through in-house manufacturing and outsourcing services are regarded as backorders. In this experiment, six units of product 1 and two units of product 3 are postponed to the second period.


File Edit Debug Distributed Desktop Window Help
Current Directory: C:\Users\Aidin\Documents\MAT
Shortcuts How to Add What's New
Start

```
    30     35     35     25      0
    35     30     10     35      0
     0     25     35     35     15
     0     35      0     30     35

Best_observed_Inhouse_manufacturing =

    94     92     24    145
   117     85      8    110

Best_observed_Out_source =

   100     20     60      0
     3      5     52      0

Best_observed_Back_ordered =

     6      0      2      0
     0      0      0      0

Best_observed_Schedule =

     1      0      0      1      4     15
    39     36     35      1      1     15
    26     31      0      1      3      8
    27     32     31      1      1     15
     5     29      0      1      3      8
    31      0      0      1      2     10
    33     35      0      1      3      8
```

Fig. 16. Results of in-house manufacturing, outsourcing and back orders achieved in final solution of experiment 1

Table 9 shows the percentage of machine loads in each cell as per the final solution. For example, the value $0.146, M1$ in the first position of cell 1 indicates that the first location in cell 1 is filled by machine type 1 and that 14.6% of all of the loads on machine type 1 are allotted during the first planning period.

Table 9
Results of machine-loads percentage in second planning period of final solution of experiment number 1

Period	Cell	Load-Machine%				
t:1	C1	0.146,M1	0.068,M2	0	0	0
		0.146,M1	0	0	0	0
		0.127,M2	0	0	0	0
		0.127,M2	0	0	0	0
	C2	0.146,M1	0.125,M1	0.167,M3	0.212,M2	0
		0.167,M3	0.212,M2	0.071,M1	0.167,M3	0
		0	0.071,M1	0.167,M3	0.167,M3	0.146,M1
		0	0.146,M1	0.127,M2	0.127,M2	0.167,M3
t:2	C1	0.132,M1	0	0	0	0
		0.132,M1	0.075,M1	0	0	0
		0.240,M2	0	0	0	0
		0.080,M2	0	0	0	0
	C2	0.132,M1	0.154,M1	0.167,M3	0.200,M2	0
		0.167,M3	0.240,M2	0.044,M1	0.167,M3	0
		0	0.11,M1	0.167,M3	0.167,M3	0.066,M1
		0	0.154,M1	0	0.240,M2	0.167,M3

After determining the machine loads over upper and lower control limits, each value must be investigated because not all of the loads exhibit machine load variation. Some loads may be generated by the allocation of previous parts routes; hence, machine load variation must not be considered. We explain this condition using an example. Suppose 25 units of product i in a period plan t are regarded as product demand. This product must be served by machine types 1 and 2. Two units of machine type 1 are located in cell positions 12 and 17. Moreover, we also suppose two units of machine type 2 that are located in cell positions 26 and 28. The effective capacity of machine type 1 is 20 units per planning period; that of machine type 2 is 22 units per planning period. If the batch size of product i is 7, a sample parts route may be generated as in Table 10.

Table 10
Determining the amounts of scheduling on each part routes

Part route number	Machines layout position	Remained capacity of each machine	minimum production allowed on this part route (min machine capacity; batch size; remained product demand)	scheduled amount	remained product demand
1	C12, C26	20,22	min (20,22,7,25)=7	7	25-7=18
2	C17, C26	20,15	min (20,15,7,16)=9	7	18-7=11
3	C12, C28	13,22	min (13,22,7,11)=7	7	11-7=4
4	C12,C26	6,8	min(6,8,7,4)=4	4	4-4=0

In the final part route, the scheduled amount is 4. This value is 57.14% less than that of other machines. If we use Eq. (32) (k = 1), then the final part route determines the control limits (Fig. 17); however, this machine should not be considered idle because no product demand needs to be scheduled in this parts route.

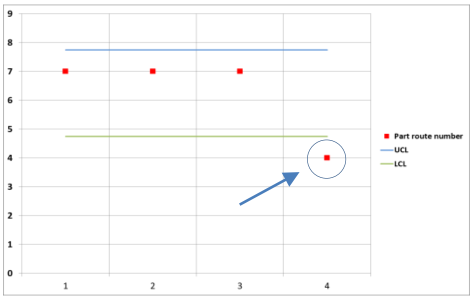


Fig. 17. An example of machine loads fallen out of the control limits but is not an idle machine

After determining the parts routes (see Fig. 13) and each machine load (see Fig. 14) and calculating the formula presented in Equation 32, each value must be analyzed to identify whether the machine is idle or over-loaded. Moreover, whether or not the achieved value should be considered noise due to product demands is investigated as well. To determine machine load variation in experiment 1, the results derived from Table 9 are analyzed further. Table 11 indicates the number of machine loads for each machine type for the first period in the final solution. For instance, machine type 1 in first cell has a load of 70. This cell contains two type 1 machines (see Fig. 12), therefore, the loads on machine type 1 can be averaged as 35 (70/2).

Table 11
Results of machine loads in first period of final solution of the experiment 1

Loads/ Machines (*t=1*)	C1			C2		
	M1	M2	M3	M1	M2	M3
Number of loads	70	38	0	169	80	210
Maximum load	35	15	0	35	25	35
loading average	35	12.67	0	28.16	20	35

The control limit lines for each machine type in the second period of the final solution are calculated according to the findings regarding machine loads in each cell (Table 12). These results are then considered in drawing a control chart for each machine type and in determining idle or over-allocated machines in a cell (Fig. 18).

