African Journal of Business Management Vol. 6(15), pp. 5456-5466, 18 April, 2012
Available online at http://www.academicjournals.org/AJBM
DOI: 10.5897/AJBM11.2715
ISSN 1993-8233 ©2012 Academic Journals

Full Length Research Paper

Developing petri net model and meta-heuristic algorithms for cyclic scheduling in 2-machine robotic cells

Mohammad Fathian[1], Isa Nakhai Kamalabadi[2], Mehdi Heydari[1], Hiwa Farughi[1]* and Foruzan Naseri[3]

[1]Industrial Engineering Department, Iran University of Science and Technology, Tehran, Iran.
[2]Industrial Engineering Department, Tarbiat Modares University, Tehran, Iran.
[3]Industrial Engineering Department, University of Kurdistan, Sanandaj, Iran.

Accepted 14 December, 2011

In this paper, the cyclic scheduling problems in 2-machine robotic cells have been studied. We investigated the timed Petri network graph for modeling the part sequencing and the optimal robot moves sequence in robotic manufacturing cells. The robotic manufacturing cell considered in this study has two identical machines and one single gripper robot. Also, we have assumed that the manufacturing cell is capable of producing identical and different parts. The main objective of this study is to minimize the cycle time. To solve this problem, we have proposed two meta-heuristic algorithms called particle swarm optimization (PSO) and simulated annealing (SA) and compared the obtained results with the exact solutions by LINGO. Also the complexity of the proposed model has been analyzed.

Key words: Cycle time, 2-machine robotic cell, Petri net, part sequencing, robot moves sequence, cyclic scheduling, meta-heuristic algorithms.

INTRODUCTION

In the present competitive world, time is an important and determining factor in industries. Along with technological progress in industries and organizations, managers' decision-making and their organizational activities, and strategies have become increasingly complex. One of these strategies is the development of automation in industries and manufacturing organizations, which involves the use of mechanical and programmable devices called robots for moving parts between the different stations. By establishing machines in cellular layout and using robots for automating the process, managers try to reduce the production time in order to increase the effectiveness of the production line, and to increase the productivity output in robotic manufacturing cells. In the last few years, researchers have been concerned with optimizing the robot move sequence in order to reduce production time in robotic manufacturing cells and many studies have been done in this regard.

The study of Sethi et al. (1989) is considered as the beginning point of the robotic cell scheduling literature. They discussed on minimizing the cycle time in the single machine robotic cell. Sethi et al. (1992) proved that in buffer-less single-gripper two-machine robotic cells producing single part-type and having identical robot travel times between adjacent machines and identical load/unload times, a 1-unit cycle provides the minimum per unit cycle time in the class of all solutions, cyclic or otherwise. For three machine case, Crama and van de Klundert (1999), and Brauner and Finke (1999) shown that the best 1-unit cycle is optimal solution for the class of all cyclic solutions. Hall et al. (1997) and Hall et al. (1998) considered the computational complexity of the multiple-type parts three-machine robotic cell problem under various robot movement policies. Kamalabadi et al.

*Corresponding author. Email: h_farughi@iust.ac.ir .

(2008) provided a new solution for the cyclic multiple parts three-machine robotic cell. They also considered the minimizing of cycle time in a blocking flow shop cell (Kamalabadi, 1996; Kamalabadi et al., 2000). This problem is studied for no-wait robotic cells too. For example, Agnetis (2000) found an optimal part schedule for no-wait robotic cells with three and two machines. Agnetis and pacciarelli (2000) studied part scheduling problem for no-wait robotic cells, and found the complexity of the problem. Crama et al. (1999) and Crama et al. (2000) studied flow-shop scheduling problems, models for such problems, and complexity of these problems. Dawande et al. (2005) reviewed the recent developments in robotic cells and, provided a classification scheme for robotic cells scheduling problem. Some other special cases have been studied such as: Drobouchevitch et al. (2006) provided a model for cyclic production in a dual-gripper robotic cell. Deineko et al. (2005) studied the special case of two machine flexible robotic cells that the first machine performs one operation, and the second machine processes K operations step by step. Akturk et al. (2000) studied robotic cell scheduling with operational flexibility. Gultekin et al. (2006) studied robotic cell scheduling problem with tooling constraints for a two-machine robotic cell where some operations can only be processed on the first machine and some others can only be processed on the second machine and the remaining can be processed on both machines. Gultekin et al. (2007) considered a flexible manufacturing robotic cell with identical parts in which machines are able to do different operations and the operation time is not system parameter and is variable. They proposed a lower bound for 1-unit cycles and 2-unit cycles. Sriskandarajah et al. (1998) classified the part sequence problems associated with different robot movement policies; in this paper, a robot movement policy is considered, which its part scheduling problem is NP-Hard, and Bagchi et al. (2006) proposed to solve this problem, by a heuristic or meta-heuristic.

It is obvious that the two fundamental problems in robotic cell scheduling are part sequencing and optimizing the robot move sequence. If the cell is meant to produce identical parts, the scheduling problem will depend on finding the optimal robot move sequence. In this paper, we have defined a new cycle for robot move sequence in a 2-machine robotic manufacturing cell, which is development of existing robot motion cycles. Our purpose is to obtain the optimal cycle time by determining the optimal parts entry to the cell. For the modeling of this problem, timed Petri network was used. Subsequently in this paper, assumptions, concepts and the robot move sequence for the proposed new cycle are introduced and the cycle time is calculated. Thereafter, concepts and relations in a Petri network are described. Then, the proposed motion cycle is described in full detail according to Petri net model. At first, the mathematical model for the problem of producing identical parts in the production cycle is obtained, and then the model is generalized to the problem of producing different parts. Thereby, by determining the optimal part sequencing for the proposed cycle the minimum cycle time is obtained. Next, the particle swarm optimization and simulated annealing algorithms with related parameter setting are described. After that, the experimental results are analyzed and concluded. Finally, conclusion of the paper is presented.

