A water flow-like algorithm for cellular manufacturing design and layout

Chin-Chih Chang[1], Tai-Hsi Wu[2]

[1]*Department of Information Management, Jen-Teh Junior College of Medicine, Nursing and Management*

Miaoli, Taiwan

[1]chinju.chang@gmail.com

[2]*Department of Business Administration, National Taipei University*

Taipei, Taiwan

[2]taiwu@mail.ntpu.edu.tw

***Abstract*— The water flow-like algorithm (WFA) is a novel heuristic approach that has the features of multiple and dynamic numbers of solution agents, and its mimicking of the natural behavior of water flowing from higher to lower levels coincides exactly with the process of searching for optimal solutions. This paper adopts the WFA logic and designs a heuristic algorithm for solving the cell formation and cell layout problems simultaneously. When compared with the mathematical programming approach which took one hour to solve problems, the proposed algorithm is able to produce optimal solutions in less than one second. These show that the proposed approach is extremely effective and efficient.**

***Keywords*—Cell formation, Cell layout; Alternative process routings, Water flow-like algorithm**

1. INTRODUCTION

Cellular manufacturing is a manufacturing philosophy in which similar parts are identified and grouped into part families; meanwhile, machines are grouped into machine cells to take benefits such as reductions in set-up times, work-in-progress inventory, throughput times and material handling costs, simplified scheduling and improved quality[1]. Although cellular manufacturing may provide great benefits, the CFP with considerations of cell formation and cell layout simultaneously are NP-hard combinational problems [2]. Many models and solution approaches have been developed to treat these problems sequentially or independently. These approaches can be classified into three main categories: mathematical programming (MP) models [3-6], heuristic/meta-heuristic solution algorithms [7-10], and similarity coefficient methods (SCM) [11-13]. Among the aforementioned heuristics/meta-heuristics, the simulated annealing (SA) and the tabu search (TS) are single-solution-agent-based algorithms. A single agent-based heuristic algorithm searches the solution space step by step through the usage of systematic or random neighborhood exploration. The genetic algorithms (GAs), however, belong to the group of multiple-solution-agent-based algorithms, which starts the optimization with a set of possible solutions, not only one possible solution. Yang and Wang [14] stated that neither the single nor the multiple agent method is agile enough to conduct an efficient and effective solution search. They hence present a water flow-like algorithm (WFA) to overcome this deficiency,

The WFA method is a dynamic-solution-agent-based algorithm. The design of the WFA method was inspired by water flowing from higher to lower levels, where a flow will split into multiple subflows when it moves through uneven terrains. Conversely, subflows merge when they meet at the same level. Water flow ceases and stagnates at the lowest depressions, when momentum cannot expel water out of the depressions. Water flowing is analogous to problem solving. A flow is regarded as a solution agent, the solution space of a problem is the geographical terrain, and the altitude of a flow represents the objective function value. Since the number of flows dynamically changes in this method, it is an agent population-varying method.

CFP has received the attention of researchers for more than three decades. However, very few of them have tried to investigate both cell formation and the cell layout simultaneously with considerations of process sequences, alternative

routings and product volume. It is important and more practical to integrate the abovementioned factors simultaneously in the design of cellular manufacturing system.

In this paper, we use the WFA to solve cell formation and cell layout simultaneously with process sequences, alternative routings and product volume considerations. The approach is illustrated by an example and compared with the mathematical programming approach.

2. PROBLEM DEFINITION

2.1. Cell Formation

In a simple CFP, cell formation in a given 0-1 machine-part incidence matrix involves rearrangement of rows and columns of the matrix to create part families and machines cells, in which the cellular movement can be minimized and the utilization of the machines within a cell can be maximized. Two matrices shown in Figure 1 are used to illustrate the concept. Fig. 1(a) is an initial matrix where no blocks can be observed directly. After rearrangement of rows and columns, two blocks can be obtained along the diagonal of the solution matrix in Fig. 1(b). For those 1's outside the diagonal blocks, they are called "exceptional elements"; while those 0's inside the diagonal blocks are called "voids".