Table 12
Control limits for each machine type in best observed solution for the experiment 1

Control Limits	t=1			t=2		
	M=1	M=2	M=3	M=1	M=2	M=3
U.C.L	45.89	31.55	35.00	45.98	34.11	35.00
L.C.L	10.44	8.45	35.00	4.02	22.56	35.00

Fig. 18 indicates that all of the charts are within the control limits in the final layout. We assumed that $k = 1$ when drawing the charts. As discussed previously, parameter K is dependent on the product demands, number of machines, and batch sizes. This parameter can be set by a decision maker.

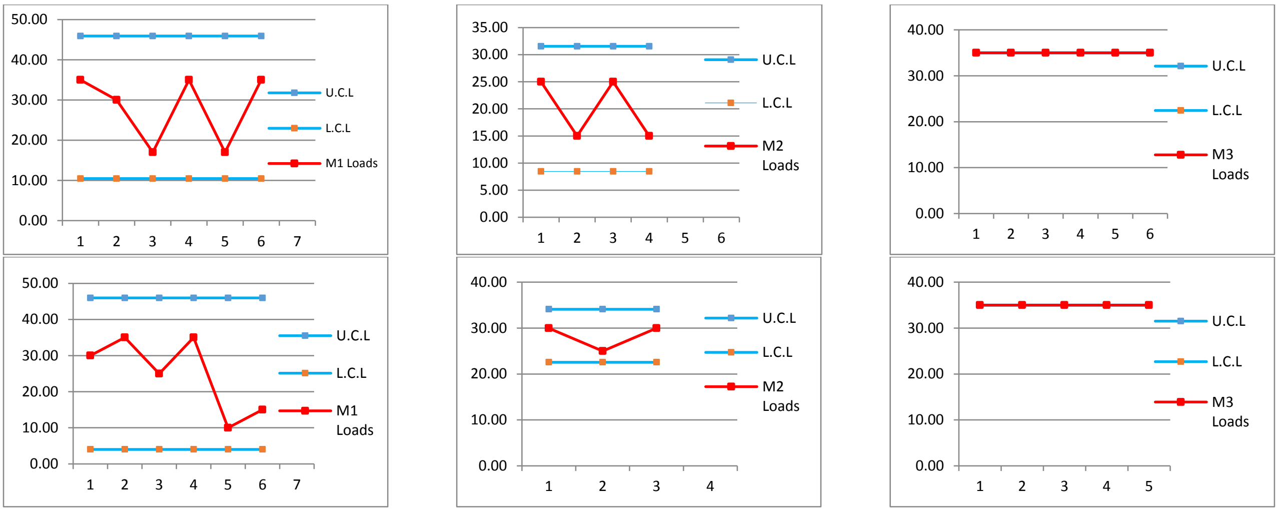


Fig. 18. Control charts for machine-loads in final result of the experiment 1

The final layout can be analyzed further. Table 13 presents the analysis of the solved experiment. In CMS studies, bottleneck products are products that must visit machines located in different cells. The development of bottleneck parts is attributed to many reasons in literature, including technical problems in machine relocation, safety problems, and the existence of a unique machine type in one cell. In this research, the restricted machine capacity in one cell can force materials to transfer to other cells for visiting parallel machines. The numbers of intercellular material movements are 3 and 5 in first and second planning periods, respectively; these numbers represent the number of bottleneck products. The amount of backorders in the second period is zero, which is in accordance with the problem assumptions in Section 3.1.

Table 13
Results of analyzing best observed layout for experiemnt number 1

Metrics	t:1	t:2
Number of Machine Loads	567	562
Percentage of existed exceptional elements	0	0
Number of observed voids after scheduling	1	1
Amount of outsourcing	188	60
Amount of backorders	8	0
Number of intercellular movements	3	5
Number of intracellular movements	16	14
Number of observed part routings	35	28
Number of bottleneck machines	0	0
Maximum machine load variation	0	0
Maximum cell load variation	0	0

5.2 Other Experiments

In this section, 17 experiments are solved with data gathered from literature. Our findings demonstrate that in all investigated cases, the GA can provide high-quality solutions (Table 14). Fig. 19 shows the process of optimizing fitness function value over generations in some of the experiments. A negative slope in the trend lines highlights the high system cost minimization performance of the proposed GA. In Fig. 14, the underlying line shows the minimum solution in each generation while the upper line represents the average fitness function values observed in a similar generation. The trend line illustrates how the GA can escape from local optima through mutation and local optimum escaping operators.