DEFINITION OF THE PROBLEM AND INTRODUCTION OF THE PROPOSED MOTION CYCLE

In robotic manufacturing cell scheduling, the major problem is how to determine the sequence of robot moves and order the parts entry in a cell that produces different parts. In past studies about 2-machine robotic cells, $S_{12}S_{21}, S_2, S_1$ motion cycles have been introduced (Sethi et al., 1989), and the robot move sequence and the order of parts entry in 2-machine robotic cells have been studied (Sethi et al., 1992). In 3-Machine robotic cells, the sequence of robot moves and parts entry to the cell has been investigated for both the same parts and different parts manufacturing cells (Crama et al., 1997, 1999). In this paper, we have defined a new cycle for two-machine robotic cells and have considered the problem of different parts entry sequencing to obtain the optimum cycle time by a timed Petri net model. In most studies on scheduling 2-machine robotic cells, the problem of flow shop has been considered. Thereby, each part is processed on the first machine and then conveyed by the robot to the second machine to be processed on. Indeed, each part must be processed on both machines. It is also assumed that the two machines are identical, that is, the processing time of a same operation is the same on both machines. In addition, the robot's movement time between any two consecutive locations is the same and it is additive between different locations. Also, the robot's loading and unloading time in all conditions is the same and the robot has a linear movement in the cycle.

Definition 1

Activity (A_{ij}) signifies the robot's conveying a part from location i to location j.

Definition 2

A n-unit cycle means that the robot has entered n parts into the cycle and for doing all the required processes on n parts, each activity has been repeated n times. At the end of each cycle, n part should be taken out of the cell by robot. Also, the beginning and the end mode of the n-unit cycle should be same.

Definition 3

The n-unit cycle time is defined as the time needed to produce n parts in a cyclic process that a robot starts from the initial state and moves in a specific sequence, so that the necessary operations for producing n parts is performed and then the robot goes on the initial position. Also, if we assume that the machines are flexible; in other words, if each machine is capable of performing all operations on each part so that every part is processed completely by a single

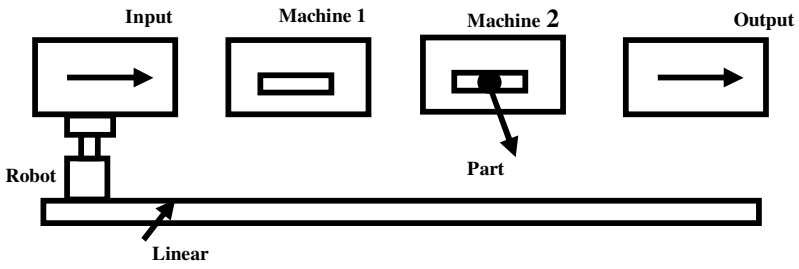

Figure 1. Layout of the new cycle.

machine, then a new motion cycle can be defined with a sequence of moves as the following:

At the beginning of the cycle, a part is being processed by the second machine and the robot is located opposite the input area. According to definition 1, the sequence of the robot movements is $A_{01}A_{23}A_{02}A_{13}$, that is, the robot picks a part on the input buffer (the needed time is ε) and takes the part to the first machine (δ). Then it loads the part on the first machine (ε) and moves towards the second machine (δ) and waits opposite this machine until it completes processing the part (w_2), after the second machine has completed the process, the robot unloads the part from the second machine (ε) and takes it to the output buffer (δ), then it loads it on the output buffer (ε). Then the robot comes back to the input buffer (3ε), picks a part from the input buffer (ε), moves it to the second machine (2δ), and loads it on it (ε), then it turns back to the first machine (δ) and waits opposite the machine until it completes processing the part (w_1). When processing is finished, the robot unloads the part from the first machine (ε), takes it to the output (2δ) and then loads it on output area (ε). Then it comes back to the input buffer (3δ). Thereby, the cycle is finished and the robot returns to the initial position. During this cycle, two parts are produced, so the cycle is called 2-unit cycle. In this cycle, it is assumed that the robot has no waiting time in the input and the output buffer. It should be noted that since each machine has the ability to perform all operations, in the process of producing n parts and in each sequence of cyclic moves, after one stage we have a repetitive sequences of robot movements which continues until n parts are produced. In this proposed cycle, the starting point of the mentioned repetitive process is the beginning of the cycle.

In this case, the cycle time for producing two parts is calculated as follows, and its parameters are:

ε : Loading or unloading time

δ: The time in which the robot moves between two successive locations

P: Processing time of each part on the machines

w_i : Waiting time in front of machine i

C_t: Cycle time:

$$2-unit\ C_t = \varepsilon+\delta+\varepsilon+\delta+w_2+\varepsilon+\delta+\varepsilon+3\delta+\varepsilon+2\delta \\ +\varepsilon+\delta+w_1+\varepsilon+2\delta+\varepsilon+3\delta = 8\varepsilon+14\delta+w_1+w_2 \quad (1)$$

$$w_1 = Max\{0, P-(8\varepsilon+4\delta+w_2)\} \quad (2)$$

$$w_2 = Max\ \{0, P-(2\varepsilon+4\delta)\} \quad (3)$$

Therefore, the time needed to produce a part in this cycle is:

$$1-unit\ \ C_t = \frac{8\varepsilon+14\delta+w_1+w_2}{2} \quad (4)$$

$$1-unit\ C_t = 4\delta+7\delta+\frac{1}{2}Max\{0,P-(2\varepsilon+4\delta)\}+\frac{1}{2}Max\{0,P-(4\varepsilon+8\delta+w_2)\} \quad (5)$$

The layout of the proposed cycle and the initial position are shown in Figure 1.

