VOL. 9, NO. 4, APRIL 2014 ISSN 1819-6608
ARPN Journal of Engineering and Applied Sciences


www.arpnjournals.com

IMPLEMENTATION OF GENETIC ALGORITHM TO OPTIMIZE THE ASSEMBLY SEQUENCE PLAN BASED ON PENALTY FUNCTION

Arun Tom Mathew[1] and C. S. P. Rao[2]
[1]VIT University, Vellore, India
[2]National Institute of Technology Warangal, Warangal, India
E-Mail: aruntom123@gmail.com

ABSTRACT

Genetic Algorithms (GA) are, conceptually, suitable to optimize the Assembly Sequence Planning (ASP) problem. GA was implemented in this research to optimize the ASP problem because they can easily handle large search spaces, flexibility in defining the constraints and derive them in a fitness function. A penalty function approach has been used to compute the fitness value for assembly sequences. The penalty function approach was chosen as the penalties are easy to define, realistically capture the difficulties associated with the assembly process and the number of penalties to consider is relatively reduced. The evaluation of the penalty function is simple and straightforward, a most desirable feature for a population-based search.

Keywords: genetic algorithm, assembly sequence planning, subassemblies, penalty function.

INTRODUCTION

In general, optimization problems to be addressed have several objectives to be optimized. As the number of objectives of the problem increases, the complexity of the problem becomes high because the objectives considered are often contradictory to one another. The researcher who tackles an optimization problem with multiple objectives needs a tool for optimizing their problem. Because objectives to be optimized in the problem are often contradictory to one another, the optimal solution of the problem is not obtained as a single solution. That is, a set of solutions is to be obtained for the problem. Since multiple solutions are to be obtained as solutions of the problem, GA's as a kind of multi-point search potentially have an advantage for optimization problems with multiple objectives. However, GA's have been mainly applied to optimization problems with a single objective.

Optimizing the Assembly Sequence Planning problem involves selecting an optimum or near optimum feasible assembly sequence according to an objective function based on optimization criteria. Genetic algorithm is chosen for the optimization of ASPP as it has the ability to handle large scale problems and flexibility in defining an objective function.

The input for the guided search operator is the model of the product. The solutions of the ASPP are generated through guided search, when all precedence relations are considered. Crossover is the main genetic operator. The condition imposed in this case is the feasibility of the children chromosomes. A penalty function approach has been used to compute the fitness value for assembly sequences. The penalty function approach was chosen because the penalties are easy to define, realistically capture the difficulties associated with the assembly process and the number of penalties to consider is relatively reduced. Also, the evaluation is simple and straightforward, a most desirable feature for a population-based search.

LITERATURE SURVEY

More recent research has begun to apply artificial intelligence based algorithms to solve the assembly sequence planning problem. In particular, motivated by the success of genetic algorithms (GA's) in solving difficult and complex combinatorial problems (Gen and Cheng, 1997), GA's have been applied to assembly sequence planning to find optimal or near-optimal assembly plans for a structure. GA's do not generate, and then evaluate, all possible candidate solutions when searching for an optimized assembly plan instead, GA's search for an optimized assembly plan using a directed stochastic search of the product's solution space of possible assembly plans. Thus, GA's attempt to find optimized assembly plans, while analyzing only a small number of possible solutions. The need for GA's in assembly sequence planning is to dramatically reduce time required to find an acceptably optimized assembly plan, for any given product structure, an assembly plan, encoded as a chromosome, representing the assembly order for components in the product structure. GA's usually evaluates an individual's fitness, with respect to the entire population, by applying a function that incorporates, for example, measures of assembly reorientations required, assembly tool changes required, and/or assembly fixtures required. Next, GA's select individuals or pairs of individuals, based on the fitness, to produce next generation of individual assembly plans. For producing any individual offspring assembly sequence, the GA chooses one genetic operator crossover, mutation, based upon pre-defined probability settings. The GA then applies the chosen operator to chromosome string(s) selected from the population, based on relative fitness, to produce the next-generation population. A crossover operator exchanges component assembly order information between two chromosomes. A mutation operator exchanges the assembly order of two components within a single selected chromosome.

Most GA's for Assembly Sequence Planning use fixed population sizes from generation to generation, to

limit algorithm run time. Thus, when the GA generates enough new valid offspring to fill the next generation, the old population is replaced by the new offspring population and a fitness value for each new member is then calculated. The GA process of generating new higher-fitness offspring from the previous generation repeats until pre-defined termination criteria are satisfied. Common termination criteria used include a run-time limit, the number of generations, or a predefined desired assembly sequence fitness level. Bonneville *et al*. [1] first introduced, and demonstrated the feasibility of using, a genetic algorithm for Assembly Sequence Planning. Bonneville *et al*.'s early GA-based assembly planner used two basic genetic operators, *crossover* and *mutation*. However, Goldberg [2] showed that genetic algorithms that use only cross-over and mutation operators are limited in performance. Sebaaly and Fujimoto [3] also proposed a genetic algorithm for solving assembly sequence planning problems. Chen [4] proposed a simplified genetic algorithm for automatically generating assembly sequences. Chen used five genetic operators (crossover, mutation, cut-and-paste, break-and-joint, and reproduction) to promote a more thorough search of the product's assembly plan solution space.