(a)

	P1	P2	P3	P4	P5
M1	0	1	0	1	0
M2	0	0	1	0	1
M3	1	1	0	1	0
M4	1	0	1	0	1
M5	0	1	0	0	0

(b)

	P1	P3	P5	P2	P4
M2	0	1	1	0	0
M4	1	1	1	0	0
M1	0	0	0	1	1
M3	1	0	0	1	1
M5	0	0	0	1	0

Fig. 1 Rearrangement of rows and columns of matrix to create cells: (a) initial matrix and (b) matrix after rearrangement.

When parts have alternative process routings (APR) is called the generalized CFP. Such as the case shown in Table 1, part #1 has two process routings R1 and R2. Kusiak [15] first described the problem and presented an integer-programming model to solve the problem. While introducing APR to CFP, the grouping of parts can be more effective due to the flexibility of the routes; however, it leads to a more complex problem than the simple CFP. Under this circumstance, not only the formation of part families and machine cells must be determined but also the selection of routings for each part has to be determined to achieve decision objectives such as the minimization of intercellular movement. For instance, Table 2 provides a feasible solution to the sample problem of Table 1 which has three cells with machine groupings for each cell as Cell 1: (M3, M7); Cell 2: (M2, M4, M6); and Cell 3: (M1, M5, M8).

TABLE 1 Initial machine-part matrix where alternative process routings are allowed

PN	P1		P2		P3		P4		P5		P6	
PV	150		95		130		80		95		135	
RN	R1	R2	R1	R2	R1	R2	R1	R2	R1	R2	R1	R2
M1	1*			1		1			1			1
M2	2	2			1				2		1	2
M3							1	1	3	1		
M4	3	1									2	3
M5				2		2						
M6		3									3	
M7			1		2	3	2	2		2		
M8			2	3						3		

PN: Part Number; PV: Production Volume; RN: Routing Number; * Process Sequence

TABLE 2 Final machine-part matrix of Table 1

PN	P4	P5	P1	P6	P2	P3
PV	80	95	150	135	95	130
RN	R2	R2	R2	R1	R2	R2
M3	1	1				
M7	2	2				3
M2			2	1		
M4			1	2		
M6			3	3		
M1					1	1
M5					2	2
M8		3			3	

2.2. Cellular Layout

In CMS, different cellular layout type and cellular layout sequence will affect the inter-cell move distance (ICMD). They are described as follows.

(a) Inter-cell move distance

There are two popular methods for measuring ICMD between a pair of cells l and l' (Tam and Li, 1991). They are Cartesian method (Eq. (1)) and Manhattan method (Eq. (2)).

$$D_{l,l'} = \left[\left(X_{l'} - X_l \right)^2 + \left(Y_{l'} - Y_l \right)^2 \right]^{1/2}, \quad (1)$$

$$D_{l,l'} = \left| X_{l'} - X_l \right| + \left| Y_{l'} - Y_l \right|, \quad (2)$$

where (X_l, Y_l) and $(X_{l'}, Y_{l'})$ are the coordinates of the measuring points of cells l and l'.

In terms of measuring points, we can use either: (a) the centroid of a cell site or (b) the nearest point between adjacent cells. In this thesis, the Cartesian method was chosen and the centroid of a cell site will be used for calculating ICMD. Thus, the ICMD between cells 1 and 2 in Fig. 2(a) is equal to $1=\left[(1-1)^2+(2-1)^2\right]^{1/2}$.

(b) Effects of cellular layout type

Different cellular layout types will result in different ICMD. Figure 3.5 shows that the ICMD between cells 1 and 3 in Fig. 2(a) will be twice the distance moved between cells 1 and 3 in Fig. 2(b). Hence, the cellular layout type is an important issue in CMS design.

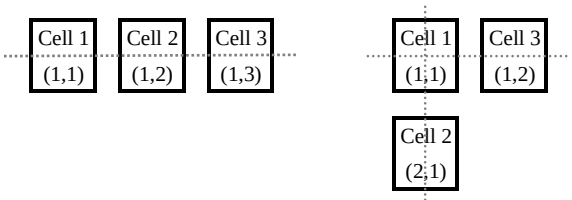

Fig. 2 Two typical cellular layouts (NC=3): (a) linear single-row layout (b) linear double-row layout

(c) Effects of cellular layout sequence

We present an example to illustrate the effects of cellular layout sequence. If the NC is equal to three and a linear single-row layout is considered as shown in Fig. 2(a), then ICMD between cells 1 and 3 will be twice the distance moved between cells 1 and 2 or between cells 2 and 3. When a linear double-row layout is considered as shown in Fig. 2(b), the corresponding ICMD between cells 2 and 3 will be $\sqrt{2}$ times the distance moved between cells 1 and 2 or between cells 1 and 3. Hence, the cellular layout is an important issue in CMS design.