PETRI NETWORK MODEL

Many systems can be modeled on Petri nets and it is possible to show their features using these networks (Maggot, 1984). Petri network is a suitable device for mathematical and graphical modeling. The graphical behavior of variables and the ability to convert them into flowchart and diagram is one of the important features of Petri nets (Kamalabadi, 1996). Petri networks were introduced for the first time by Adam Petri in 1962. Lee and Yung (1955) presented a new method for planning flexible process sequences using Petri networks (Maggot, 1984). Petri nets are directed split graphs which are divided into two groups of place and transition. The directed arcs link some places to some transitions or link some transitions to some places. Notation in a Petri network is a vector whose elements show the number of arcs. Each Petri network is shown as $PN = \{P, T, A, W, M_0\}$ (Maggot, 1984) in which:

P: a finite set of places $P = \{P_1, P_2 ...., P_n\}$

T: a finite set of transitions $t = \{t_1, t_2 ...., t_m\}$

A: a finite set of arcs $A \subset \{P \times T\} \cup \{T \times P\}$

W: weight function associated with each location $P = \{1,2,.....\}$ $W : A$

M0: Initial network markup $P = \{0,1,2,.....\}$ $M_0 : P$

Mark changes in a Petri network involving firing (transposition) is as the following:

1. The firing is performed when the location P_i has at least $W(P_i, t)$ token, whereby $W(P_i, t)$ is the arc weight of Pi to t.
2. An active transposition may lead to firing depending on whether its movement is performed or not.
3. If the transition t is fired, $W(P_i, t)$ of tokens is reduced from each P_i to t input place and $W(P_0, t)$ number of tokens is added to each

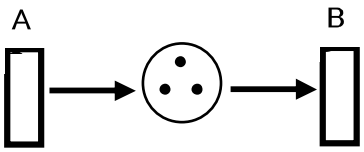

Figure 2. Marked graph related to theorem 1.

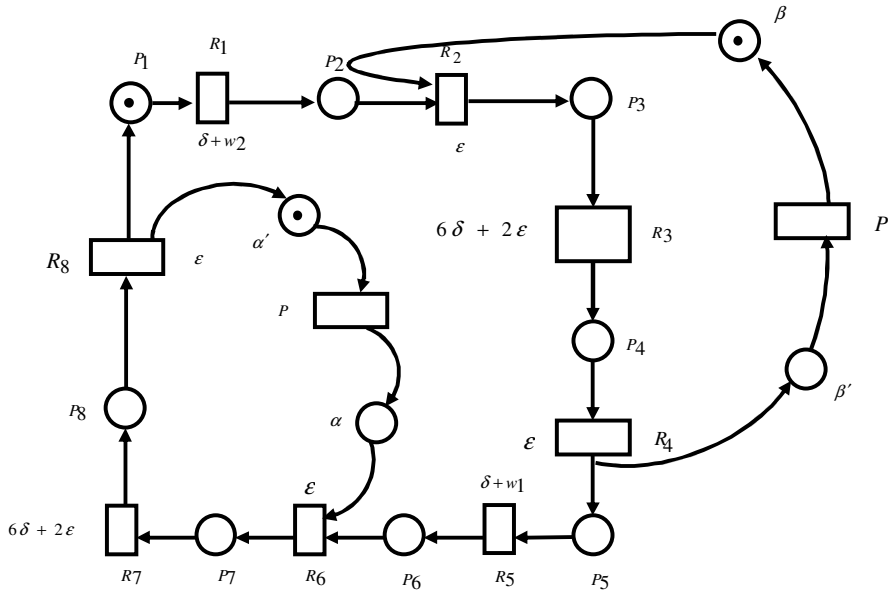

Figure 3. Petri network graph for same parts production cycle.

P0 output location of transition t are added. W (t, P_0) is the arc weight of t to P_0 (Maggot, 1984).

Theorem 1: In a marked graph, for each location that has M_i tokens the following relationship $S_B \geq S_A + M_i C_t$, whereby S_B, S_A are the starting times of transitions of B and A, and C_t is the cycle time in timed Petri network. Figure 2 shows this marked graph.

Timed Petri network model for producing identical parts in the new cycle

At the beginning of the proposed cycle, it is assumed that a part on the second machine is being processed and the robot is located opposite the input buffer. For simplicity of calculation and drawing Petri networks, in modeling the problem and drawing its graph we assume that the starting point of the cycle moves to the moment when the robot loads the first machine and leaves for the second machine; at this moment, a part is being processed by both machines. At the end of the cycle the robot returns to this situation. Accordingly, the graph for the same parts production cycle is shown in Figure 3.

