Operation Sequencing using Modified Particle Swarm Optimization

ZALMIYAH ZAKARIA and SAFAAI DERIS
Faculty of Computer Science and Information System
Universiti Teknologi Malaysia
81310 Skudai, Johor,
MALAYSIA
zalmiyah@utm.my, safaai@utm.my

Abstract - *Planning and scheduling (PS) problems in advanced manufacturing systems, such as flexible manufacturing systems, are composed of a set of interrelated problems, such as operation sequencing, machine selection, routing, and online scheduling. Operation sequencing deals with the problem of determining in what order to perform a set of selected operations such that the resulting sequence satisfies the precedence constraints as well as alternative operation constraints established by both the parts and operations. In this paper, modified particle swarm optimization (MPSO) has been used to generate a feasible operation sequence for a real world manufacturing problem. In addition, the directed mutation is used to accelerate the individuals move toward the optimal solutions. The quality of the result and its numerical performance is discussed in comparison with a standard genetic algorithm (SGA). After 10 runs, the result from SGA show that the possibilities for the solution to fall in the near optimal solution is about 30% compared with the result from MPSO which always force the constraints to be fully satisfied.*

Keywords: Operation sequencing, particle swarm optimization, process planning and scheduling.

1 Introduction

Process planning is the activity of translating a set of design requirements and specifications into technologically feasible instructions describing how to manufacture a part [1]. Generally, a process plan contains processes, process parameters, machines, routes, set-ups and tools required for production of parts. Normally process planning involve several or all of the following activities: (1) selection of required operations; (2) sequencing of selected operations; (3) selection of required tools; (4) determining setup requirements; (5) determining of operation parameters. Of these activities, operation sequencing is the most complex due to the need to consider several types of constraints and the size of the resulting solution space.

The operation sequencing problem is the problem of simultaneous selecting and sequencing operations required to produce a part while satisfying the precedence relations among operations [2].

There are several approaches have been used to determine an optimal sequence include integer programming [3], branch and bound [4], simulated annealing [5], heuristic [6], ant colony optimization [7], [8] and evolutionary techniques [2],[9],[10] ,[11], [12].

2 Approaches and Methods

The overall goal of this research is the development of an integrated planning and scheduling framework for a real world manufacturing environment. Thus this research involves two main research problems namely, process planning and production scheduling. This paper is more focusing on the former problem.

In this research, process planning is performed in two stages: resource-independent planning and resource-dependent planning. The purpose of resource-independent stage is to provide a means for determining the best set of plans for a part independent of the status of the shop floor resources. Then later when production of that part is released to the shop floor, the resource-independent planning phase completes the planning tasks (machine selection, route, parameter determination, etc.) based on knowledge of what shop-floor resources are available. Therefore, this paper is concerned with defining a feasible operation sequences independent of the availability of resources.

2.1 Sequencing Constraints

The task of operation sequencing is complicated by the large number of interactions that exist between the various factors which affect decision-making. According to Usher and Bowden [2], the factors which are resource independent shown in Table 1. As revealed in Table 1, the constraints which affect sequencing can be divided into those which address either the feasibility or optimality of a sequence. This division permits the construction of a system which applies the feasibility constraints to the task of generating alternative sequences, and the optimality criteria to the task of judging the quality of the resulting alternatives. A feasible sequence is one which does not violate any of the feasibility constraints listed in Table 1.

Table 1. Sequencing constraints
[Adopted from Usher and Bowden, 1996]

Feasibility constraints	Location reference Accessibility Non-destruction Geometric tolerance Strict precedence
Optimality criteria	Number of setups Continuity of motion Loose precedence

In this research, we only consider feasibility constraints, because the optimality criteria will be considered in another stage. The feasibility constraints adopted here are shown in Table 2.

Table 2. New sequencing constraints

Feasibility constraints	Location reference Accessibility Non-destruction Strict precedence Alternative constraint

The location constraint is concerned with an examination of the defined part features to determine what reference face is used to locate each feature. This reference identifies the necessity that the locating surface be machined prior to the associated feature. In order to machine a feature it must be accessible. The accessibility constraint evaluates each feature's accessibility based on the feature type and its location relative to other features. Features are defined as either primary or secondary. The primary features define the basic shape of the part (diameters, tapers, etc) and secondary features provide the detailed shape aspects (grooves, bends, etc.). The fact that since a secondary feature is defined as residing on a primary feature, it makes sense not to machine the secondary feature until the primary feature has been formed. Therefore, before a secondary feature, such as a groove, is cut on the taper of the part, the taper (a primary feature) must be machined to specifications.