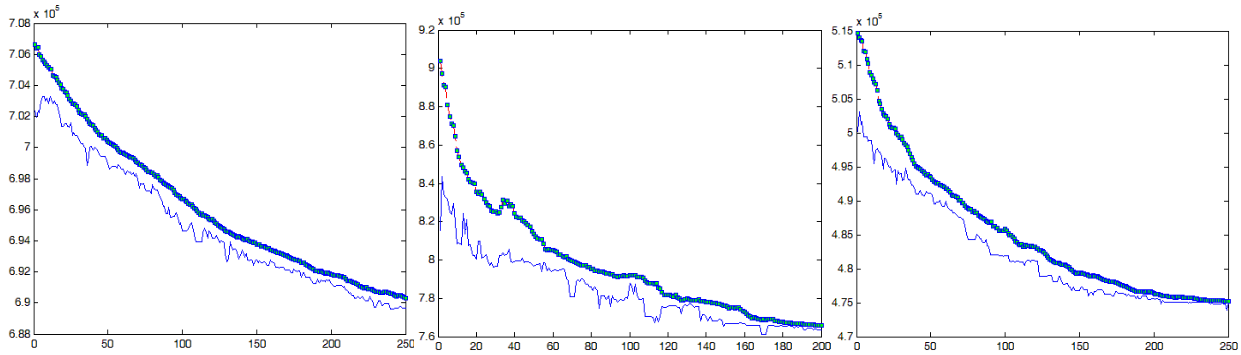


Fig. 19. Results of Fitness Function Trend Lines for Examples 11, 12 & 16 of table 14 (respectively)

Table 14

Comparing Results of Lingo 12.0 and the proposed GA for solving a number of experiments[2]

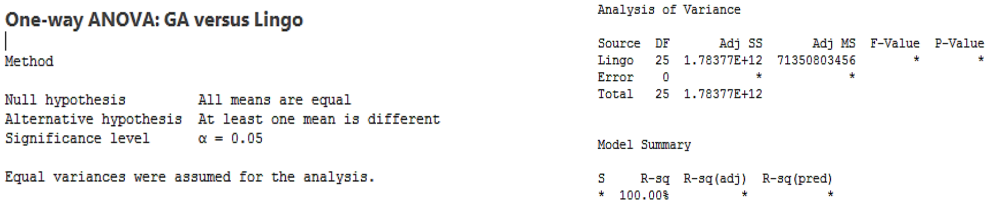
Row	Problem	Number of Int. Vars.*	GA				Lingo 12.0			Diff	M.I.P.M*** (inter-cellular)	M.L.P.M**** (inter-cellular)	M.I.P.M (intra-cellular)	M.L.P.M (intra-cellular)
	(i/j/s/k/t)		Gen	Pop.size	Best solution	CPU Time	Itr. **	Best solution	CPU Time					
1	2/2/2/2/3	96	50	50	58500	3.15	67	58500.0	2.87	0.000	0.880	0.960	0.500	0.460
2	2/2/2/2/5	160	75	50	191700	5.57	553	191700.0	3.13	0.000	0.960	0.860	0.580	0.640
3	2/2/2/2/7	224	100	75	274800	9.39	447	274800.0	6.20	0.000	1.013	0.893	0.467	0.680
4	2/2/2/2/9	288	100	120	362500	17.04	167	362500.0	10.33	0.000	0.992	1.066	0.542	0.417
5	4/3/3/2/3	432	150	200	10940	75.20	456	11940.0	65.02	8.375	3.040	2.985	2.520	2.460
6	4/3/3/2/5	720	150	200	31032	100.86	398	34172.0	78.03	9.189	3.095	2.935	2.525	2.505
7	4/3/3/2/8	1152	250	200	62932	379.43	606	63640.0	240.32	1.113	3.065	3.070	2.525	2.460
8	6/4/4/3/3	1728	200	200	675281	388.56	887	684034.5	411.80	1.280	8.150	7.895	3.260	3.465
9	6/4/4/3/5	2880	150	200	391570	499.12	1230	394079.5	482.63	0.637	8.390	8.225	3.070	3.235
10	6/4/4/3/7	4032	200	150	486840	714.73	1779	551819.5	693.22	11.775	8.113	7.700	3.387	3.327
11	6/4/4/3/10	5760	250	300	689630	182.71	2916	689736.0	224.03	0.015	8.023	7.790	3.353	3.467
12	7/3/3/3/3	1134	200	200	761085	140.02	589	918925.0	89.43	17.177	3.920	3.990	2.375	2.265
13	7/3/3/3/5	1890	200	200	75862	155.34	940	84811.30	141.67	10.552	3.965	4.075	2.370	2.300
14	7/3/3/3/7	2646	200	250	109040	189.25	1100	117111.8	137.24	6.892	4.048	3.956	2.364	2.460
15	7/3/3/3/9	3402	250	250	143080	393.57	1980	150133.4	411.73	4.698	3.968	4.036	2.412	2.328
16	9/4/4/4/3	3456	250	250	473869	332.28	1700	486088.0	311.03	2.514	*7.876*	*8.032*	*4.912*	*5.104*
17	9/4/4/4/5	5760	300	250	788780	427.41	3130	799458.0	395.22	1.336	7.840	8.232	5.108	4.636
18	9/4/4/4/7	8064	300	250	110810	999.36	8064	1113641	822.64	0.498	8.056	8.056	4.892	4.788
19	9/4/4/4/9	10368	300	250	141510	873.82	3889	1419174	937.06	0.287	8.036	8.176	4.972	4.784
20	12/4/4/3/3	3456	250	250	404530	544.78	1431	409547.2	465.32	1.225	*8.0280*	*7.9920*	*6.4920*	*6.6320*
21	12/4/4/3/5	5760	300	300	534034	837.84	3109	539466.9	826.17	1.007	8.1960	8.0560	6.5120	6.4200
22	12/4/4/3/7	8064	300	300	750620	578.51	4684	938647.2	651.71	20.032	8.0833	7.8000	6.6133	6.8233
23	15/4/4/5/3	7200	300	250	442680	561.32	7200	462412.3	732.08	4.267	9.820	9.433	2.927	2.980
24	15/4/4/5/5	12000	300	250	753750	808.79	5319	765911.0	939.89	1.588	9.756	9.736	2.768	3.060
25	15/4/4/3/7	10080	300	300	106050	923.93	5370	1068340	872.65	0.734	*9.557*	*9.660*	*2.970*	*2.997*
26	15/4/4/3/9	12960	300	300	135850	844.23	6644	1362692.	1022.53	0.308	7.8920	7.9680	8.3720	8.5440