The parameters needed for modeling this problem by Petri network are the following:

R_1: moving toward the second machine and waiting opposite the second machine ($\delta + w_2$)

R_2: unloading the part from the second machine (ε)

R_3: moving towards the output buffer, loading the part on the output buffer, moving towards the input buffer, unloading the part from the input buffer and moving towards second machine ($6\delta + 2\varepsilon$)

R_4: loading the part on the second machine (ε)

R_5: moving toward the first machine and waiting opposite the first machine ($\delta + W_1$)

R_6: the robot's unloading the part on the first machine (ε)

R_7: moving towards the output buffer and loading the part on the output buffer, moving towards the input buffer, unload the part from the input buffer, and moving toward the first machine($6\delta + 2\varepsilon$)

R_8: loading the part on the first machine (ε)

α ': starting moment of the first machine's processing M_1

α : The moment that the first machine has finished its task and is waiting for the robot to unload the part from it

β' : starting moment of the second machine's processing (M_2)

β : The moment that the second machine has finished its task and is waiting for the robot to unload the part from it

P: operating time of the first or the second machine

Mathematical model for production planning problem of producing same parts in the proposed cycle are the following:

$$\text{Min } C_t \qquad (6)$$

Subject to:

$$P_1 : S_1 - S_8 + C_t = \varepsilon \quad (7)$$

$$P_2 : S_2 - S_1 = \delta + w_2 \quad (8)$$

$$P_3 : S_3 - S_2 = \varepsilon \quad (9)$$

$$P_4 : S_4 - S_3 = 6\delta + 2\varepsilon \quad (10)$$

$$P_5 : S_5 - S_4 = \varepsilon \quad (11)$$

$$P_6 : S_6 - S_5 = \delta + w_1 \quad (12)$$

$$P_7 : S_7 - S_6 = \varepsilon \quad (13)$$

$$P_8 : S_8 - S_7 = 6\delta + 2\varepsilon \quad (14)$$

$$\alpha : S_6 - S_8 + C_t \geq \varepsilon + P \quad (15)$$

$$\beta : S_2 - S_4 + C_t \geq P + \varepsilon \quad (16)$$

$$S_i, w_1, w_2 \geq 0 \quad (17)$$

In this model, the objective function is to minimize the cycle time of producing same parts. The constraints are written according to the properties of timed Petri network and what were written in "Theorem 1".

Mathematical model for production planning problem of producing different parts in the proposed cycle

In the proposed new cycle, we assume that n different parts must be produced, and that the processing time for all parts is specified. Therefore, we can model the Petri network in a way that the optimal sequence of parts' entering the cell is determined. According to this model, n parts are produced by this production cycle in the minimum possible time.

Petri network graph related to the production of these n parts is obtained by repeating the same parts production cycle $\frac{n}{2}$ times as described earlier. Due to the fact that in the proposed same parts production cycle 2 parts are produced, for producing n parts in the cycle, the mentioned sequence must be repeated $\frac{n}{2}$ times. For modeling the new proposed different parts production cycle, in addition to the parameters used for producing identical parts, the following parameters are needed:

X_{1ij}: if the part i, is the j[th] part given to the first machine
X_{2ij}: if the part i, is the j[th] part given to the second machine
t: part counter $t \neq i$

Mathematical model for production planning problem with different parts are the following:

Min C_t (18)

Subject to:

$$P_{1,1} : S_{1,1} - S_{8,n} + C_t = \varepsilon \quad (19)$$

$$P_{1,j} : S_{1,j} - S_{8,j} = \varepsilon \qquad j = 2,....,n \quad (20)$$

$$P_{2,j} : S_{2,j} - S_{1,j} = \delta + w_2 \qquad j = 1,....,n \quad (21)$$

$$P_{3,j} : S_{3,j} - S_{2,j} = \varepsilon \qquad j = 1,....,n \quad (22)$$

$$P_{4,j} : S_{4,j} - S_{3,j} = 6\delta + 2\varepsilon \qquad j = 1,....,n \quad (23)$$

$$P_{5,j} : S_{5,j} - S_{4,j} = \varepsilon \qquad j = 1,....,n \quad (24)$$

$$P_{6,j} : S_{6,j} - S_{5,j} = \delta + w_1 \qquad j = 1,....,n \quad (25)$$

$$P_{7,j} : S_{7,j} - S_{6,j} = \varepsilon \qquad j = 1,....,n \quad (26)$$

$$P_{8,j} : S_{8,j} - S_{7,j} = 6\delta + 2\varepsilon \qquad j = 1,....,n \quad (27)$$

$$\alpha_1 : S_{6,1} - S_{8,1} + C_t \geq \sum_{i=1}^{n} x_{1in} . a_i + \varepsilon \quad (28)$$

$$\alpha_j : S_{6,j} - S_{8,j} \geq \sum_{i=1}^{n} x_{1ij} . a_i + \varepsilon \qquad j = 2,....,n \quad (29)$$

$$\beta_1 : S_{2,n} - S_{4,n} + C_t \geq \sum_{i=1}^{n} X_{2in} . bi + \varepsilon \quad (30)$$

$$\beta_j : S_{2,j} - S_{4,j} \geq \sum_{i=1}^{n} X_{2ij} . bi + \varepsilon \qquad j = 1,....,n-1 \quad (31)$$

$$\sum_{i=1}^{n} X_{1ij} + \sum_{i=1}^{n} X_{2ij} = 1 \qquad \forall j = 1,....., n \quad (32)$$

$$\sum_{j=1}^{n} X_{1ij} + \sum_{j=1}^{n} X_{2ij} = 1 \qquad i = 1,.....,n \quad (33)$$

$$X_{2in} = X_{2i0} = 1 \qquad i = 1,.....,n \quad (34)$$

$$X_{1in+1} = X_{1i1} = 1 \qquad i = 1,.....,n \quad (35)$$

$$X1_{i.j-1}(\sum_{t \neq i} X2_{t,j}) + X2_{i,j-1}(\sum_{t \neq i} X1_{t,j}) = 1 \quad \forall i, t, j \quad (36)$$

$$S_{ij}, w_1, w_2 \geq 0 \quad (37)$$

$$X_{1ij}, X_{2jj} = \{0,1\} \quad (38)$$

Constraints added to this model are the following:

Constraint (32) states that at any stage only one part must enter and it must be assigned only to one machine.