The non-destruction constraint is concerned with ensuring that a subsequent operation does not destroy the properties of features machined in prior operations. This type of problem is limited to the interactions that occur between the secondary features which reside on the same primary feature. One example would be the need to tapering the parts prior to punching the parts.

Another constraint considers strict precedence whereby order is determined based on feature type and properties. One example would be an eye forming whose properties require the use of a bushing operation. However, before bushing can be performed, there is a need to form the eye first, and possibly reams, the internal part. The need for these preparatory operations is actually determined during operation selection. Therefore, the results of this constraint will not actually influence the plan until the operations are considered when writing out the sequence. The last constraint pertains to the alternative operation defined for the part. There are several alternative operations performed on the parts. One example would be one part only needs one type of end cutting, it is either diamond cutting or width cutting; or it is either end trimming or end grooving.

These feasibility constraints give us the capability to define a set of precedence between the features of a part resulting in the construction of a precedence relationship matrix (PRM) to represent these precedence relationships.

2.2 Particle Swarm Optimization

Particle swarm optimization (PSO) is a new population-based search algorithm based on the simulation of the social behavior of the swarms in nature such as flocking birds, schooling fish, etc. It was introduced by Russell Eberhart and James Kennedy in 1995 [13]. It is easily implemented in most programming languages and has proven to be both very fast and effective when applied to a diverse set of optimization problems. PSO combines cognition model that values self experience and social model that values experience of neighbors.

PSO has been applied successfully to a wide variety of search and optimization problems like travel salesman problem [14],[15], flow/job shop scheduling problem [16],[17],[18],[19], university timetabling problem [20][21], machining parameter optimization [22] and generator maintenance scheduling [23].

A swarm consists of N particles flying around in a D-dimensional search space. Each particle holds a position (candidate solution to the problem) and a velocity (the flying direction and speed of the particle). Each particle successively adjust its position toward the global optimum according to two factors: the best position visited by itself (*pbest*) and the best position visited by the whole swarm (*gbest*). Each particle of PSO can be considered as a point in the solution space. If the number of particle is N, then the position of the i-th (i=1,2...N) particle is expressed as X_i. The best position passed by the particle is $pbest_i$. The velocity is expressed with V_i. The best position of the swarm is *gbest*. Therefore, particle i will update its own velocity and position according to equations:

$$V_i^{t+1} = w \times V_i^t + c_1 \times rand()_1 \times (pbest_i - X_i^t) + c_2 \times rand()_2 \times (gbest - X_i^t) \quad \ldots\ldots\ldots\ldots\ldots\ldots \quad (1)$$

$$X_i^{t+1} = X_i^t + V_i^{t+1} \quad (2)$$

where, V_i^{t+1} and V_i^t are velocities of particle i at time $t+1$ and t, respectively. X_i^{t+1} and X_i^t are positions of particle i at time $t+1$ and t, respectively. c_1 and c_2 are two constant weighting factor related to *pbest* and *gbest*, respectively. $rand()_1$ and $rand()_2$ are two random number between 0 and 1. $pbest_i$ is *pbest* position of particle i, *gbest* is *gbest* position of swarm and w is the inertia weight.

The basic PSO algorithms are as follow:

1. *Initialize the swarm from the solution space (position and velocity of each particle)*
2. *Evaluate fitness of each particle.*
3. *Modify gbest, pbest and velocity.*
4. *Move each particle to a new position.*
5. *Go to step 2, and repeat until convergence or a stopping condition is satisfied.*

2.3 Comparison to Genetic Algorithms

There are several similarities and dissimilarities between PSO and GA. They are as follow:

- Similarity
  - Both algorithms start with a group of a randomly generated population.
  - Both have fitness values to evaluate the population.
  - Both update the population and search for the optimum with random techniques.
  - Both do not guarantee success.
- Dissimilarity
  - Unlike GA, PSO has no evolution operators such as crossover and mutation.
  - In PSO, the potential solutions, called particles, fly through the problem space by following the current optimum particles.
  - Particles update themselves with the internal velocity.
  - They also have memory, which is important to the algorithm.
- Advantages
  - PSO is easy to implement and there are few parameters to adjust.
  - Compared with GA, all the particles tend to converge to the best solution quickly

2.4 Operation Sequence Coding

Application of an evolutionary search technique like genetic algorithms (GA) or particle swarm optimization (PSO) requires a method for representing a solution. Since operation sequencing problem is an order-based problem like travel salesman problem, we used path representation to represent the sequence. In this problem a sequence is represented as a list of n operations. If operation 'i' is the j-th element of the list, operation 'i' is the j-th operation to be performed. Hence, the sequence 3-2-5-6-1-4 is simple represented by 325614.