**Number of Integer Variables* ****Mean of Initial Period Movements*
***Number of Iterations* *****Mean of Last Period Movement*

Results in Table 14 shows that for small and medium size problems GA takes more time to achieve the final solutions but for large scale problems the proposed GA can provide better solutions in less computation time. In addition, in all medium and large scale problems GA can provide solutions with better quality. To compare the outcomes of GA and Lingo 12.0 for the proposed method, a single way

[2] *. In order to get dataset readers can email to the first author of this research.*

ANOVA is done using Minitab® 17.0 software. The results show that in the confidence level of 95%, the proposed GA can provide results with better quality (Fig. 20).



```
One-way ANOVA: GA versus Lingo

Method

Null hypothesis         All means are equal
Alternative hypothesis  At least one mean is different
Significance level      α = 0.05

Equal variances were assumed for the analysis.
```

```
Analysis of Variance

Source  DF      Adj SS       Adj MS  F-Value  P-Value
Lingo   25  1.78377E+12  71350803456        *        *
Error    0            *            *
Total   25  1.78377E+12


Model Summary

S     R-sq  R-sq(adj)  R-sq(pred)
*  100.00%          *           *
```

Fig. 20. Results of One way ANOVA for comparing the mean of GA and Lingo 12.0 (α=0.05)

In this section the results of solving the experiment number 6 in table 14 with Lingo 12.0 and GA are explained in details.

```
Local optimal solution found.
Objective value:               34172.00
Objective bound:               34172.00
Infeasibilities:               0.000000
Extended solver steps:         0
Total solver iterations:       398

Model Class:                   PINLP

Total variables:               720
Nonlieanr variables:           720
Integer variables:             720
Total constraints:             856
Nonlinear constraints          721
Total nonzeros:                3530
Nonlinear nonzeros:            1800
```

Fig. 21. Results Observed for case study number 6 solved with Lingo 12.0 (medium size problem)

```
ans =
   2.5250      2.6200      2.4000      2.5100

ans =
   3.0950      3.0100      3.0500      2.8800

ans =

               31.032


            Elapsed time is 100.858822 seconds.
```

Fig. 22. Results Observed for case study number 6 solved with the Proposed GA (medium size problem)

As shown in Fig. 21, the Lingo 12.0 was able to find the local optimum solution. Fig. 22 in contrast shows the results of the proposed GA. the first 2 rows in this figure show the average of intercellular and intracellular movements for completing one product of each product type in final solution. The rest of the results are compared in Table 15. Hence for the rest of the research only results of GA will be represented and explained.

Table 15
Comparing results of Lingo 12.0 and GA for a medium size problem

Metrics	GA	Lingo 12.0
Objective Function	31032	34172.0
CPU time	100.86	78.03
Number of Variables	720	720
Average of Inter cellular movements per batch	8.54	-
Average of Cellular Movements per batch	7.96	-
Number of observed voids	0	1

The results of the solving all experiments without control limits show that demands for dynamic parts can disrupt material transfer fluency (parts routes).

```
Best_observed_cellposition( : , : , 1 , 3) =

   1     4     2     3     0     0
   1     3     3     4     0     0
   1     2     4     4     0     0
   2     1     3     2     0     0
   3     2     3     4     0     0

Best_observed_cellposition( : , : , 2 , 3) =
   0     0     0     0     0     0
   0     0     0     0     0     0
   0     0     0     0     0     0
   0     0     0     0     0     0
   0     0     0     0     0     0
```

```
Best_observed_X (:, : , 1 , 3) =

  60    62    60    26     0     0
  96    36    58   122     0     0
  92   212   182   120     0     0
  26   292    26   148     0     0
   0    74    34   134     0     0

Best_observed_X (:, : , 2 , 3) =
   0     0     0     0     0     0
   0     0     0     0     0     0
   0     0     0     0     0     0
   0     0     0     0     0     0
   0     0     0     0     0     0
```