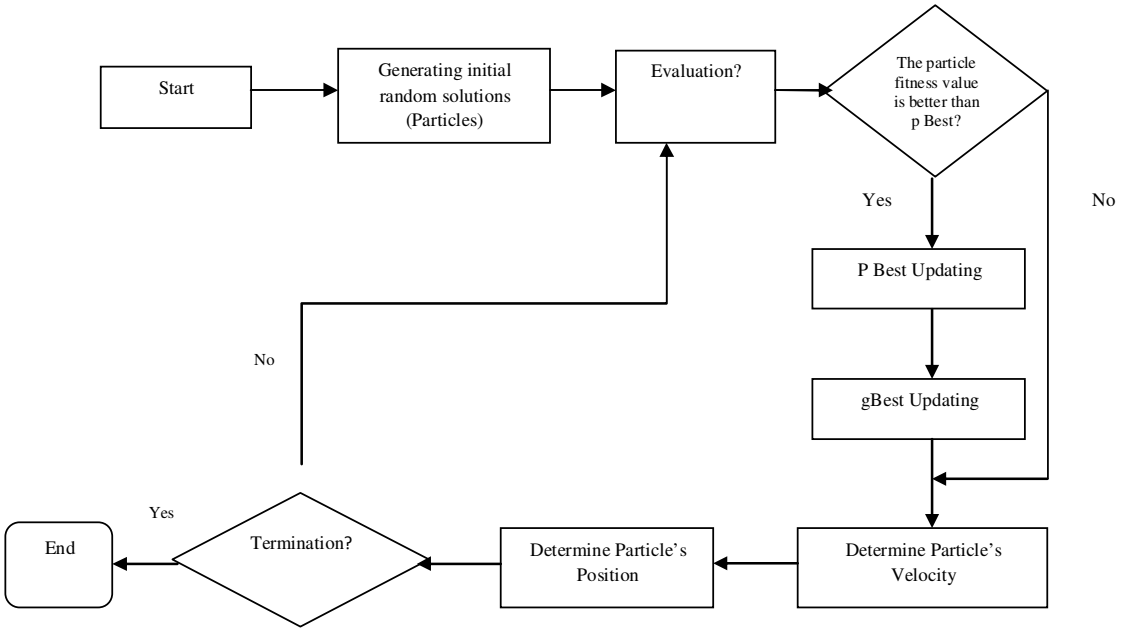

Figure 4. The general scheme of Particle Swarm Optimization algorithm.

Constraint (33) states that part i must enter only in one stage of the cycle.

Constraint (34) states that the part being processed on the second machine at the beginning of cycle is the n^{th} part in the previous cycle and represents the sequence in new cycle.

Constraint (35) states that the part being processed by the first machine at the beginning of cycle is $n+1^{st}$ part in the previous production cycle.

Constraints (36) state that the sequence of parts entering the new cycle must be observed, i.e. if in a stage one part is given to one of the machines, in next stage the next part is given to the other machine.

Constraints $P_{i,j}$ are written according to "Theorem 1". Also, the objective function of the problem is to minimize the cycle time to produce n various parts in the new proposed cycle.

Analysis of the computational complexity of the problem

In the new proposed cycle, the cycle time for producing n different parts depends on the sequence of parts entry. In the Petri network model presented for the production of different parts, waiting times on the first and second machines are as follows:

$$W_1 = Max\left\{0, (X_{1_{i\ j-1}}.P_i) - (4\varepsilon + 8\delta + w_2)\right\} \quad (39)$$

$$W_2 = Max\left\{0, (X_{2_{i\ j-1}}.P_i) - (2\varepsilon + 4\delta)\right\} \quad (40)$$

Therefore, it is clear that the waiting times are not independent of the sequence of parts entry. In a specific sequence, cycle time can be calculated in a polynomial time. Accordingly, this problem is placed in NP problems class (Hall et al., 1998). The results show that the problem $FRC3\left|K \geq 2, S_2\right|C_t$ is an NP-Complete problem which has been studied in 3-machine robotic cells producing different parts (Hall et al., 1998). Thus, the problem and the mathematical model presented in the new proposed cycle resemble this problem and it is similarly an NP-complete problem.

PARTICLE SWARM OPTIMIZATION (PSO) META-HEURISTIC

Particle swarm optimization (PSO) is a population based stochastic optimization technique that was developed by Kennedy and Eberhart (1995). In PSO, each solution is a bird in the flock and is referred to as a particle (Shi et al., 1998). The PSO has been applied successfully to a wide variety of optimization problems to find optimal or near-optimal solutions. Due to the complexity of the proposed model, it is very difficult to obtain optimum solution for this kind of problems by means of traditional approaches. Therefore, in this paper, we apply the PSO for large size problems. The general scheme of the applied PSO is provided in Figure 4. In the PSO, the position, x_i, of the *i*th particle is adjusted by a stochastic velocity, V_i. At each iteration of the algorithm, x_i and V_i are calculated according following equations (Shi et al., 1998).

$$\boldsymbol{V_{id} = W.V_{id} + c_1.rand(a).(P_{id} - x_{id}) + c_2.Rand(b).(P_{gd} - x_{id})} \quad (41)$$

$$X_{id} = X_{id} + V_{id} \quad (42)$$

$$-V_{\max} \leq V_{id} \leq V_{\max} \quad (43)$$

Equation 41 calculates a new velocity for each particle based on its previous velocity, the particle's position at which the best fitness so far has been achieved (P_{id} , or pBest), and position of the best particle of population (P_{gd} , or gBest). In this equation, Rand (a) and rand (b) are two random numbers that uniformly distribute in range

Table 1. Continuous representation.