3. MATHEMATICAL MODEL

The decision objective is mainly to solve the cell formation and the cell layout simultaneously in terms of minimization of intercellular movement distance unit. The 0-1 integer programming model is given below, and the notations are introduced first.

A. Notations

ICMD : The distance of inter-cell movements required by the system
m : Number of machines
p : Number of parts
NC : Number of cells
V_i : Production volume for part i
Q_i : Number of routings for part i
U_m : Maximum number of machines in each cell
L_m : Minimum Number of machines in each cell
K_{ij} : Number of operations in routing j of part i
$u_{ij}^{(a)}$: Index for machines which belongs to the a-th operation of part i along route j
$D_{l,l'}$: The distance between cell l and l'
Y_{kl} : 1, if machine k locates in cell l; 0, otherwise
Z_{ij} : 1, if routing j of part i selected; 0, otherwise
$X_{ijklk'l'}$: 1, if routing j of part i is selected; machine k locates in cell l and machine k' locate in cell l'; 0, otherwise

B. Mathematical model

$$\text{Min } ICMD=\sum_{i=1}^{p}\sum_{j=1}^{Q_i}\sum_{k=1}^{K_{ij}-1}\sum_{l=1}^{NC}\sum_{l'=1}^{NC}X_{iju_{ij}^{(k)}lu_{ij}^{(k+1)}l'}V_iD_{l,l'} \quad (3)$$

Subject to:

$$\sum_{j=1}^{Q_i}Z_{ij}=1,\ \forall i \quad (4)$$

$$L_m\le\sum_{k=1}^{m}Y_{kl}\le U_m\ ,\ \forall l \quad (5)$$

$$\sum_{l=1}^{NC}Y_{kl}=1,\ \forall k \quad (6)$$

$$X_{ijklk'l'}\le Z_{ij}\ ,\ \forall i,j,k,k',l,l' \quad (7)$$

$$X_{ijklk'l'}\le Y_{kl}\ ,\ \forall i,j,k,k',l,l' \quad (8)$$

$$X_{ijklk'l'}\le Y_{k'l'}\ ,\ \forall i,j,k,k',l,l' \quad (9)$$

$$Z_{ij}+Y_{kl}+Y_{k'l'}-X_{ijklk'l'}\le 2\ ,\forall i,j,k,k',l,l' \quad (10)$$

$$Y_{kl},Y_{k'l'},Z_{ij},X_{ijklk'l'}\in\{0,1\}\quad \forall i,j,k,k',l,l' \quad (11)$$

In the above model, Eq. (3) is the objective function, which seeks the minimization of total distance of intercellular movement. Eq. (4) indicates that only one process routing will be assigned to each part. Eq. (5) imposes the upper and lower limits of the cell size. Eq. (6) restricts that each machine will be assigned to exactly one cell. Eqs. (7) to (9) ensure that if one of the primary binary variables takes has a zero value, then their corresponding new variables will take a zero value as well. Eq. (10) ensures that if all of the primary variables take unit values, then their

corresponding new variables take unit values as well. Eq. (11) indicates that Y_{kl}, $Y_{k'l'}$ Z_{ij} and $X_{ijklk'l'}$ are 0–1 binary decision variables.

4. PROPOSED ALGORITHM

The design of the WFA (Yang and Wang [14]) was inspired by the natural behavior of water flowing from higher to lower levels. On the earth's surface, a flow will split into multiple sub-flows when rugged terrains are traversed. Sub-flows, however, will merge when they arrive at the same location. Governed by gravity and driven by fluid momentum, flows can run to higher levels or run over bumps to navigate various terrains. Water flow will cease and stagnate at the locally or globally lowest depression; when the momentum left cannot expel the water out of the depression, it will stagnate at its current location. No movement is allowed until other flows merge with it or until the water evaporates into the atmosphere. When the evaporated water accumulates to some extent, it will return to the ground as several new downpour flows, such that rainfall occurs occasionally. As the solution space of a problem can be mapped to the geographical terrain, and the objective value is mapped to the altitude, each flow can then be regarded as a solution agent. Water moving to a lower position can be considered as a solution searching for the optima. Thus, the solution search process has been modeled as water flow.