Fig. 23. Emerging complex dummy sub-cells after scheduling in a CMS

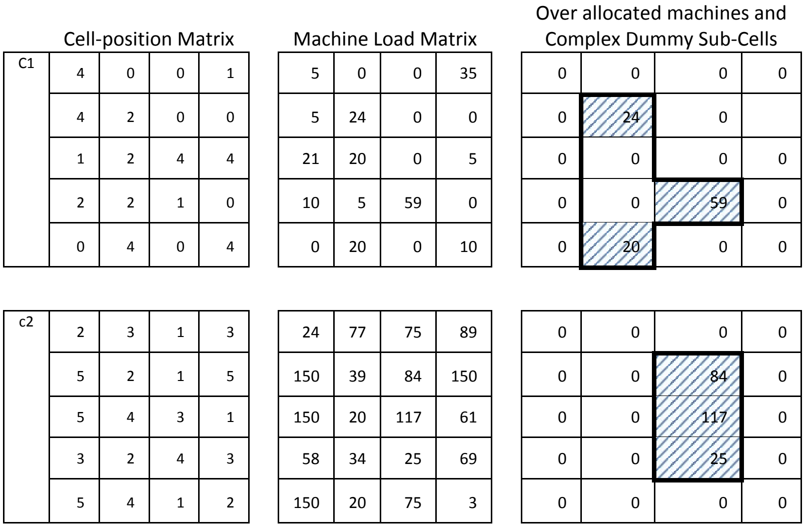
Additional short-distance intra-cellular routes are generated by increasing the number of products or product demands. As a result, the number of bottleneck machines increases, whereas other parallel machines remain idle. As per a comparison of the numbers of inter- and intra-cellular movements, intra-cellular WIP movements increased in the time horizon in 53.8% of the studied cases under dynamic parts demand. Although such movements may be useful and may reduce material transfer costs, these movements may induce cell load variation. This variation is concisely called "*complex dummy sub-cells*". This phenomenon emerges as a consequence of increasing the setting of short-cut distances in material transfer on parts routes that enhance cell load variation. Fig. 23 presents the "*complex dummy sub-cells*" as a CMS output. The sub-cells do not follow a specific rule and may emerge in any area in a cell depending on the arrangements of the machines and on the parts routes scheduled on the machines. This phenomenon illustrates the tendency of systems to allocate WIPs to a set of machines that are located contiguously. Parallel machines in the same cell are located farther away and remain idle. To examine the level of cell sensitivity to "*dummy complex sub-cells,*" a metric is developed on the basis of the neighborhood radius of the WIPs.

5.3 Proposing a System Entropy Metric for Calculating Cell-load variation:

In small layouts, dummy sub-cells are easily recognized by the naked eye. However, these cells are difficult to identify as they grow in number or size. Hence, a mathematical metric must be used. To investigate the effect of uncertain product demands on cell load variation, a mathematical metric is developed that operates on the basis of tracking material transfer between and within each cell in the planning horizon. The metric can indicate the possibility of existing dummy sub-cells in a region with neighborhood radius R.

$$CDSB_{p,q} = (\left\{M_{(p,q),j,k,t} \middle| \left(\frac{Machine\text{−}load_{j,k,t}}{\sum_k^K Machine\text{−}load_{j,k,t}} > \bar{X}_{j,k,t} + kS_{j,k,t}{}^2\right); \forall\, j\ \in (R-1 \le p \le R+1\,, R-1 \le q \le R+1\,)); k,t\right\} \qquad (33)$$

where $M_{p,q,j,k,t}$ is machine type j that is placed in row p and column q of cell k during period t. $\bar{X}_{j,k,t}$ is the average machine load on machine type j in cell k during period t. $S_{j,k,t}$ is the standard deviation of machine type j. R is the neighborhood radius that shows the cubic area around each center point (p,q). Fig. 24 depicts an example of dummy sub-cell recognition in the first period of the final solution for experiment 8 while the machine loads are not controlled using control limits.



	Cell-position Matrix				Machine Load Matrix				Over allocated machines and Complex Dummy Sub-Cells			
C1	4	0	0	1	5	0	0	35	0	0	0	0
	4	2	0	0	5	24	0	0	0	24	0	
	1	2	4	4	21	20	0	5	0	0	0	0
	2	2	1	0	10	5	59	0	0	0	59	0
	0	4	0	4	0	20	0	10	0	20	0	0
c2	2	3	1	3	24	77	75	89	0	0	0	0
	5	2	1	5	150	39	84	150	0	0	84	0
	5	4	3	1	150	20	117	61	0	0	117	0
	3	2	4	3	58	34	25	69	0	0	25	0
	5	4	1	2	150	20	75	3	0	0	0	0

Fig. 24. Results of $\boldsymbol{CDSB_{p,q}}$ Equation for first period in final solution of experiment number 8

The formula above is used in a macro program of MATLAB 2009. This macro works based on the set center point, the considered cubic block around the center point and all of the recognized sets of machines that are over-allocated in this area given the relations defined in MCIM. If the algorithm detects an overloaded machine in the cell, then all of the processes mentioned earlier are repeated to increase R

values until all members of the complex sub-cells are recognized. The outcome is the set of over-allocated machines located nearby.

Table 16
Observed dummy sub cells in the solved experiments

No.	t=1	t=2	t=3	t=4	t=5
1	-	-	-	-	-
2	{1,2}	{2,5}	{1,7}	{2,6}	-
3	-	-	-	-	-
4	{10,12}	{7,12,16}	-	-	-
5	{8,9,,19}	{3,8,14,18,19}	{4,8,13,19}	{4,8,9,12,19}	-
6	{12,13,17,18}	{7,10,12,13}	-	-	-
7	{21,27,28}	-	-	-	-
8	{23,32,34,35}	{4,12}; {22,28,31}	{4,13,18}	{7,12}; {23,29}	-
9	{1,2,4,11}; {34,36}	{1,8,19};{34,36}	{5,6};{26,28}		-
10	{7,11};{28,29}	{7,9,10}	{9,11};{26,29}	-	{9,10,12,14};{26,29}
11	{5,11,12}	-	{2,3}	{3,15,24}	
12	-	-	-	-	-
13	-	-	-	-	-
14	-	-	-	-	-
15	-	-	{6,7}	{5,8,11}	-
16	-	-	-	-	-
17	{18,23,26}	{22,25}	-	-	-