FITNESS FUNCTION

The objective function taken for present study is to minimize the cost of assembling the product. The fitness function for evaluating chromosome or assembly sequence is defined as to find a sequence with maximum penalty value.

A single optimization criterion, the fitness function for each sequence is defined as:

$$f(x).FF = \frac{PFF_i(1) + PFF_i(2) + \ldots\ldots\ldots + PFF_i(n)}{n}$$

Where PFFi (i) is the componential fitness function associated with the assembly of a product. The fitness function is defined as a sum of componential fitness functions, PFF (i), each corresponding to an assembly task. The value of the fitness function depends on the definition of the optimization criterion and the position of the gene in the assembly sequence.

Objective Function; $F = \frac{1}{f(x)}$

THE PENALTY MATRIX

It is assumed that components are assembled vertically, downwards, on an assembly press. The penalty represents the cost incurred by assigning an unsuitable or unfeasible assembly sequence. The penalty value of an assembly operation is calculated using the penalty method proposed by Chen *et al*. [4]. Table-1 represents an illustrative example of the penalty weightings of some crucial factors. For instance, the penalty index is set 0 if component 'i' is absolutely located above component 'j'; the penalty index is 5 for a close above and little difficult to assemble; the penalty index is set 9 if the assembly relation is a "loose above" relation; the penalty index is set 999, if the assembly relation is prohibited. By employing the idea of penalty, other crucial factors concerning Assembly Sequence Planning, such as the frequency of changing tools, the similarity of assembly operations, the quality of assembly, the complexity of assembly, etc., can be considered simultaneously. Importantly, for avoiding the inaccurate estimation of the penalty of respective factors, it needs experienced engineers to perform the evaluation. The penalty values are as follows:

Table-1. Description of penalty index.

Penalty rank	penalty index	Description
1	0	Simple work, forced precedent sequence, direct or absolute above relation
2	1-5	A little difficult, need careful operation, tools changing infrequently, closed above relation
3	6-9	Very difficult, easily damage the component, tools changing frequently, loose above relation
4	999	Prohibit no relation.

The value of those penalties, from experience, is consistent with the difficulty to perform the related operations at the shop-floor level.

In order to minimize the time taken for searching for the solution, the initial population is taken from the feasible solution generated using the algorithm described earlier. After evaluation of each string based on its fitness value, these strings are further optimized by applying three major GA operators namely reproduction, cross over and mutation. The selection of better strings is based on the actual count arrived in initial sequence. The generic operator is used to generate new population that has better strings than old population is called parent-1 and is used for the next generic operation i.e. crossover. The strings obtained from reproduction are then mated at a given probability called as the crossover rate. The resultant string should be a feasible sequence and satisfies all the precedence constraints. The mutation operator makes random changes in one or more elements of the string. Mutation is done with a small probability called the mutation probability. This is done to protect loss of some potentially useful strings. The strings obtained after mutation should be a feasible sequence and satisfy all the precedence constraints.

The settings of GA parameters were as follows.

- Population size = 100
- Crossover probability $P_c = 0.7$
- Mutation probability $P_m = 0.3$
- Stopping criterion: 5000 generation was selected.

IMPLEMENTATION OF GENETIC ALGORITHM TO OPTIMIZE THE ASSEMBLY SEQUENCE PLAN

Genetic algorithm is applied to various products to generate the optimal assembly plan. The feasible assembly sequences generated becomes the initial set of chromosomes after crossover and mutation; the algorithm should make sure that the child chromosomes also represent a feasible assembly sequence satisfying all the precedence constraints. These feasible sequences generated are based on the algorithms developed by Arun Mathew and C.S.P Rao [5, 6]. Based on the penalty matrix in Table-2 for the product shown in Figure-1, the optimum sequence is generated.

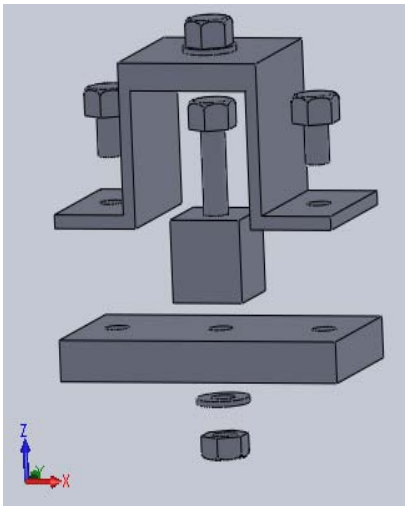

Figure-1. Sample assembly.

