A Novel Approach on Optimal Assembly Sequence Planning of Fixture Assembly using Genetic Algorithm

E.Raj kumar and K.Annamalai
1 VIT University, Vellore campus, Tamil nadu,632014, India.
2 VIT University, Chennai campus, Tamil nadu, India.
Email: rajkumar.e@vit.ac.in, annamalaik@vit.ac.in

Abstract—Assembly planning is very important for competitive manufacturing where assemble-to-order of products is in practice. Assembly planning is a complex task and an optimal assembly plan is detrimental to meet customer demands. Different approaches have been attempted to generate optimal assembly plan. Researches for these types of combinatorial problems have attempted random search methods. This paper details the genetic algorithm application for generating optimal assembly sequence plans of fixture assembly.

Index Terms—Genetic algorithm, assembly sequence planning, optimization

I. INTRODUCTION

Assembly planning is a combinatorial problem. The number of assembly plans is a factorial function of the number of components in the product. The selection of assembly plans has a bearing effect on the final product. The choice of the assembly sequence in which parts or subassemblies are put together in the mechanical assembly of a product can drastically affect the efficiency of the assembly process. Hence an efficient assembly plan, greatly determines lead-time, production cost, and, thus, potential product success. Jezernik et.al [4] described that the assembly-planning problem involves the identification, selection and sequencing of assembly operations stated as their effects on the parts. The identification of assembly operations usually leads to the set of all feasible assembly plans. Besides it also relies on other factors like how the single parts are interconnected in the whole assembly, i.e. the structure of the graph of connections. Two kinds of approaches have been used for the optimal assembly plan. One, the more qualitative, uses rules in order to eliminate assembly plans that include difficult tasks or awkward intermediate subassemblies. A more quantitative approach uses an evaluation function that computes the merit of assembly plans. Both disassembly and reassembly sequences can be generated by searching the solution trees in the And/Or graph, leads in planning repair sequences [3]. The selection of sequences involves a trade-off between speed of plan generation and the quality of plan generated, where the quality is measured by some criterion used to compare plans. Homem de Mello “et.al” [3] analysed five representations of assembly sequences, which are based on directed graph, on And/Or graphs, on establishment conditions, and on precedence relationships. The identification of assembly operations usually leads to the set of all feasible assembly plans. The number of them grows exponentially with the number of parts, and depends on other
factors, such as how the single parts are interconnected in the whole assembly; represented in the graph of connections. Bai and Rong [1] developed a method to represent modular fixture elements. A modular fixture element assembly relationship database (MFEARDB) was established based on assembly features and mating relationships. Yi et al. [9] described a fastener-based approach in fixture assembly planning. Part Liaison Graph (PLG) and Fixture Liaison Graph (FLG) were generated to represent the assembly relationships. Dai et al. [2] presented a modular fixture design and assembly system based on geometric analysis. A modular fixture tower database (MFTD) was generated to represent the assembly relationship of fixture elements. The fixture towers were selected and placed in the assembly. Ma et al. [6] presented a computer-aided fixture design system named FIX-DES. Feature-based representation was used to describe the contact surface and the functionality of fixture elements. The assembly relationships between two fixture elements were automatically generated and rules were derived for the reasoning. The output of the system was a list of fixture elements and 3D drawings of fixture design. Peng [7] developed a VR-based modular fixture design system in which the user is allowed to manipulate the virtual models in a virtual environment for the fixture design and assembly. Rajan et al. [8] presented a Virtual Reality (VR)-based motion tracking system to evaluate product assembly sequence and Jig design. The ergonomic analysis is performed by evaluate the recorded user hand’s motion in the assembly. However, this method needs human intervention to assemble the product which takes a lot of time and efforts. Kang and Peng [5] developed a Web-based fixture assembly planning system. The fixture assembly sequence is automatically generated based on geometry reasoning.

II. PROBLEM DESCRIPTION

Increasing competition has forced the manufacturers to seek ways and means of cutting down the production costs, improving the product quality, and reducing the manufacturing lead times. Hence the selection of assembly plans has a bearing effect on the final product. The process of joining parts together to form a unit is known as assembly. The joining process results in the connection of one part with parts already assembled. A sub-assembly is a group of parts having the property of being able to be assembled independently of other parts of the product. An assembly plan is a set of assembly tasks with ordering amongst its elements. Each task consists of joining a set of sub-assemblies to give rise to an ever-larger sub-assembly. An assembly sequence is an ordered sequence of the assembly tasks satisfying all the ordering constraints. Each assembly plan corresponds to one or more assembly sequences. The genetic algorithm entails difficulty in encoding an assembly plans to chromosome for initial population generation, the randomness of crossover and mutation rules, which needs a still attention to develop an easy method. The work is focused on choosing an optimal or near-optimal assembly plan. The Assumptions involved in this work is of same that has been proposed by Homem de Mello et.al [3] comprises of the following: Exactly two parts or sub-assemblies are joined at each time, whenever parts are joined forming a subassembly, all contacts between parts in that subassembly are established, the feasibility of joining two subassemblies is independent of how those sub-assemblies were built.