The use of this algorithm can help a decision-maker consider complex dummy sub-cells in cubic blocks of various sizes. For example, the dummy sub-cell block of over-allocated machines can be identified as highlighted in Fig. 24 in the second cell of experiment 8 when neighborhood radius (*R*) 1 is considered. However, if we assume that *R* = 1 for the first cell of experiment 8, then each over-allocated machine cannot identify other nearby over-allocated machines. Instead, if *R* is assumed to be 2, then a dummy complex can be recognized as illustrated in Fig. 24. The remaining dummy sub-cells in the solved cases are presented in Table 16. As shown although in all cases, the proposed method found the minimum total system cost (which contains feasible part routes) but in many cases dummy sub-cells emerged which can be considered as a reason for increasing cell load variation. Afterward, the experiments are run once again considering the mentioned control limits during the part routing process in each stage. The results that are shown in the Table 17 indicate that the proposed method can effectively avoids machine load variation in all experiments.

Table 17
Results of Solving Numerical Examples Gained from the Literature

Scale	No.	Problem Source	K I m j L T C.S NOP	d(i,t)	GA Results				BD	EE	Vo.	MLV	CPU time (Seconds)
					Gen	Pop.size	Mu	Total System Cost					
Small Size	1	Askin & Huang (2001)	2 2 2 2 2 4 8 [2 4]	d~N(300,50)	50	100	0.1	4454	2	0	0	0	7.55
	2	Askin & Huang (2001)	2 2 4 2 2 4 20 [3 6]	d~N(400,80)	50	100	0.1	6304	1	0	0	0	11.73
	3	Suer & Cedeno (1996)	1 4 4 4 4 4 15 [3 2 3 2]	d~N(300,100)	50	100	0.1	19550	2	1	3	0	9.61
	4	Mahdavi et al. (2010)	2 4 4 4 4 2 20 [4 5 3 6]	d~N(300,100)	100	100	0.1	20300	1	0	0	0	27.24
	5	Mahdavi et al. (2012)	2 4 4 4 4 4 16 [4 3 6 5]	d~N(220,80)	100	100	0.1	17090	1	0	0	0	78.13
	6	Aryanezhad et al. (2009)	3 3 3 3 3 3 20 [5 4 5]	d~N(300,100)	100	100	0.1	17360	2	0	0	0	27.27
Medium Size	7	Askin & Huang (2001)	8 2 2 2 2 4 4 [14 6]	d~N(500,150)	200	200	0.1	1970	5	0	0	0	13.77
	8	Askin & Huang (2001)	8 2 2 2 2 4 4 [8 8]	d~N(200,100)	200	200	0.1	3170	5	0	1	0	12.81
	9	Aryanezhad et al. (2009)	4 5 5 5 5 4 12 [2 4 3 4 4]	d~N(500,170)	250	300	0.1	11690	2	1	2	0	15.38
	10	Li et al. (2012)	2 5 5 5 5 4 15 [3 5 4 2 3]	d~N(400,200)	150	200	0.1	27510	2	1	1	0	107.5
	11	Mahdavi et al. (2010)	2 6 6 6 6 3 20 [5 4 3 6 5 4]	d~N(300,75)	200	200	0.1	17800	2	0	1	0	71.74
	12	Mahdavi et al. (2012)	2 5 5 5 5 4 15 [7 8 5 9 4]	d~N(300,150)	150	200	0.1	51666	2	2	2	0	227.94
	13	Norman et al. (2002)	2 6 6 6 6 5 14 [3 5 2 4 3 4]	d~N(500,200)	200	200	0.1	11190	2	1	0	0	107.37
Large	14	Askin & Huang (2001)	2 8 6 6 8 4 30 [3 2 2 3 2 3]	d~N(500,140)	200	200	0.1	12610	2	0	1	0	102.78
	15	Askin & Huang (2001)	2 8 8 8 8 4 20 [2 4 3 5 4 3 2 4]	d~N(350,75)	200	200	0.1	20765	2	0	1	0	212.43
	16	Aryanezhad et al. (2009)	5 5 5 5 5 5 30 [4 3 4 3 3]	d~N(200,100)	300	300	0.1	11235	2	0	1	0	64.62
	17	Aryanezhad et al. (2009)	5 4 5 5 4 4 8 [4 3 4 3 3]	d~N(500,150)	300	300	0.1	11370	2	1	2	0	120.83

Mu: Mutation rate BD: Block Diagonals EE: Exceptional Elements Vo.: Voids MLV: Machine load variation

* While gathering data in each case, the authors endeavored to use applicable data as much as possible. We generated the remaining data needed for solving problems that did not appear in each case.

6. Conclusion

This paper has proposed a new method for determining the best tradeoff values between in-house manufacturing and outsourcing in a restricted capacity D-CMS. The effect of uncertain market demands on D-CMS has been analyzed, and the findings have revealed that uncertain product demands could aggravate disruption in material transfer routes. This increase facilitates increasing machine load variation and generation of complex dummy sub-cells. As discussed, the existence of complex dummy sub-cells establishes bottleneck machines, whereas other parallel machines remain idle. It is shown that the market changes have significant impact on machine loads in cellular manufacturing systems and can cause cell-load variations in 47% of the studied cases.

To control the cell load variation a new controlling method is used during the process of part routing of products. Using this method, parallel machines in a cell will be allocated not more than an upper control limit that is calculated using the mean and standard deviation of a specific machine type in a cell. Such method can prevent over loading a machine while other parallel machines are remained idle or less allocated.