The WFA algorithm consists of four primary operations: (1) flow splitting and moving, (2) flow merging, (3) water evaporation, and (4) precipitation. The pseudo-code for the general procedure for implementing the WFA is shown in Fig. 3.

```
WFA_Algorithm ( )
{
   Generate an initial solution.
   WHILE(stop criterion is false)
   {
         Flow splitting and moving.
         Flow merging.
         Water evaporation.
         IF (rainfall required)
         {
              Precipitation.
              Flow Merging.
         }
         IF (new best solution found)
              Update best solution record.
   }
}
```

Fig. 3 A generic framework for WFACF

4.1. Flow splitting and moving operation

It is assumed that there is only one water flow in the beginning of the WFA, and that its location is randomly generated. Driven by fluid momentum and potential energy, the flow starts to move to new locations to explore the solution space for new and better solutions. Fig. 4 demonstrates the splitting and moving operation for searching neighborhood solutions. As flow i splits into subflows, the number of subflows n_i is obtained based on Eq. (12):

$$n_i = \max\left\{1,\ \text{int}\left(\frac{W_i V_i}{T_m}\right)\right\}, \tag{12}$$

where W_i is the mass of flow i, V_i is the velocity of flow i and T_m is the base momentum.

The mutation strategy (Fig. 5) is implemented to determine the rough directions for k subflows; that is, the locations of X_{i1}, X_{i2}, …, X_{ik} can be identified. The insertion-move is then performed to find the best neighborhood solution around X_{i1}; that is, the X^*_{i1}. This is repeated until the best neighborhood solution for each of the subflows has been found. For each iteration, these newly generated subflows may merge with others sharing the same location, proceed in a single stream, split further into more subflows at later iterations, or stagnate in the current location until the stopping criteria of the algorithm is met.

When the flow is split into sub-flows, its original mass has to be accordingly distributed to sub-flows based on Eq. (13):

$$w_{ik} = \left(\frac{f(X_{ik})}{\sum_{k=1}^{n_i} f(X_{ik})} \right) W_i , \quad (13)$$

where $f(X_{ik})$ is the objective value of solution X_{ik}.

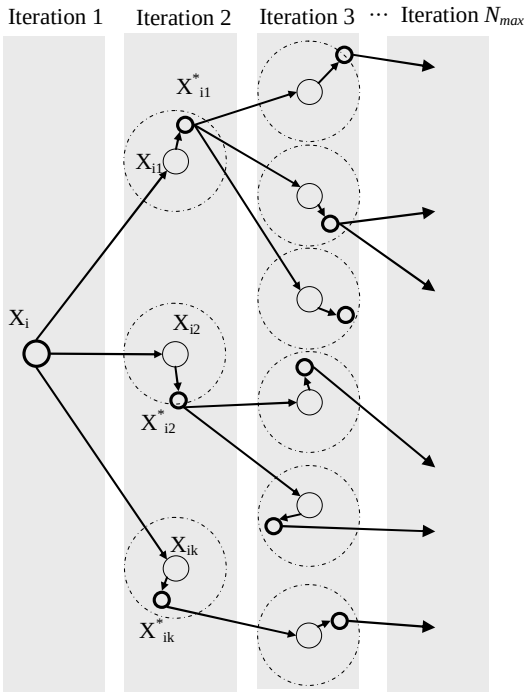

Fig. 4 Proposed flow splitting and moving operation for searching neighborhood solutions

```
Mutation_strategy ( β )
{
    Let the current solution (S) equal to the current best solution (S*).
     FOR each machine DO
    {
        Generate a random number  r ∈ U(0,1) .
        IF ( r > β )
            Exchange machine with any cells other than the current one.
        ELSE
            Stay machine in the current cell.
    }
}
```

