**AUTOMATIC PRECEDENCE CONSTRAINT GENERATION FOR ASSEMBLY SEQUENCE PLANNING USING A THREE DIMENSIONAL SOLID MODEL**

Alfadhlani1[[1]](#endnote-2), T. M. A. Ari Samadhi2, Anas Ma’ruf 2, Isa Setiasyah Toha2

1*Industrial Engineering Department, Andalas University, Limau Manis, Padang, 25163, Indonesia*

2 *Industrial Engineering Department, Bandung Institute of Technology, Jl. Ganesha 10, Bandung, 40132, Indonesia*

(Received: Month Year / Revised: Month Year / Accepted: Month Year)

ABSTRACT

Assembly sequence planning of a product can be generated through three phases: the first phase is generating precedence constraints, the second phase is searching assembly sequence alternatives, and the last is selecting the best assembly sequence. Assembly sequences generation needs the precedence constraints to find the feasible assembly. A collision between two components can cause blocking of one component by the other one when assembled. This research proposes an automated method for generating precedence constraints. This method uses some information, namely: the collision-free assembly path, the number of connections between components, and the volume of the component. The information is extracted from the CAD (Computer Aided Design) database. The methods resulted in this research will be used for developing an automated way of assembly sequence generation using the three-dimensional (3D) Solid drawing in the form of stacked drawing in a CAD system.

*Keywords:* Assembly sequence; CAD; Collision detection; Precedence constraint

**1. INTRODUCTION**

An assembly is a process for joining of parts to form completed product that needs to be evaluated as early as possible in the product design stage so that a component will not be difficult to install because of tolerance error, inappropriate dimensions, and geometry errors. A product designer needs to improve the design if assembly difficulties occur. This additional task will increase the production cost. The assembly is essential to be planned because there are some feasible assembly sequence alternatives which can be selected based on dimension and geometry. Proper assembly sequence will reduce the operation difficulty, tool quantity and work hours, so it will also reduce production cost (Lai and Huang, 2004).

The designer can evaluate the assembly process in the early stage of the design using CAD (Computer Aided Design). Some previous researchers have proposed assembly sequence generation method based on CAD system such as Delchambre (1992), Ariastuti et al., (1998), Tseng and Li (1999), Toha et al., (2004), Alfadhlani and Toha (2005; 2008). Some of this research has proposed an automatic method to generate the assembly sequence.

The possible assembly sequence for complex product assembly planning is searched based on the precedence relation information (Lai and Huang, 2004). It contains information about the list of components that must be assembled beforehand (predecessor) and the choice of components to be joined later (successor). In the assembly planning, the precedence relation has to be determined to ensure that the planned assembly operation can be applied. Therefore, the generating geometric model of assembly product must be generating precedence constraints.

Delchambre (1992) distinguished the precedence constraints in two types: hard constraints and soft constraints. Hard constraints arise because of the geometry condition of components and their position in the final assembly. Soft constraints are stacking constraints and technological constraints. Stacking constraints are if external fasteners (such as screws) hold together a stack of other components, so it is best to impose a given assembly sequence for this group of components. While technological constraints are specified by the operator, arising because of the use of specific tools. Soft constraints are recommended for consideration. If the generated assembly sequences are feasible without considering the soft constraints, these constraints can be ignored.

Ariastuti et al., (1998), and Toha et al., (2004) used "face and joint" information as precedence constraints which are determined from the assembly line in CAD system, while Li et al., (2010) identified the precedence constraints by using connectors knowledge-based approach, they used standard connectors such as threaded fasteners or keys. Morato et al., (2013) generated precedence constraints based on components motion planning and components interaction cluster that can mutually affect each other’s accessibility when assembled. All three methods above used the disassembly approach that required a complex geometry analysis.

The feasible assembly must be free of collisions between components when assembled. The collision is a condition of a component blocked by another component when assembled. The collision-free assembly path (CFAP) information in this paper is detected by using a CFAP algorithm proposed by Alfadhlani et al., (2011). This paper discusses the development of an automated method for generating precedence constraint, using the disassembly approach, by considering the CFAP information, the number of component connections, and the component volume. All these data are extracted from the CAD database using the component database formation algorithm proposed by Alfadhlani et al., (2011). The SolidWorks 2005 is used as a CAD system, and the stacked drawing in a 3D solid model is used as input.