Since the medium- and large-scale D-CMS problems cannot be solved using regular optimizing methods, to use the proposed cell-load variation controlling method, a GA is proposed. In the first step in order to verify the GA, the outcomes of the GA and Lingo 12.0 are compared using 26 case studies. Although both algorithms reported the same results for small size problems but for medium and large scale problems, the GA can provided solutions with better quality. In continue 17 datasets that are gathered from the literature have been solved. Then the outcomes have been analyzed using a metric that are developed to calculate the machine load variation in cells. The results have shown that the proposing method could significantly control the machine load variation for all machine types and sub consequently avoids cell load variation in all studied cases. Future studies can investigate the effects of complex dummy sub-cells on maintenance costs.

Acknowledgements

The authors would like to thank Dr. Mohammad Gholami (Post-doctoral fellow; University of Calgary-Abb.CA) for his positive comments during the writing of this manuscript.

References

Agarwal, A., & Sarkis, J. (1998). A review and analysis of comparative performance studies on functional and cellular manufacturing layouts. *Computers & industrial engineering, 34*(1), 77-89.

Ahkioon, S., Bulgak, A., & Bektas, T. (2009). Cellular manufacturing systems design with routing flexibility, machine procurement, production planning and dynamic system reconfiguration. *International Journal of Production Research, 47*(6), 1573-1600. doi: 10.1080/00207540701581809

Ariafar, S., Firoozi, Z., & Ismail, N. (2014). *A Triangular Stochastic Facility Layout Problem in a Cellular Manufacturing System.* Paper presented at the International Conference on Mathematical Sciences and Statistics 2013.

Ariafar, S., & Ismail, N. (2009). An improved algorithm for layout design in cellular manufacturing systems. *Journal of Manufacturing Systems, 28*(4), 132-139. doi: 10.1016/j.jmsy.2010.06.003

Baker, R., & Maropoulos, P. G. (2000). Cell design and continuous improvement. *International Journal of Computer Integrated Manufacturing, 13*(6), 522-532.

Banerjee, I., & Das, P. (2012). Group technology based adaptive cell formation using predator–prey genetic algorithm. *Applied Soft Computing, 12*(1), 559-572.

Boulif, M., & Atif, K. (2006). A new branch-&-bound-enhanced genetic algorithm for the manufacturing cell formation problem. *Computers & operations research, 33*(8), 2219-2245.

Chen, M., & Cao, D. (2004). Coordinating production planning in cellular manufacturing environment using Tabu search. *Computers & industrial engineering, 46*(3), 571-588.

Defersha, F. M., & Chen, M. (2006). A comprehensive mathematical model for the design of cellular manufacturing systems. *International Journal of Production Economics, 103*(2), 767-783.

Dimopoulos, C., & Mort, N. (2001). A hierarchical clustering methodology based on genetic programming for the solution of simple cell-formation problems. *International Journal of Production Research, 39*(1), 1-19.

Durmusoglu, M. B., & Nomak, A. (2005). GT cells design and implementation in a glass mould production system. *Computers & industrial engineering, 48*(3), 525-536.

Egilmez, G., & Süer, G. (2014). The impact of risk on the integrated cellular design and control. *International Journal of Production Research, 52*(5), 1455-1478.

Egilmez, G., Süer, G. A., & Huang, J. (2012). Stochastic cellular manufacturing system design subject to maximum acceptable risk level. *Computers & Industrial Engineering, 63*(4), 842-854.

Elmi, A., Solimanpur, M., Topaloglu, S., & Elmi, A. (2011). A simulated annealing algorithm for the job shop cell scheduling problem with intercellular moves and reentrant parts. *Computers & industrial engineering, 61*(1), 171-178.

Goncalves Filho, V., & José Tiberti, A. (2006). A group genetic algorithm for the machine cell formation problem. *International Journal of Production Economics, 102*(1), 1-21.

Gonçalves, J. F., & Resende, M. G. (2004). An evolutionary algorithm for manufacturing cell formation. *Computers & industrial engineering, 47*(2), 247-273.

Gravel, M., Luntala Nsakanda, A., & Price, W. (1998). Efficient solutions to the cell-formation problem with multiple routings via a double-loop genetic algorithm. *European Journal of Operational Research, 109*(2), 286-298.

Gupta, Y. P., Gupta, M. C., Kumar, A., & Sundram, C. (1995). Minimizing total intercell and intracell moves in cellular manufacturing: a genetic algorithm approach. *International Journal of Computer Integrated Manufacturing, 8*(2), 92-101.

Haleh, H., Iranmanesh, H., & Kor, H. (2009). *A new hybrid evolutionary algorithm for solving multi objective cell formation problem.* Paper presented at the International Conference on Computers & Industrial Engineering, 2009. CIE 2009.

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

Hsu, C.-M., & Su, C.-T. (1998). Multi-objective machine-component grouping in cellular manufacturing: a genetic algorithm. *Production Planning & Control, 9*(2), 155-166.

Jeon, G., & Leep, H. R. (2006). Forming part families by using genetic algorithm and designing machine cells under demand changes. *Computers & operations research, 33*(1), 263-283.

Moussa, S. E., & Kamel, M. (1998). A part-machine assignment algorithm for cellular manufacturing with machine capacity constraints. *Computers & industrial engineering, 35*(3), 483-486.