Fig. 5 Pseudo code of mutation strategy

The velocity of each sub-flow is computed from the equation of energy conservation. μ_{ik}, the velocity of sub-flow k split from flow i, is:

$$\mu_{ik} = \begin{cases} \sqrt{V_i^2 + 2g\delta_{ik}}, & if \quad V_i^2 + 2g\delta_{ik} > 0 \\ 0 \quad , & otherwise \end{cases} \quad (14)$$

where g is the gravitational acceleration, and δ_{ik} is the altitude drop from flow i to its sub-flow k; that is, the improvement of objective value moving from current solution i to its neighborhood solution k. When $V_i^2 + 2g\delta_{ik} < 0$, the momentum delivered to sub-flow k has been used up, implying that this sub-flow will stagnate in its current location (e.g. the solution is trapped in local optima) without splitting and movement. Such stagnant flow will gradually evaporate into the atmosphere, returning to the ground by precipitation later on.

At the end of the splitting and moving operation, the original flow becomes discarded because sub-flows have been generated. Information regarding the current number of sub-flows and solutions sets will then be recorded.

4.2. Flow-merging operation

When more than two flows move to the same location, they will merge into one flow with a bigger mass and momentum. Whether a flow shares the same location with others in the WFA is thus systematically examined. If a flow does share the same location, the latter flow is then merged into the former one. Assuming that flows i and j share the same location, then flow j will be deleted and the mass and velocity of flow i will be updated as follows:

$$W_i = W_i + W_j \quad (15)$$

$$V_i = \frac{W_i V_i + W_j V_j}{W_i + W_j} \quad (16)$$

Using the flow-merging operation, the WFA reduces the number of solution agents when multiple agents result in the same objective value in order to avoid redundant searches.

4.3. Water evaporation operation

It is natural for water to evaporate and return to the ground through precipitation after possible movement from its original location. Water evaporation and precipitation coincide with the "escaping from local optima" mechanism that many heuristic algorithms nowadays use to avoid being trapped and to explore more solution spaces.

Each flow in the WFA is subject to water evaporation, where part of the water evaporates into the atmosphere. We define an velocity-based evaporation that is conversely related to improvement in velocity, such that flows with smaller velocities should evaporate more quickly than those with larger velocities. The formula is presented below:

$$W_i = (1 - \rho_i) W_i \quad (17)$$

$$\text{where } \rho_i = \begin{cases} 1, & if\ \mu_{ik}=0 \\ 0, & if\ \dfrac{\mu_{ik}}{V_i} \geq 1 \\ 1-\dfrac{\mu_{ik}}{V_i}, & if\ 0<\dfrac{\mu_{ik}}{V_i}<1 \end{cases}$$

4.4. Precipitation operation

In this study, when water vapor accumulates to half of its original total mass, W_0, it will return to the ground in some form such as rain. The formula is presented below:

$$W_i^{'} = \frac{W_i}{\sum_{k=1}^{N} W_k}\left(W_0 - \sum_{k=1}^{N} W_k\right) \quad (18)$$

4.5. Proposed Algorithm (WFACF)

The framework of the proposed WFA procedure, namely WFACF, is given in Fig. 6 and is described in detail below.

Step 1. Read initial solution.
Step 2. Initial WFACF parameter setting.
Step 3. If *counter_iter* $\leq N_{max}$, repeat Steps 4 to 16; otherwise, go to Step 17.
Step 4. For each flow, execute Steps 5 to 16.
Step 5. Calculate the number of subflows based on Eq. (12).
Step 6. Flow splitting and moving through ***mutation strategy*** and ***insertion-move operation***.
Step 7. Check whether the new best solution is found. If yes, update best solution.
Step 8. Calculate mass and velocity based on Eqs. (13) and (14).
Step 9. Merge flows with the same objective values and update the resulting mass and velocity based on Eqs. (15) and (16).
Step 10. Update the total number of water flow.
Step 11. Perform evaporation operation and update the resulting mass for each water flow based on Eq. (17).
Step 12. Check whether precipitation condition is met. If yes, perform Steps 13, 14, and 15; otherwise, go to Step 16.
Step 13. Perform ***mutation strategy*** to the current best solution to generate new solutions deviated from the current ones.
Step 14. Distribute mass to flows poured based on Eq. (18).
Step 15. Check whether the new solution has the same objective value. If yes, merge it and update the resulting mass and velocity based on Eqs. (15) and (16), then update the total number of water flow.
Step 16. Let *counter_iter* = *counter_iter* + 1, go to Step 3.
Step 17. Report the best solutions so far, and stop the algorithm.