Generated random numbers	0.36	1.21	2.45	0.59	1.75
Sequence	5	3	1	4	2
Continuous representation	2.45	1.21	0.36	1.75	0.59

[0, 1]. $c1$ and $c2$ are two learning factors, which control the influence of pBest and gBest on the search process and w is an inertia weight that balances global exploration and local exploitation and was proposed to decrease linearly with time from a value of 1.4 to 0.5 (Kamalabadi et al., 1996). $W_{\max}$ is an upper limit on the maximum change of particle velocity.

A new solution for part sequencing and robot move sequences in a 2-machine robotic cell based on the particle swarm meta-heuristic.

Solution representation

The solution representation should be so that we are able to decode it easily to reduce the cost of the algorithm. In this paper, a continuous representation is used (Kamalabadi et al., 2008).

To construct the continuous representation, first we need to generate as many as the number of the jobs to be produced random numbers between [0, $X_{\max}$], then the first smallest of them will be assigned to the position that contain the first job, the next smallest will be assigned to position that contain the second job, and so on as shown in Table 1.

SIMULATED ANNEALING META-HEURISTIC ALGORITHM

Simulated annealing algorithm (SA) is inspired by the process of metals annealing. In a real annealing process, metal's temperature is increased to a point that all the molecules are scattered in a molten form, then temperature will slowly decrease. Temperature decrease in real annealing is like decreasing the value of objective function for minimization problems. As temperature decrease rate in real annealing is effective on the quality of the final material, temperature decrease rate in SA algorithm is also effective on the quality of the final solution. One characteristic of SA algorithm is that it accepts non-improving solutions; this causes the algorithm not to get captured at a local optimum. This algorithm begins with a primary random solution; then in each temperature some of primary solution neighborhoods are studied. If objective function's value of the neighbor's solution is better than objective function's value of the primary solution, the neighbor's solution will be accepted, otherwise, to escape the local optimum the neighbor's solution with probability of $EXP(-\frac{\Delta}{T})$ will be accepted. This process will be repeated until a predefined number of neighborhoods are studied in each temperature. Afterwards, the temperature will decrease; the predefined number of neighborhood is also studied in this new temperature. The algorithm will stop when it reaches the stopping criterion. The pseudo-code of simulated annealing algorithm as used by (Xambre et al., 2003) is illustrated in Figure 5. The basic parameters of SA algorithm are as follows:

Solution representation

In this paper, the following chromosome structure has been used.

In this structure, n is the number of jobs; n_1 to n_n denote the number of a machine to which the related part is designated. Each one of these numbers is selected from one to two (the number of machines is two). After the structure of the chromosome was determined, the primary solution is randomly generated. To generate a chromosome for the problem the presented structure as shown in Figure 6 has been used.

Initial temperature

Initial temperature is one of the basic parameters of SA algorithm. The initial temperature should be selected in a way that most of non-improving solutions are accepted in the first repetition. In this article, the heuristic method based on (Safaei et al., 2008) as shown in Figure 7 has been used to determine the initial temperature.

Temperature decrement rule

SA algorithm begins with a relatively high temperature which decreases slowly in each repetition. There are various methods to reduce temperature in each repetition; in this paper, geometric scheduling criterion has been used:

$$T_k = \alpha T_{k-1} \qquad 0<\alpha<1 \qquad (44)$$

In the afore-stated relation, T_k is system's temperature in the k^{th} repetition and α is temperature reduction rate. Selecting a large value for α results in slow temperature decrement and better solution space searching, on the other hand, it increases algorithm's run time. Selecting a small value for α , results in fast temperature decrement and fast solution space searching. Therefore, α value should be selected in a way that there will be a balance between algorithm's run time and the quality of solutions. In this paper, α value has been considered to be 0.9.

Neighborhood structure

Neighbor solutions are a set of feasible solutions which are obtained from the primary solution. Each neighbor solution can be obtained through one movement (a change in the present solution). In this paper, the following neighborhood structure has been used; in this structure one of the genes is randomly selected and then its value is changed in a way that the resulted solution will be feasible The neighborhood structure is shown in Figure 8.

Number of repetitions in each temperature (*L*)

This parameter controls the number of investigated neighborhoods in each temperature. *L* value should be selected large to the extent

Select an initial temperature T_0

select an initial solution, s_0, *and make it the current solution , S, and the current best solution* s^*;

repeat

set repetition counter n=1

repeat

generates solution s_n *in the neighborhood of S*

calculates $\Delta = f(s_n) - f(s)$

if ($\Delta \le 0$) *then* $s = s_n$

else $s = s_n$ *with probability of p=* $EXP(-\frac{\Delta}{T})$

if $(f(s_n) < f(s^*))$ *then* $s^* = s_n$

n=n+1

until n> number of repetition allowed at each temperature level (L)

reduce the temperature T

Until stop criterion is true.