Muruganandam, A., Prabhaharan, G., Asokan, P., & Baskaran, V. (2005). A memetic algorithm approach to the cell formation problem. *The International Journal of Advanced Manufacturing Technology, 25*(9-10), 988-997.

Nsakanda, A. L., Diaby, M., & Price, W. L. (2006). Hybrid genetic approach for solving large-scale capacitated cell formation problems with multiple routings. *European Journal of Operational Research, 171*(3), 1051-1070. doi: 10.1016/j.ejor.2005.01.017

Papaioannou, G., & Wilson, J. M. (2010). The evolution of cell formation problem methodologies based on recent studies (1997–2008): Review and directions for future research. *European Journal of Operational Research, 206*(3), 509-521.

Paydar, M. M., Mahdavi, I., Sharafuddin, I., & Solimanpur, M. (2010). Applying simulated annealing for designing cellular manufacturing systems using MDmTSP. *Computers & industrial engineering, 59*(4), 929-936.

Paydar, M. M., Saidi–Mehrabad, M., & Kia, R. (2013). Designing a new integrated model for dynamic cellular manufacturing systems with production planning and intra–cell layout. *International Journal of Applied Decision Sciences, 6*(2), 117-143.

Rafiee, K., Rabbani, M., Rafiei, H., & Rahimi-Vahed, A. (2011). A new approach towards integrated cell formation and inventory lot sizing in an unreliable cellular manufacturing system. *Applied Mathematical Modelling, 35*(4), 1810-1819.

Renna, P., & Ambrico, M. (2015). Design and reconfiguration models for dynamic cellular manufacturing to handle market changes. *International Journal of Computer Integrated Manufacturing, 28*(2), 170-186.

Rogers, D. F., & Kulkarni, S. S. (2005). Optimal bivariate clustering and a genetic algorithm with an application in cellular manufacturing. *European Journal of Operational Research, 160*(2), 423-444.

Safaei, N., Saidi-Mehrabad, M., & Jabal-Ameli, M. (2008). A hybrid simulated annealing for solving an extended model of dynamic cellular manufacturing system. *European Journal of Operational Research, 185*(2), 563-592. doi: 10.1016/j.ejor.2006.12.058

Safaei, N., & Tavakkoli-Moghaddam, R. (2009). Integrated multi-period cell formation and subcontracting production planning in dynamic cellular manufacturing systems. *International Journal of Production Economics, 120*(2), 301-314.

Seifoddini, H., & Djassemi, M. (1993). A dynamic part assignment procedure in machine cell formation. *Computers & industrial engineering, 25*(1), 473-476.

Seifoddini, H., & Djassemi, M. (1996). Improving the performance of cellular manufacturing by a dynamic part assignment approach. *Computers & industrial engineering, 30*(4), 719-726.

Sofianopoulou, S. (1999). Manufacturing cells design with alternative process plans and/or replicate machines. *International Journal of Production Research, 37*(3), 707-720.

Solimanpur, M., Vrat, P., & Shankar, R. (2004). Ant colony optimization algorithm to the inter-cell layout problem in cellular manufacturing. *European Journal of Operational Research, 157*(3), 592-606.

Sullivan, W. G., McDonald, T. N., & Van Aken, E. M. (2002). Equipment replacement decisions and lean manufacturing. *Robotics and Computer-Integrated Manufacturing, 18*(3), 255-265.

Tavakkoli-Moghaddam, R., Aryanezhad, M.-B., Safaei, N., Vasei, M., & Azaron, A. (2007). A new approach for the cellular manufacturing problem in fuzzy dynamic conditions by a genetic algorithm. *Journal of Intelligent and Fuzzy Systems, 18*(4), 363-376.

Tavakkoli-Moghaddam, R., Javadian, N., Javadi, B., & Safaei, N. (2007). Design of a facility layout problem in cellular manufacturing systems with stochastic demands. *Applied Mathematics and Computation, 184*(2), 721-728.

Tavakkoli-Moghaddam, R., Javadian, N., Khorrami, A., & Gholipour-Kanani, Y. (2010). Design of a scatter search method for a novel multi-criteria group scheduling problem in a cellular manufacturing system. *Expert Systems with Applications, 37*(3), 2661-2669. doi: 10.1016/j.eswa.2009.08.012

Tompkins, J., White, J., Bozer, Y., & Tanchoco, J. (2003). Facilities planning. 2003: Wiley, New York.

Wang, T.-Y., Wu, K.-B., & Liu, Y. (2001). A simulated annealing algorithm for facility layout problems under variable demand in cellular manufacturing systems. *Computers in Industry, 46*(2), 181-188.

Won, Y., & Currie, K. (2007). Fuzzy ART/RRR-RSS: a two-phase neural network algorithm for part-machine grouping in cellular manufacturing. *International Journal of Production Research, 45*(9), 2073-2104.

Xambre, A. R., & Vilarinho, P. M. (2003). A simulated annealing approach for manufacturing cell formation with multiple identical machines. *European Journal of Operational Research, 151*(2), 434-446.

Yu, J., & Sarker, B. R. (2003). Directional decomposition heuristic for a linear machine-cell location problem. *European Journal of Operational Research, 149*(1), 142-184.

Yu, J., & Sarker, B. R. (2006). A directional decomposition heuristic for one-dimensional, non-equidistant machine-cell location problems. *Computers & Operations Research, 33*(1), 64-92.