Note that in the mutation strategy, a threshold probability value (β) set at 0.8 implies that each machine has a 20% probability of being assigned to other cells. In the WFACF procedure, the mutation strategy is used in Steps 6 and 13 with different threshold probability values (β): 0.8 in Step 6 and 0.5 in Step 13. The main purpose of Step 6 is to find some neighborhoods of the current solution, thus the probability of being assigned to other cells is set at a comparatively low value. On the other hand, the purpose of Step 13 is to explore solutions of unvisited regions through the precipitation operation; thus, it becomes necessary to increase the probability of being assigned to other cells to find solutions more deviated from the current best.

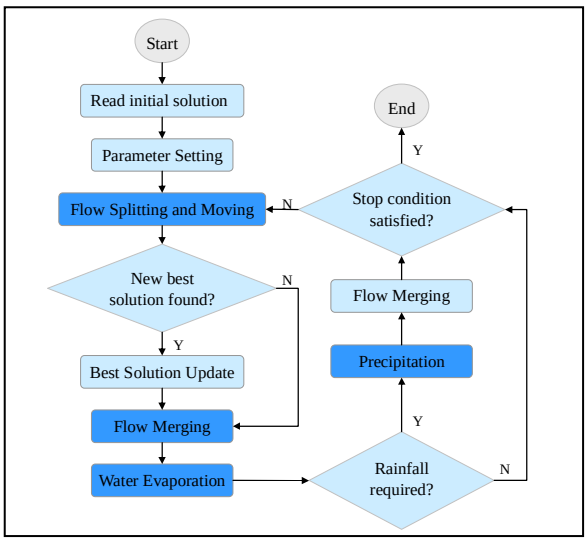

Fig. 6 Framework of the proposed WFACF

5. Research Results

In order to demonstrate the effectiveness of our proposed model and methodology for cell formation and cell layout problems, we apply our proposed approach to a numerical example presented by Kazerooni et al. [15] and compared the optimal solutions obtained by the branch and bound (B & B) algorithm with the LINGO 8.0 software. This example consists of 40 parts and 24 machines. A maximum of five machines is allowed in each cell. The proposed algorithm was

coded in C++ using Microsoft Visual Studio 6.0 and implemented on a Intel(R) 1.66GHz PC with 1GB RAM.

The computational results are summarized and compared in Table 3. The results show that the proposed WFACF is able to achieve global optimum in less than nine seconds, which indicates that our proposed approach is extremely effective and efficient. Furthermore, we observed that the explicit preference for linear double-row layout over linear single-row layout based on their solution quality.

TABLE 3 Comparison of Lingo and our WFACF

Method	Lingo(B & B)		Proposed method	
Layout type	*linear single-row*	*linear double-row*	*linear single-row*	*linear double-row*
NC	6	6	6	6
ICMD	1825	1633	1825	1633
CPU (s)	2170	4171	3.18	3.43