Table-2. Penalty matrix for the example.

	A	b	c	d	E	f	G	h	I	j
A	999	9	5	999	999	9	0	9	999	5
B	9	999	8	999	999	999	999	999	999	999
C	5	0	999	0	6	999	999	0	999	999
D	999	999	9	999	0	999	999	999	999	999
E	999	999	8	7	999	999	999	999	999	999
F	9	999	999	999	999	999	3	999	9	9
G	4	999	999	999	999	0	999	999	999	999
H	9	999	5	999	999	999	999	999	999	999
I	999	999	999	999	999	0	999	999	999	0
J	0	999	999	999	999	6	999	999	3	999

Based on the fitness function, the sequence having the maximum fitness value is the optimal sequence and is shown in the Figure-2.

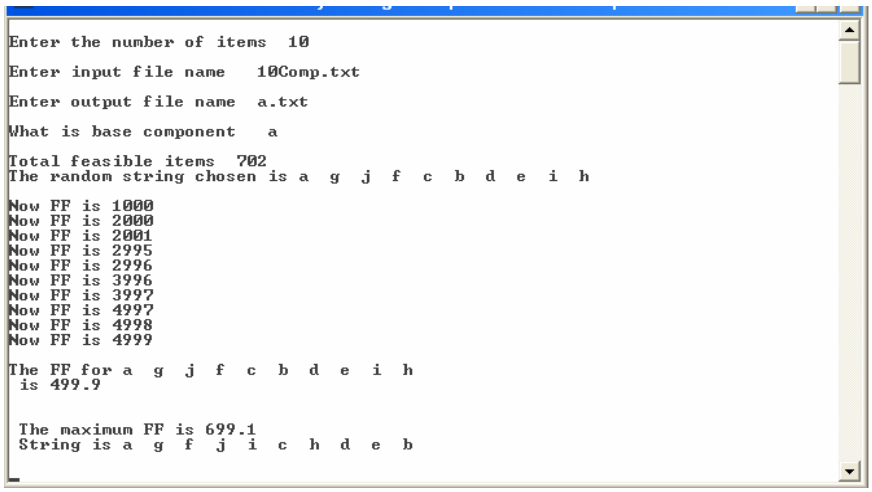

Figure-2. Optimal assemble sequence generated.

CONCLUSIONS

Genetic Algorithms are, conceptually, suitable to optimize the ASP problem. GA were selected in this research to optimize the ASP problem based on their qualities and especially because they can easily handle large search spaces and the flexibility in defining the constraints and the quality measures and derive them in a fitness function. GA wasn't successful in optimizing assembly sequence plans that involved large assemblies. Once the number of components are reduced, GA can be successfully applied to generate the optimal or near optimal assembly sequence plans.

A penalty function approach has been used in this paper to compute the fitness value for assembly sequences. The penalty function approach is chosen because the penalties are easy to define, realistically capture the difficulties associated with the assembly process and the number of penalties to consider is relatively reduced. The value of those penalties, from experience, is consistent with the difficulty to perform the related operations at the shop-floor level. The evaluation of the penalty function is simple and straightforward, a most desirable feature for a population-based search. In addition, by employing the idea of penalty, other critical factors concerning assembly planning, such as the frequency of changing tools, the similarity of assembly operations, the quality of assembly, the complexity of assembly, etc., can be considered simultaneously.

REFERENCES

[1] Bonneville, F., Perrard, C. and Henrioud, J.M. 1995. A genetic algorithm to generate and evaluate assembly plans. IEEE Symposium on Emerging Technology and Factory Automation. 2: 231-239.

[2] Goldberg, D. E. 1989. Genetic Algorithms in Search, Optimization, and Machine Learning. Addison-Wesley, USA.

[3] Sebaaly, M.F. and Fujimoto, H. 1996. A Generic Planner for Assembly Automation. In: Proceedings of the IEEE conference on Evolutionary Computation. pp. 401- 406.

[4] Chen, S. F. 1998. Assembly Sequence Planning-a genetic approach. In: Proceedings of the 24th ASME Design Automation Conference, September 12-16, Atlanta.

[5] Arun Mathew and Rao C.S.P. 2010. A CAD System for Extraction of Mating Features in an Assembly. Assembly Automation. The international Journal of assembly technology and management. 30(2): 142-146.

[6] Arun Mathew and Rao C.S.P. 2010. A Novel Method of using API to Generate Liaison Relationships from an Assembly. Journal of Software Engineering and Applications. 3(2): 167-175.