Figure 5. Pseudo-code of SA algorithm for minimization problems (Xambre et al., 2003).

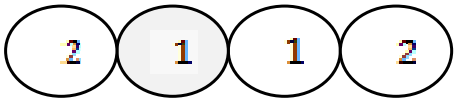

Figure 6. Solution structure of SA algorithm.

Sub Init Temp

Do

Generate two solution x_1, x_2 *at random*

Loop until $f(X_1) \neq f(X_2)$

Set $T_0 = \frac{-|f(X_1) - f(X_2)|}{\ln(0.9)}$

End sub

Figure 7. Pseudo-code of determining the initial temperature of SA algorithm (Safaei et al., 2008).

Figure 8. Neighborhood structure.

that results in an effective neighborhood search. On the other hand, *L* value should not be so large that would result in ineffective search and increase run time. In this paper, number of repetitions in each temperature (*L*) has been considered to be 10.

Stopping criterion

In the presented algorithm, stopping criterion has been considered to be the final temperature. The final temperature should be selected in a way that the probability of accepting non-improving solutions in the final repetitions will be close to zero.

EXPERIMENTAL RESULTS

To validate the proposed model and the implemented algorithm, various test problems are examined. The algorithm is coded into the *MATLAB 7.1* and run on the PC, processor at 2.4 GHz and Windows 7 using 4.00 GB of RAM. The experiments are implemented in two folds: first, for small-sized problems, the other for large-sized ones. For both of these experiments, we consider the following assumptions: 1. the values of ε and δ are equal to 1; 2. each experiment is repeated 15 times; 3. the processing time for all parts on the all machine are uniformly generated in range [10, 100].

Small-sized problem (number of parts is smaller than 20 parts)

The problem instances are randomly generated. The number of particle, termination criterion of PSO, Learning factors ($c1$ and $c2$), and $V_{\max}$ are fixed to 50, 50, 2, 2, and 3, respectively. For each instance, the results obtained are compared with the Lingo 8.0. By comparing the computational results of this algorithm and the optimum solutions of these problems which are solved by lingo 8.0, it can be concluded that this algorithm is able to find optimum solutions for all of these problems. The results of solving the problems by *LINGO 8.0* are illustrated in Table 4.

Large-sized problem (number of parts is more than 20 parts)

The problem instances are randomly generated. The number of particle, termination criterion of PSO, learning

Table 2. Computational results of PSO.

Problem no.	Number of parts	Worst solution value	Average solution value	Best solution value	Average performance time (s)
1	5	116	115	113.5	0.39624
2	10	183	183	183	0.49608
3	15	358.5	355.8	352.5	0.48672
4	20	572.5	572.5	572.5	0.54288
5	25	549.5	543	535.5	0.62712
6	30	662.5	662.5	662.5	0.6708
7	35	681.5	680.25	679	0.7215
8	40	851.5	851.5	851.5	0.7761
9	45	1128.5	1128.8	1127	2.45232
10	50	989.5	989.5	989.5	2.60832
11	55	1213	1216.5	1230.5	2.8392
12	75	1550	1549.5	1548.5	3.8376
13	100	2516	2516	2516	5.11368
14	120	2777.50	2777.50	2777.50	6.38976

Table 3. Computational results of SA.

Problem no.	Number of parts	Worst solution value	Average solution value	Best solution value	Average performance time (s)
1	5	118.5	118.5	118.5	0.2496
2	10	206	206	206	0.4407
3	15	387.5	387.5	387.5	0.1716
4	20	580.5	580.5	580.5	0.1783
5	25	600.5	600.5	600.5	0.1840
6	30	690.5	690.5	690.5	0.1872
7	35	776.5	763.3	715.5	0.1872
8	40	964.5	964.5	964.5	0.2028
9	45	1253.5	1252.7	1252	0.2145
10	50	1132.5	1132.5	1132.5	0.1872
11	55	1372	1368.8	1368	0.1840
12	75	1775.5	1767.5	1763.5	0.1778
13	100	2809	2809	2809	0.1497
14	120	3130.5	3130.5	3130.5	0.156

Table 4. Computational results of Lingo 8.0.

Problem no.	Number of parts	Best OFVa	BOUND	Run time (second)
1	5	113.5	113.5	<1
2	10	183	183	<1
3	15	352.5	352.5	<1
4	20	572.5	572.5	1800
5	25	535.5	535.5	2700
6	30	662.5	662.5	7200
7	35	679	679	14400
8	40	1254.5	0	18000

Table 4. cont'd

9	45	1487	0	18000
10	50	1623.5	0	18000
11	55	1931	0	18000
12	75	2122	0	18000
13	100	3459.5	0	18000
14	120	4634.5	0	18000

a. Objective function value.

factors ($c1$ and $c2$), and $V_{\max}$ are fixed to 100, 100, 2, 2, and 3, respectively. Because of complexity of the problem, the Lingo software cannot produce any results for most of the large-sized problems. However according to the computational results shown in Tables 2 and 3, algorithm PSO and SA can achieve proper solutions in acceptable time.

As aforementioned, each one of these problems have been solved by LINGO 8.0 software at first. LINGO 8.0 is able to produce global optimal solutions for problems. But, when dimensions of the problem increase, this software cannot obtain optimal solutions in a reasonable time. In this article, the maximum running time of LINGO 8.0 has been considered to be five hours. That is, if LINGO 8.0 cannot obtain the global optimal solution in less than five hours, the solution algorithm will stop and the best solution obtained by the software will be considered as its output. Therefore, meta-heuristic algorithms should be used for solving high dimension problems.