The rest of this paper is divided into some sections. Section 2 elaborates the rules of precedence diagramming. Section 3 explains the process of generating precedence constraint based on a collision-free assembly path. Section 4 mentions the priority rules for selecting the component to be released. Section 5 discusses the use of associative law as a rule for improving precedence constraint. Section 6 describes the algorithm of precedence constraint generation. Section 7 provides an example of the implementation of proposed methods for generating precedence constraint, and the conclusions are given in Section 8.

**2. METHODS**

**2.1. Precedence Diagram**

A precedence constraint can be represented in the form of a network or arrow diagram. See Figure 1a, a node represents a component to be assembled; an arrow represents an assembly activity that is connecting node-*i* and node-*j*. Arrow length is not proportional to the duration of activity. Activity (3, 4) cannot be done before activities (1, 3) and (2, 3) are completed, see Figure 1b.

|  |  |
| --- | --- |
|  |  |
| (a) | (b) |

Figure 1 Precedence diagram

The precedence diagram is formed by taking into account the following matters:

1. Each assembly activity is represented by one and only one arrow in the diagram
2. Two or more activities are not allowed to connect the same two nodes
3. The correct precedence relations in the diagram is ascertained by answering some questions
4. Which activities must be completed before the observed activity can be done?
5. Which activities must follow the observed activities?
6. Which activities can be done in parallel with the observed activity?

**2.2. Precedence Constraint Generation**

The precedence constraint is used to guarantee the feasibility of the generated assembly sequence, and it is generated by considering CFAP. The CFAP information is obtained by using the automatic method proposed by Alfadhlani et al., (2011). The precedence constraint generation in this paper is carried out with the following three steps: 1) determine the components to be released based on the CFAP; 2) do the disassembly test, and 3) re-evaluate the CFAP value without involving the components that have been released.

The components to be released is determined based on CFAP value which equals to 1 at least in one direction of the coordinates system. The sign in the direction of CFAP must be changed if using the disassembly approach. See Table 1, the evaluation of the CFAP value with a logical AND, results in the value equal to 1, found in the component-1 in direction +*y* and component-3 in direction –*y*. If components 1 and 3 are released, then its direction must be changed to the opposite one. Component-1 can be released to direction –*y*, and component-3 to direction +*y*.

Table 1. Collision-free assembly path (Alfadhlani et al., 2011)

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Component | Contact | *+x* | *-x* | *+y* | *-y* | *+z* | *-z* |  |
| 1 | 1,2 | 1 | 1 | 1 | 0 | 1 | 1 |
| 1,3 | 0 | 0 | 1 | 0 | 0 | 0 |
| AND | 0 | 0 | 1 | 0 | 0 | 0 |
| 2 | 2,1 | 1 | 1 | 0 | 1 | 1 | 1 |
| 2,3 | 0 | 0 | 1 | 0 | 0 | 0 |
| AND | 0 | 0 | 0 | 0 | 0 | 0 |
| 3 | 3,1 | 0 | 0 | 0 | 1 | 0 | 0 |
| 3,2 | 0 | 0 | 0 | 1 | 0 | 0 |
| AND | 0 | 0 | 0 | 1 | 0 | 0 |

If component-3 is selected to be released, then the results of re-evaluation of its CFAP are shown in Table 2. Component-1 can be released to all directions except to direction +*y*, whereas component-2 can be released to all directions except to direction –*y*. It can be stated that component-3 is the disassembly predecessor of components-1 and 2. So, if component-*i* can be released after component-*j*, then component-*j* is the predecessor of component-*i*.

Table 2. Re-evaluation of collision-free assembly path

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Component | Contact | *+x* | *-x* | *+y* | *-y* | *+z* | *-z* |
| 1 | 1,2 | 1 | 1 | 1 | 0 | 1 | 1 |
| AND | 1 | 1 | 1 | 0 | 1 | 1 |
| 2 | 2,1 | 1 | 1 | 0 | 1 | 1 | 1 |
| AND | 1 | 1 | 0 | 1 | 1 | 1 |

**2.3. Component Selection for Disassembly**

Disassembly approach is used to generate precedence constraints. The CFAP must be re-evaluated every time a component released. If there are more than one candidate components that can be released, the priority rules for selecting them are needed.

The feasibility of assembly sequences is guaranteed by using the CFAP information, while the stability is secured using the priority rules. Assembly operations need stability because probably an operator will often do a reorientation of the product during assembly. Alfadhlani and Toha (2005) generated assembly sequences using the criteria of the number of mating between components that have been installed and that has not been installed, and the volume of the component. Mating is a contact between the components of a product, defined referring to the direction of the normal vector and the shape of the contacted face.