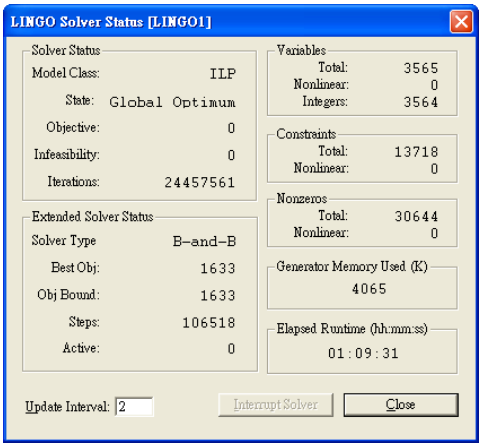

Fig. 7 Lingo solver status for linear double-row layout

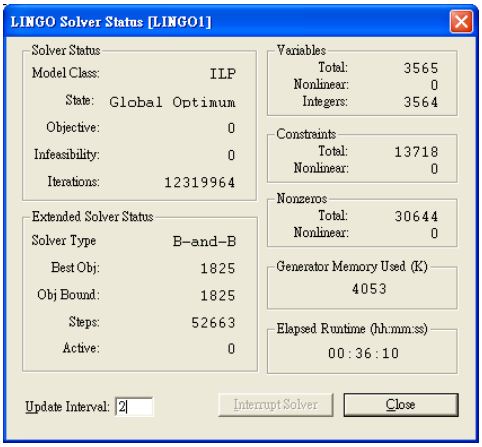

Fig. 8 Lingo solver status for linear single-row layout

6. CONCLUSIONS

Very limited amount of articles have simultaneously considered the issues of production volume, production sequence, alternative process routings and cell layout in CFP so far. Accounting for these factors makes the CFP complex but more realistic. In this paper, a mathematical model has been proposed to tackle the cell formation and cell layout simultaneously with consideration of both alternative routing and production volume. Due to the combinatorial nature of this model, a WFA algorithm has been designed for solving this problem.

The results of the proposed method are compared with the optimal solutions obtained by the LINGO 8.0 software. The comparisons show that the proposed method offers good solutions for the CFP considering production volume, production sequence, alternative process routings and cell layout. More test problems from the open literature are to be tested in the near future to fully confirm the efficiency and effectiveness of the proposed approach.

REFERENCES

[1] U. Wemmerlov, and N. Hyer, “Research issues in cellular manufacturing,” *International Journal of Production Research*, vol. 25, pp. 413–431, 1987.

[2] A. Ballakur and H.J. Steudel, “A within cell utilization based heuristic for designing cellular manufacturing systems,” *International Journal of Production Research*, vol. 25, pp. 639–655, 1987.

[3] F. Boctor, “A linear formulation of the machine-part cell formation problem,” *International Journal of Production Research*, vol. 29, pp. 343–356, 1991.

[4] J. Wang, “Formation of machine cells and part families in cellular manufacturing systems using a linear assignment algorithm,” *Automatica*. vol. 39, pp. 1607–1615, 2003

[5] Z. Albadawi, H. A. Bashir and M. Chen, “A mathematical approach for the formation of manufacturing cells,” *Computers and Industrial Engineering*, vol. 48, pp. 3–21, 2005.

[6] M.S. Jabal Ameli, J. Arkat, and F. Barzinpour, “Modelling the effects of machine breakdowns in the generalized cell

formation problem," *International Journal of Advanced Manufacturing Technology*, vol. 39, pp.838–850.

[7] D. Lei and Z. Wu, "Tabu search approach based on a similarity coefficient for cell formation in generalized group technology," *International Journal of Production Research*, vol. 43, pp. 4035–4047, 2005.

[8] T. H. Wu, C. C. Chang, and S. H. Chung, "A simulated annealing algorithm for manufacturing cell formation problems," *Expert Systems with Applications*, vol. 34, pp. 1609–1617, 2008.

[9] A. Tariq, I. Hussain, and A. Ghafoor, "A hybrid genetic algorithm for machine-part grouping," *Computers and Industrial Engineering*, vol. 56, pp. 347–356, 2009.

[10] T. H. Wu, S.H. Chung, and C. C. Chang, "A water flow-like algorithm for manufacturing cell formation problems," *European Journal of Operational Research*, vol. 205, pp. 346–360, 2010.

[11] J. McAuley, "Machine grouping for efficient production," *Production Engineer*, vol. 52, pp. 53–57, 1972.

[12] G. J. Nair and T. T. Narendran, "CASE: a clustering algorithm for cell formation with sequence data," *International Journal of Production Research*, vol. 36, pp. 157–179, 1998.

[13] Y. Yin and k. Yasuda, "Similarity coefficient methods applied to the cell formation problem: a comparative investigation," *Computers and Industrial Engineering*, vol. 48, pp.471–489, 2005.

[14] F. C. Yang, and Y. P. Wang, "Water flow-like algorithm for object grouping problems," *Journal of the Chinese Institute of Industrial Engineers*, vol. 24, pp. 475–488, 2007.

[15] M. Kazerooni, H. S. Luong and K. Abhary, "A genetic algorithm based cell design considering alternative routing," *Computer integrated manufacturing system*, vol. 10, pp. 93–107, 1997.