Conclusion

In this paper, after a discussion of the possible existing robot moves cycles in a 2-machine robotic cells, a new cycle with the assumption that the machines are the same and flexible with the ability to perform all the necessary operations for producing the same and different parts was considered. The main problem in this research is to minimize the cycle time in producing same and different parts by optimizing part sequencing and robot moves sequence in the robotic cell. Accordingly, we provided a mathematical programming model based on timed Petri network. Then the computational complexity of the model was analyzed and it was shown to be an NP-Complete problem. Also to solve this model, we proposed two meta-heuristic algorithms named PSO and SA algorithms.

Then the obtained solutions by these algorithms were compared to exact solutions by LINGO software. The results indicated that PSO algorithm can obtain better solutions in acceptable running time compared to other algorithms. Among the issues that can be considered in future researches are: i) to test the performance of the existing robot move sequences in the form of new proposed cycle; ii) to extend these problems to 3-machine robotic cells.

REFERENCES

Agnetis A (2000). Scheduling no-wait robotic cells with two and three machines. Eur. J. Oper. Res., 123: 303-314.

Agnetis A, Pacciarelli D (2000). Part sequencing in three-machine no-wait robotic cells. Oper. Res. Lett., 27: 185-192.

Akturk MS, Gultekin H, Karasan OE (2006). Robotic cell scheduling with operational flexibility. Discrete Appl. Math., 145: 334–348.

Bagchi TP, Gupta JND, Sriskandarajah C (2006). A Review of TSP Based Approaches for Flow shop Scheduling. Eur. J. Oper. Res., 169: 816-854.

Brauner N, Finke G (1999). On a conjecture about robotic cells: New simplified proof for the three machine case. Inform., 37(1): 20-36.

Crama Y (1997). Combinatorial optimization models for production scheduling in automated manufacturing systems. Eur. J. Oper. Res., 99: 136–153.

Crama Y, Kats V, Klundert VD (2000). Cyclic scheduling in robotic flow shops. Ann. Oper. Res., 96: 97-124.

Crama Y, Klundert VD (1999). Cyclic scheduling in 3-machine robotic flow shops. J. Schedul., 2: 35-54.

Dawande M, Geismar HN, Sethi SP, Sriskandarajah C (2005). Sequencing and scheduling in robotic cells: Recent developments. J. Schedul., 8: 387-426.

Deineko VG, Steiner G (2005). Robotic-Cell Scheduling: Special Polynomially Solvable Cases of the Traveling Salesman Problem on Permuted Monge Matrices. J. Comb. Optim., 9(4): 381-399.

Drobouchevitch IG, Sethi SP, Sriskandarajah C (2006). Scheduling dual gripper robotic cell one unit cycles. Eur. J. Oper. Res., 171: 598-631.

Gultekin H, Akturk M S, Karasan OE (2006). Cyclic scheduling of a 2-machine robotic cell with tooling constraints. Eur. J. Oper. Res., 174: 777–796.

Gultekin H, Akturk MS, Karasan OE (2007). Scheduling in a three-machine robotic flexible manufacturing cell . Comput. Oper. Res., 34: 2463–2477.

Hall NG, Kamoun H, Sriskandarajah C (1997). Scheduling in robotic cells: Classification, two and three machine cells. Oper. Res., 45: 421-439.

Hall NG, Kamoun H, Sriskandarajah C (1998). Scheduling in robotic cells: Complexity and steady state analysis. Eur. J. Oper. Res., 109: 43-65.

Kamalabadi IN (1996). A New Formulation for Scheduling Problems Through Petri-nets. Iranian Mathematical Conference.

Kamalabadi IN, Gholami S, Mirzaei AH (2008) . A New Solution for the Cyclic Multiple-Part Type Three-Machine Robotic Cell Problem based on the Particle Swarm Meta-heuristic. J. Ind. Syst. Eng., 1(4): 304-317.

Kamalabadi IN, Hall NG, Sriskandarajah C (2000). Minimizing cycle time in a blocking flow shop. Oper. Res., 48: 177-180.

Kennedy JE (1995).Particle swarm optimization. Proc. IEEE Int. Conf. Neural Netw. Australia, 1942–1948.
Maggot J (1984). Performance Evaluation of Concurrent Systems Using Petri Nets. Inform. Process. Lett., 8(1): 7-13.
Safaei N, Saidi-Mehrabad M, Jabal-Ameli MS (2008). A hybrid simulated annealing for solving an extended model of dynamic cellular manufacturing system. Eur. J. Oper. Res., 185: 563-592.
Sethi SP,d. Groupe d'études et de recherche en analyse des(1989). Sequencing of robot moves and multiple parts in a robotic cell. Montréal, Groupe d'études et de recherche en analyse des decisions.
Sethi SP, Sriskandarajah C, Sorger G, Blazewicz J, Kubiak W (1992). Sequencing of parts and robot moves in a robotic cell. Int. J. Flex. Manuf. Syst., 4: 331-358.
Sriskandarajah C, Hall NG, Kamoun H, Wan H (1998). Scheduling large robotic cells without buffers. Ann. Oper. Res., 76: 287–321.
Shi Y, Eberhart R (1998). A modified particle swarm optimizer. Proc. IEEE Int. Conf. Evol. Comput., pp. 69–73 .
Xambre AR, Vilarinho PM (2003). A simulated annealing approach for manufacturing cell formation with multiple identical machines. Eur. J. Oper. Res., 151: 434-446.