The priority criteria based on mating is related to the connection between components. The connection is defined as a relationship between components in the final assembly, known with the presence of mating. The connection may have more than one mating. In this paper, the number of mating criterion is changed with the number of connections, while the volume criterion is used as Alfadhlani and Toha (2005) proposed. This research uses the disassembly approach, but Alfadhlani and Toha (2005) used the assembly approach, so that component selection priority is not based on the largest volume but the smallest volume.

The following criteria for selecting a component are proposed, if there are some disassembly component candidates:

1. Select the component that has the least of connections with other components that have not been selected yet.
2. Select the candidate component that has the smallest volume and meets the criteria of point 1.
3. Select a component arbitrarily among the components that meet priority criteria in points 1 and 2, if there are more than one candidates.

The selection criteria in point 1 consider the close relationship between the next component to be assembled and the components that have been assembled, as proposed by Alfadhlani and Toha (2005). The closeness of the relationship is related to the connection and the stability between components in the final assembly. Because the disassembly approach is used in this research, then the next component to be released is that it has the least number of connections with components that have not been released yet. Point 2 is used if criteria in point 1 cannot find a unique candidate.

De Fazio and Whitney (1987) formed sub-assembly and assembly sequence using the criteria of the maximum number of actions completed per-assembly operation and the maximum number of liaisons completed per-assembly operation. While Ariastuti et al., (1998) used the number of exploded assembly line criteria. Both of these criteria are the same as the number of connections criterion which is proposed in this research. The liaison is a representation of the connection between the components, while the exploded assembly line exists because of the connections between the components. Ariastuti et al., (1998) prioritized that the component to be assembled first was the one that had the highest number of the exploded assembly line. It is similar to the idea of making a choice based on the highest number of connections

See Figure 2, component-1 and component-3 are candidates for disassembly and have the same amount of connections. Using the above priority criteria, then component-3 is selected as the component to be disassembled first because it has a smaller volume than the component-1.



Figure 2 The number of connections and volume of components

The proposed selection criteria are formulated in the form of Rules A and used in the algorithm of precedence constraint generation. The priority rules for selecting candidates are as follows:

**Rule A**: Priority rules for selecting candidates components

1. If there is more than one candidate that can be released, then choose a component that has the least connections with components that have not been selected yet.
2. If the rule in point 1 obtains more than one candidate, then select a candidate component that has the smallest volume.
3. If the rules in points 1 and 2 obtain more than one candidate, then select a component arbitrarily that meets the rules in points 1 and 2.

**2.4. Precedence Constraints Improvement**

If there are components 1, 2, 3, and 4, with the following precedence relations:

1. 4 precedes 3
2. 3, 4 precede 2
3. 3 precedes 1

Then points *a* through *c* above is determined based on a particular direction for the six-axis observed. The following additional criteria are needed to obtain a complete precedence constraint:

1. For component-*i*, which is a predecessor of component-*j*, then predecessors of component-*i* are also as predecessors of component-*j*. For the above example, component-3 is a predecessor of component-1, thereby the component-4 that is a predecessor of component-3, also as the predecessor of component-1. So, the precedence relation of component-1 is refined into 3, 4 precede 1.
2. For all the predecessors of component-*i*, which are the predecessors of component-*j*, while component-*i* connected with component-*j*, then the component-*i* is also the predecessor of component-*j*. For the above example, the whole predecessors of component 2 are the predecessor of component-1, as obtained in step 1. Component- 2 is connected to component-1, then component-2 is also a predecessor of component-1. The precedence relation of component-1 can be refined into 2, 3, 4 precede 1.

The criteria for improving the precedence relation here are formulated in Rules B using the associative law as follows:

Rule B: Precedence constraints improvement

1. If component-*i* is a predecessor of component-*j*, then all the predecessors of component-*i* are also the predecessors of component-*j*.
2. If all the predecessors of component-*i* are the predecessors of component-*j*, and component-*i* is connected with component-*j*, then component-*i* is also the predecessor of component-*j*.

**2.5. Precedence Constraint Generation Algorithm**

An automatic method for generating the precedence constraint was developed in this research based on the information, namely: the collision-free assembly path (CFAP), the number of connections, and the volume of components. All this information can be determined based on the mating conditions data which are obtained from the CAD database. Alfadhlani et al., (2011) has proposed an automatic method to defined the assembly collision-free path, and that method was adopted in this research.