Then, we used this sequence as the position of a particle, which is represented by X_i. Thus, each position of a particle i, X_i is a sequence of operations to be performed, in order to produce a part, as follows:

1	3	9	A	B	C	D	E	F	G	H	J

Then velocity of each particle i, represented by V_i is a randomly generated mutation rate between 0 and V_{max} where $V_{max} = 0.5 * length(X_i)$ and $length(X_i)$ is the length of the position X_i.

The sequence of operations is bounded to the precedence constraints. Table 3 shows the example of precedence constraints for a number of processes. $O_{04},\ldots, O_{07}$ is a set of flexible-route operations which can be performed in any order.

Table 3. Precedence constraints

Id	Operation	Precedes
O_1	Shearing	O_2
O_2	Center Hole Punching	O_3, O_4, O_5, O_6, O_7
O_3	Berlin Eye Forming	*nil*
O_4	Short tapering	O_5, O_6, O_7
O_5	End punching	O_4, O_6, O_7
O_6	Bevel hole punch	O_4, O_5, O_7
O_7	Diamond cut	O_4, O_5, O_6

There are several approaches have been used to represent precedence relationships among features. They are feature precedence graph (FPG) [2], rules [5] and precedence relationship matrix (PRM) [8]. In this research, we used another kind of precedence-relation matrix as shown in Figure 1 to represents the constraints and relationships between the operations.

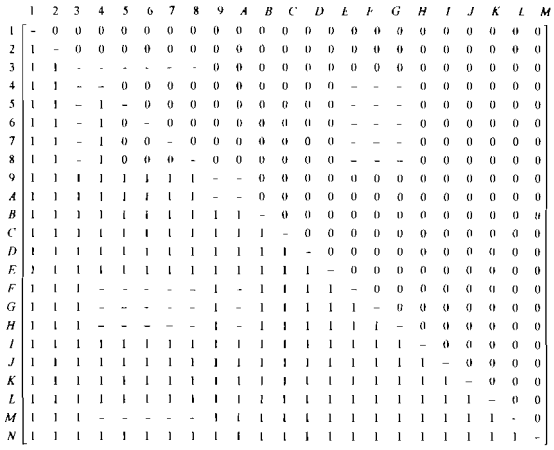

Figure 1. Precedence-relation matrix

The value of the matrix is either

$$\rho_{ij} = \begin{cases} 0 & \text{if } i \text{ can precede } j \\ 1 & \text{if } i \text{ can not precede } j \\ - & \text{if } i \text{ and } j \text{ are two alternative operations} \end{cases}$$

We use mutation operator to make changes to the sequence. Mutation is a unary operator that introduces random modifications of the sequence in order to add diversity to the solution. During the past decade, several mutation operators have been proposed for permutation representation, such as inversion, insertion, displacement, and reciprocal exchange mutation.

In order to preserve valid sequence, here we used reciprocal exchange method which swaps two values in the sequence. The algorithms will randomly choose two mutation points and swap the values in those particular points.

As shown in Figure 2, reciprocal exchange mutation selects two positions at random and swaps the values on these positions.

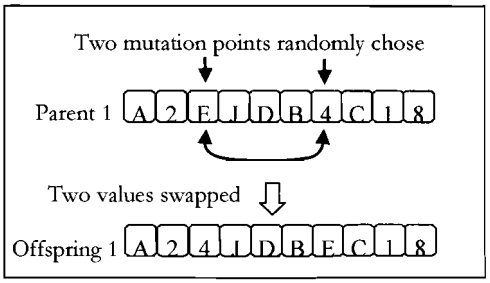

Figure 2. Mutation using reciprocal exchange

Then, the new positions or sequences of next generation are produced by following several steps:

- Movement of the particles is processed by the following procedure (Adopted from [20]):
  1. Each particle (X_t) must be randomly swap two operations for V_t times.
     $S_{t+1} = V_t * mutation\ (X_t)$
  2. Randomly copy a sequence of operations from the local best (P_t) to particle (S_{t+1}).
     $W_{t+1} = rand * copy\ (S_{t+1}, P_t)$
  3. Randomly copy a sequence of operations from the global best (G_t) to W_{t+1}.
     $X_{t+1} = rand * copy\ (W_{t+1}, G_t)$

2.5 Fitness Function

For our problem, the fitness of a sequence is obtained by computing the cost of penalty for the constraints violation according to the sequence. Thus our objective function is to

$$\min \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} p_{ij} \quad \forall\, i, j \text{ in the sequence}$$

subject to the precedence constraints represented by precedence-relation matrix (PRM) shown in Figure 1.