III. METHODOLOGY

The proposed model starts from conceiving a product with recognizing all the parts and their connections. A liaison graph is built based on the connection of parts i.e. geometric constraints and their ordering amongst them. The precedence relations are derived from the constraints imposed on a component when located in the final assembly. The product is a fixture assembly product shown in figure 1. A component can be either completely constrained or partially constrained, when the component is assembled along at least one mating direction. Next, identify the number of possible assembly pair’s, which represents assembly task. A purposely-developed crossover and mutation is used to manipulate the chromosome, representing the solution for the formulated problem.

A. Product modelling

The first stage to search the assembly plans for a given product is to have a good representation of the product. This is obtained from liaison graph or graph of connection. It’s a relationship between two parts which are touching or effectively touching, whether physically attached or not”. In Liaison graph, parts are dots, joints are lines. A product ‘P’ built from a set of component ‘C’ is modeled by its liaisons graph [C, Γ], where ‘Γ’ is the set of the mechanical liaisons that exist between the components. A fixture assembly, constitutes seven components (node), linked by five liaisons (connections). Once the liaisons graph is built, possible assembly pairs are derived from the graph.
The Assumptions involved in this work comprises of the following:

a. Exactly two parts or sub-assemblies are joined at each time.

b. Whenever parts are joined forming a subassembly, all contacts between parts in that subassembly are established.

c. The feasibility of joining two sub-assemblies is independent of how those sub-assemblies were built.

B. Mathematical Model of a fixture assembly

By applying Set Theory, a fixture body can be defined as a set or an assembly of fixture elements.

Let $F$ denote a fixture and $e_i (i = 1, 2...n_e)$ a fixture element in $F$, where $n_e$ is the number of fixture elements in $F$, i.e.

$$F = \{ e_i | i \in n_e \}$$  

(1)

This is a representation of a fixture at the level of fixture elements.

A fixture consists of several sub-assemblies. Each sub-assembly performs one or more fixturing functions (usually one). These kinds of sub-assemblies in a fixture are considered as fixture functional units. In a fixture unit, all elements are connected one with another directly where only one element is connected directly with the baseplate and one or more elements in the subset are contacted directly with workpiece serving as locator, clamp or support.

Let $U_t$ denote a fixture unit in a fixture.

$$U_t = \{ e_{ij} | i \in n_{et} \}$$  

(2)

Where $n_{et}$ is the number of elements in unit $U_t$.

Therefore a representation of a fixture at the level of fixture units can be written in the following way:

$$F = \{ U_t | t \in n_u \}$$  

(3)

$$F = \{ e_{ij} | i \in n_{et} \}$$  

(4)

Where $n_u$ is the number of units in fixture ‘F’.

$$G = (V, E)$$  

(5)

$$V = \{ v | v \in \text{fixture elements} \}$$  

(6)

$$E = \{ e | P(v_i, v_j) \land (v_i, v_j) \in V \}$$  

(7)

where $V$ is a set of vertices representing fixture elements used in building a specific fixture unit; and $E$ is a set of directed pairs of members of $V$ and is an edge representing the assembly relationship between fixture elements (i and j).
IV. GENETIC ALGORITHM

GAs is a stochastic search algorithm that attempts to mimic natural biological processes that resulting genetic variation from generation-to-generation in living species. The description of the genetic algorithm concepts for the assembly-planning problem is presented in this section.

A. Generation of an initial population

The first step in developing a genetic algorithm for assembly planning is to map the problem solutions (assembly sequences) to chromosomes. The chromosome must be encoded in such a way that it contains the information about the solution which it represents. Hence ‘n’ strings will be created randomly. The each node corresponds to a gene that represents tasks number in a particular level, required to build an assembly along with other tasks in that particular levels of the plan.

<p>| | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>P1</td>
<td>2</td>
<td>9</td>
<td>14</td>
<td>27</td>
<td>28</td>
</tr>
<tr>
<td>P2</td>
<td>3</td>
<td>14</td>
<td>25</td>
<td>31</td>
<td>27</td>
</tr>
</tbody>
</table>

B. Objective function

The objective function for the problem is to obtain an optimum or near-optimal assembly plans based on the minimization of the makespan for a given product. The objective function is defined as follows

\[ m = P - 1 \]  
\[ P_t (n) = OC (n) + DoF (n) \]  
\[ C_t (AP) = \sum_{i=1}^{m} P_t (i) \]  
\[ MS = \text{Min } [C_{tj}] \]

Where: \( m \): Number of tasks in each assembly Tree, \( P \): Number of parts, \( P_t (n) \): Processing time for the task ‘n’ as i denote number of task required to complete a final product, \( n \): Number of tasks represents an assembly pair obtained from the precedence relations of liaisons graph, \( OC (n) \): Operational complexity (based on DFA method), \( DoF (n) \): Subassembly Degrees of freedom, \( C_t \): Completion time for each Assembly plan as ‘j’ denotes population size : 20. \( MS \): Makespan for each number of Assembly plans.