The CFAP is determined using the Component Collision Detections Algorithm (Alfadhlani at al., 2011). The data required as input are taken from the database of product drawing on the *SolidWorks* 2005 CAD system and stored in two databases, namely: Component Database, and Collision-Free Assembly Path (CFAP) Database. The Component Database contains the information about a list of all pairs of contacted components and their mating type, the volume of each component, the coordinates of the point of contact, and their normal vector directions; all these information are extracted by using the Component Database Algorithm (Alfadhlani at al., 2011). While the CFAP Database contains the same information as the Component Database, but are added with the data of the number of connections and the CFAP information of each component. See Figure 3.



Figure 3 A frameworks for generating precedence constraints

Furthermore, the priority rules for selecting components, the rules for improving precedence constraint, and the precedence constraint generation steps, that have been described above, are used in constructing the following Precedence Constraints Algorithm:

1. Iteration = 0
2. Set Selected Component List = Ø.
3. Identify CFAP of each component in the CFAP Database that has a value equal to 1 at least on one axis and set as a candidate, saves the candidates, its connections and its volume in Candidate Component List.
4. Iteration = Iteration +1.

Check the candidates and its connections in the Candidate Component List. Calculate *a* is the number of candidate connections with other components in the Selected Components list. Calculate *b* is the number of candidate connections with other components that have not been selected, *b* = total connections – *a*.

1. Use **Rule A**:

Select a component to be released in Candidate Component List. That is the candidate that has the smallest value of *b*. If there are more than one candidates, select the candidate which has the smallest volume. If there are still more than one candidates, choose a component arbitrarily among candidates which have the smallest value of *b* and the smallest volume. Save the component and the number of connections in Selected Components List and delete from Candidates List.

1. Set all selected components previously and connect to the new one as its predecessors.
2. Delete the new selected component from CFAP Database. Then re-evaluate the value of CFAP of each component.
3. Check all components in CFAP Database which have CFAP value equal to 1, set as a new candidate and save in Candidates List.
4. Check the Candidates List, if there are components that have not been evaluated, go back to step 2, otherwise, go to the next step
5. Update the Predecessor List of the last selected component by adding all components which connect to it.
6. Update the predecessor of the component using **Rules B**:

For all components-*j* and *j* = 1, 2… *N*:

* + 1. If component-*i* is a predecessor of component-*j*, then all predecessors of component-*i* are also predecessors of components-*j*.
    2. If all predecessors of component-*j* are the predecessors of component-*k*, and component-*j* connected with component-*k*, then component-*j* are also a predecessor of component-*k.*

1. Save the information of components and its predecessors on Precedence Constraint List in the Component Basis Data, then stop iteration

The result of this algorithm is the precedence constraints, represented by the relationship of components and its predecessors. The result is saved in Component Data Base. At this stage, the Component Database contains information about a list of all pairs of the contacted components and their mating type, the volume of each component, coordinates of the point of contact, their normal vector directions, and predecessors list of each component. The flowchart of precedence constraints generation is shown in figure 4.



Figure 4 Precedence constraints generation flowchart

**3. IMPLEMENTATION EXAMPLE**

This research using the Bench Vice for testing the algorithm, this product is adopted from Tickoo (2004) and was redrawn for this requirement.Its assembly orientations are multidirectional and orthogonal to the *x*, *y*, and *z*. The Bench Vice consists of 13 components as shown in Figure 5.

|  |  |
| --- | --- |
|  |  |

Figure 5 Bench Vice Assembly (Tickoo, 2004)

The CFAP information, which is determined using the CFAP algorithm by Alfadhlani et al., (2011), is shown in Table 3. This information is used for generating the precedence constraints. Based on this information, it is known that the candidates to be released in the first iteration are components 5, 6, 7, 8, 9, 10 and 13, because these components have a value equal to 1 in the collision-free assembly path.

Table 3 CFAP value of each component of Bench Vice

(Initial data, using the method of Alfadhlani et al., 2011)

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Component | *+x* | *-x* | *+y* | *-y* | *+z* | *-z* |
| 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 2 | 0 | 0 | 0 | 0 | 0 | 0 |
| 3 | 0 | 0 | 0 | 0 | 0 | 0 |
| 4 | 0 | 0 | 0 | 0 | 0 | 0 |
| 5 | 0 | 0 | 1 | 0 | 0 | 0 |
| 6 | 0 | 0 | 1 | 0 | 0 | 0 |
| 7 | 0 | 0 | 1 | 0 | 0 | 0 |
| 8 | 0 | 0 | 1 | 0 | 0 | 0 |
| 9 | 0 | 0 | 1 | 0 | 0 | 0 |
| 10 | 0 | 0 | 1 | 0 | 0 | 0 |
| 11 | 0 | 0 | 0 | 0 | 0 | 0 |
| 12 | 0 | 0 | 0 | 0 | 0 | 0 |
| 13 | 0 | 0 | 0 | 1 | 0 | 0 |