3 Results and Discussion

The goal of sequencing is to find an operation sequence which satisfies the constraints mentioned in the previous section. The constraints have been represented in the form of precedence-relation matrix (PRM).

In order to demonstrate the practicability and efficiency of the proposed algorithm, different numerical simulations are tested and evaluated. The algorithm is run on a personal computer with an Intel Pentium IV, 512MB RAM, on Microsoft Windows 2000 Professional. The codes are written in the LISP language.

Each trial run of our program started with a randomly created generation of individuals. The program was allowed to evolve this generation up to 50 times.

In order to show the effectiveness of the proposed algorithms, several runs have been done to be compared with the result from standard genetic algorithms (SGA).

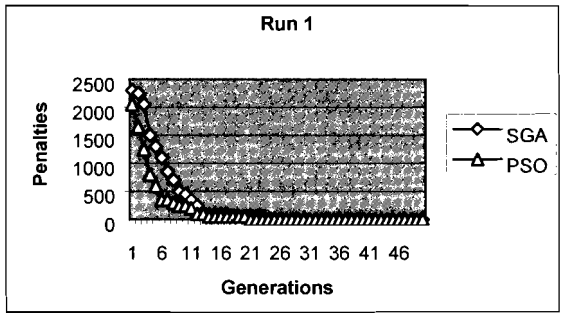

Figure 3. Modified PSO vs Standard GA (Run 1)

Figure 3-5 show the comparison results for Modified PSO vs Standard Genetic Algorithms (SGA).

The graphs show that in each trial modified PSO found the solution earlier than SGA.

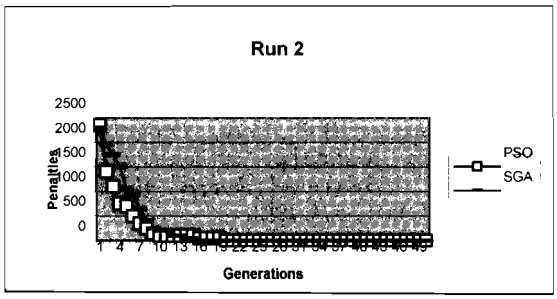

Figure 4. Modified PSO vs Standard GA (Run 2)

As stated earlier, PSO have a number of initial solutions which represented by a number of particles and every

particles strive to get their own optimal solution. The results shown in Figure 3-5 prove that the cooperation among the particles assist the algorithms to converge earlier, compared to SGA which only have one candidate solution to be manipulated in order to get the optimal solution.

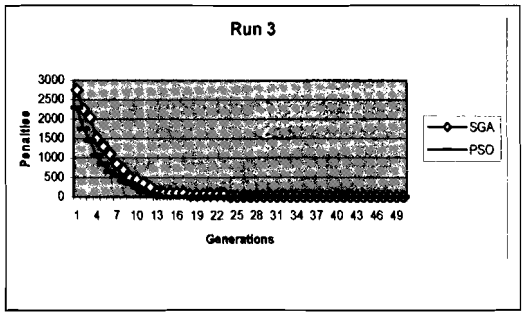

Figure 5. Modified PSO vs Standard GA (Run 3)

This is of the most significant advantage for PSO compared to GA. With a number of candidate solutions PSO can come out with a near optimal solution faster than GA. However, the author believes that GA also can perform this advantage through parallel structure. Hence, we conclude that the performance of PSO is comparable with parallel GA.

4 Conclusions

This paper discusses the implementation of modified PSO to solve operation sequencing problem. The results of this work also show that the modified PSO found the solution faster than SGA. It is believe that the cooperation among a number of particles help the algorithms to find the optimal solution faster than SGA.

References

[1] P. Scallan, *Process Planning: The design/ manufacture interface*, Butterworth-Heinemann. Burlington MA. 2003.

[2] J.M. Usher, and R.O. Bowden, "The application of genetic algorithms to operation sequencing for use in computer-aided process planning," *Computers & Industrial Engineering*, 30(4), 999-1013, 1999.

[3] C.-J. Lin, and H.-P. Wang, "Optimal operation planning and sequencing: minimization of tool changeovers," *International Journal of Production Research*, 31(2), 311-324, 1993.