C. Evaluate the fitness function

The fitness function is normally used to transform the objective function value in to a measure of relative fitness. Since the objective is a minimization problem, the objective function value itself is used as fitness function. The fitness function must be calculated for the initial set population and after performing genetic operation so as to evaluate the fitness for each string. The significance of evaluating the fitness value for each string provides information to decide whether the strings are carried for next iteration (generation) to obtain the desired result.

D. Selection for reproduction

There are various methods to select the chromosomes for reproduction. Some are Roulette wheel selection, Boltzmann selection, Tournament selection, Rank selection, Steady state selection etc. But in this work the Rank selection method is utilized.

E. Rank selection

Once after an initial population is generated, fitness value is computed for each string in that population. This leads the chromosomes having an equal chance to be selected. But, this method leads to slower convergence, because; the best chromosomes do not differ much from the other ones. Selection is the process of determining the number of times, or trials, a particular individual are chosen for reproduction and, thus, the number of offspring that an individual will produce.

F. Genetic operator

Two families of genetic operators purposely have been developed for searching the whole solution space. The first includes operators that search locally for new sequences from parent chromosomes. The other family of operators is intended to search for sequences in assembly plans of parents as well as the crossbreed chromosomes. This basically made by introducing a new task in a solution, and substituting certain tasks in order to maintain the validity of the chromosomes.
Crossover:- The basic operator for producing new chromosomes in the GA is that of crossover. Like its counterpart in nature, crossover produces new individuals that have some parts of both parent's genetic material. The simplest form of crossover is that of single-point crossover. In this approach strings are subjected to crossover based on the specified crossover probability. The probability of crossover is usually high and ranges from 0.6 to 0.8 and this has been considered in the problem.

\[
\begin{array}{cccccccc}
P1 & 3 & 14 & 14 & 27 & 28 & 31 \\
P2 & 2 & 9 & 25 & 31 & 27 & 28 \\
\end{array}
\]

Mutation:- Mutation is a random process where one allele of a gene is forcibly replaced by another to produce a new genetic structure. In GA, mutation is randomly applied with low probability '\(P_m\)', typically in the range 0.001 and 0.05, and modifies elements within the chromosomes.

\[
\begin{array}{cccccccc}
P1 & 3 & 14 & 14 & 31 & 28 & 31 \\
P2 & 2 & 9 & 25 & 27 & 27 & 28 \\
\end{array}
\]

V. CASE STUDY

To validate the proposed methodology a simple product is considered. The product is a fixture assembly product (rectilinear workpiece), consists of 7 subassemblies. The solution is represented in Figure 2. Thus, the proposed genetic algorithm is applied for the product and results are discussed. The coding for the same is done using visual studio C++ language.

![Figure 2. Representation of solution (and/or graph).](image)

GA Coding:- GA coding was developed by using visual studio C++ program. The following input parameters used for generating optimal assembly sequence. Table 1 represents the best assembly sequence.

**Input parameters:**
- Number of parts: 7
- Number of levels: 8
- Number of tasks: 31
- Population size: 20

**Output:** The best solution is

<table>
<thead>
<tr>
<th>Task</th>
<th>Assembly sequence</th>
<th>Processing Time(seconds)</th>
</tr>
</thead>
<tbody>
<tr>
<td>11</td>
<td>3 12 11 24</td>
<td>23</td>
</tr>
<tr>
<td>12</td>
<td>6 2 3 28</td>
<td>23</td>
</tr>
</tbody>
</table>

**Table I: Best Assembly Sequence**

797
VI. CONCLUSION

In This Work, a Genetic Algorithm is proposed for scheduling an assembly task, with the objective as the minimization of makespan. It starts from the liaisons graph, which forms the basis for constructing precedence matrix. Further this precedence matrix and geometric constraints are used to generate a feasible assembly plans (or) to verify for feasibility, represents an initial set of population to carry out genetic operation. The assembly plans are evaluated based on makespan, which includes operational complexity and degrees of freedom. A purposely-developed genetic operator is used to perform crossbreed and mutation function. The best solution is 6 238121124 and the corresponding processing time is 23 seconds. Thus a Genetic Algorithm is proposed to solve an assembly sequence problem, a much more difficult problem than other sequencing problems. The algorithm responds quickly to produce a set of good assembly plans, but entails difficult in generating all the valid ones. The proposed methodology can be adapted to assemblies where computational time is significant.

REFERENCES