[4] C. Koulamas, "Operation sequencing and machining economics," *International Journal of Production Research*, 31(4), 957-975, 1993.

[5] Ma, G.H., Zhang, Y.F., Nee, A.Y.C., "A simulated annealing based optimization algorithm for process planning", *International Journal of Production Research*, 38(12), 2371-2387, 2000.

[6] D.-H. Lee, D. Kiritsis and P. Xirouchakis, "Search heuristics for operation sequencing in process planning," *International Journal of Production Research*, 39(16), 3771-3788(18), 2001.

[7] P.R. McMullen, "An ant colony optimization approach to addressing a JIT sequencing problem with multiple objectives," *Artificial Intelligence in Engineering*, 15(3), 309-317, 2001.

[8] P.K. Jain and G. V. Kumar, "Operation Sequencing using Ant Colony Optimization Technique," *IEEE International Conference on Systems, Man and Cybernetics*, 2005.

[9] Z. Zakaria and S. Deris, "Operation Sequencing using Multi-population Directed Genetic Algorithms." Proc. Computer Science and Mathematics Symposium, KUSTEM, Kuala Terengganu, 9-10, December 2006.

[10] B. Awadh, N. Sepehri and O. Hawaleshka, "A computer-aided process planning model based on genetic algorithms," *Computers & Operations Research*, 22(8), 841-856, 1995.

[11] N.-C. Chiu, S.-C. Fang and Y.-S. Lee, "Sequencing parallel machining operations by genetic algorithms," *Computers & Industrial Engineering*, 36(2), 259-280, 1999.

[12] L. Li, J.Y.H Fuh, Y.F. Zhang and A. Y. C. Nee, "Application of genetic algorithm to computer-aided process planning in distributed manufacturing environments," *Robotics and Computer-Integrated Manufacturing*, 21(6) 568-578, 2005.

[13] J. Kennedy, and R. Eberhart, "Particle Swarm Optimization," Proc. of the 1995 IEEE International Conference on Neural Networks. 1942-1948, 1995.

[14] K.-P. Wang, L. Huang, C.-G. Zhou and W, Pang, "Particle swarm optimization for traveling salesman problem," *International Conference on Machine Learning and Cybernetics*. 3: 1583-1585. IEEE. 2003.

[15] P. Wei, W. Kang-Ping, C.-G., Zhou, L.-J., Dong, M., Liu, H.-Y., Zhang, J.-Y., Wang, "Modified particle swarm optimization based on space transformation for solving traveling salesman problem," *Proceedings of the Third International Conference on Machine Learning and Cybernetics*, Shanghai. 26-29 August 2004.

[16] S. Chandrasekaran, S. G. Ponnambalam, R. K. Suresh, N. Vijayakumar, "A Hybrid Discrete Particle Swarm Optimization Algorithm to Solve Flow Shop Scheduling Problems," *Conference on Cybernetics and Intelligent Systems.*: IEEE. 2006.

[17] Z. Lian, X. Gu and B. Jiao, "A similar particle swarm optimization algorithm for permutation flowshop scheduling to minimize makespan." *Applied Mathematics and Computation.* 175(1): 773-785. 2006.

[18] Z. Lian, B. Jiao and X. Gu, "A similar particle swarm optimization algorithm for job-shop scheduling to minimize makespan." *Applied Mathematics and Computation.*183(2): 1008-1017. 2006.

[19] Jerald, J., P. Asokan, G. Prabaharan and R. Saravanan, "Scheduling optimisation of flexible manufacturing systems using particle swarm optimisation algorithm," *The International Journal of Advanced Manufacturing Technology.* 25(9):964-971. 2005.

[20] S.-C. Chu, Y.-T. Chen and J.-H. Ho, "Timetable Scheduling Using Particle Swarm Optimization," *First International Conference on Innovative Computing, Information and Control.* ICICIC '06. 2006.

[21] D.R. Fealko, "*Evaluating Particle Swarm Intelligence Techniques for Solving University Examination Timetabling Problems,"* Nova Southeastern University: PhD Thesis. 2005.

[22] G. Liang, G. Haibing, and Z. Chi, "Particle swarm optimization based algorithm for machining parameter optimization," *Fifth World Congress on Intelligent Control and Automation* (WCICA 2004). 4: 2867-2871. 2004.

[23] K. Chin Aik, and D. Srinivasan, "Particle swarm optimization-based approach for generator maintenance scheduling," *Proc. of the Swarm Intelligence Symposium,* SIS '03: IEEE. 2003.