Table 5 shows the priority value, disassembly direction and overall iterations for generating the precedence constraints of Bench Vice. See iteration 1, the candidate components that will be released are 5, 6, 7, 8, 9, 10 and 13. Because all candidates have the same value of *b* (the number of candidates connection with components that have been released), then select the candidate with the smallest volume that is component-13. After component-13 is released then update CFAP, the results are shown in Table 4. Furthermore, the component-3 is entered as a candidate.

Table 4 CFAP value of each component of Bench Vice

(Data after component 13 is selected, using the method of Alfadhlani et al., 2011)

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Component | *+x* | *-x* | *+y* | *-y* | *+z* | *-z* |
| 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 2 | 0 | 0 | 0 | 0 | 0 | 0 |
| 3 | 0 | 1 | 0 | 0 | 0 | 0 |
| 4 | 0 | 0 | 0 | 0 | 0 | 0 |
| 5 | 0 | 0 | 1 | 0 | 0 | 0 |
| 6 | 0 | 0 | 1 | 0 | 0 | 0 |
| 7 | 0 | 0 | 1 | 0 | 0 | 0 |
| 8 | 0 | 0 | 1 | 0 | 0 | 0 |
| 9 | 0 | 0 | 1 | 0 | 0 | 0 |
| 10 | 0 | 0 | 1 | 0 | 0 | 0 |
| 11 | 0 | 0 | 0 | 0 | 0 | 0 |
| 12 | 0 | 0 | 0 | 0 | 0 | 0 |

This process is continued until 13 iterations. If all components have been chosen to be released, then the iteration is stopped.

The final step is the improvement of the predecessor list of components using Rule B. Since the components-3, and 4 are predecessors of component-1, then predecessors of components 3 and 4 are also predecessors of component-1. Thus predecessors of component-1 are 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, and 13.

Table 5 Precedence constraints generation of Bench Vice

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Iteration | Candidate | Connection components | *a* | *c* | *b* | Volume (mm3) | Selected component | New candidate | Disassembly direction | Predecessors |
| 1 | 5 | 2,4 | 0 | 2 | 2 | 804.08 | 13 | 3 | *-y* | - |
|  | 6 | 2,4 | 0 | 2 | 2 | 804.08 |
|  | 7 | 1,11 | 0 | 2 | 2 | 545.8 |
|  | 8 | 1,11 | 0 | 2 | 2 | 545.8 |
|  | 9 | 1,12 | 0 | 2 | 2 | 545.8 |
|  | 10 | 1,12 | 0 | 2 | 2 | 545.8 |
|  | 13 | 2,3 | 0 | 2 | 2 | 483.06 |
| 2 | 5 | 2,4 | 0 | 2 | 2 | 804.08 | 7 | empty | *-* | - |
|  | 6 | 2,4 | 0 | 2 | 2 | 804.08 |
|  | 7 | 1,11 | 0 | 2 | 2 | 545.8 |
|  | 8 | 1,11 | 0 | 2 | 2 | 545.8 |
|  | 9 | 1,12 | 0 | 2 | 2 | 545.8 |
|  | 10 | 1,12 | 0 | 2 | 2 | 545.8 |
|  | 3 | 1,2,13 | 1 | 3 | 2 | 25334.69 |
| 3 | 5 | 2,4 | 0 | 2 | 2 | 804.08 | 9 | empty | *-* | - |
|  | 6 | 2,4 | 0 | 2 | 2 | 804.08 |
|  | 8 | 1,11 | 0 | 2 | 2 | 545.8 |
|  | 9 | 1,12 | 0 | 2 | 2 | 545.8 |
|  | 10 | 1,12 | 0 | 2 | 2 | 545.8 |
|  | 3 | 1,2,13 | 1 | 3 | 2 | 25334.69 |
| 4 | 5 | 2,4 | 0 | 2 | 2 | 804.08 | 8 | 11 | *+x, -x,*  *+y,+z, -z* | - |
|  | 6 | 2,4 | 0 | 2 | 2 | 804.08 |
|  | 8 | 1,11 | 0 | 2 | 2 | 545.8 |
|  | 10 | 1,12 | 0 | 2 | 2 | 545.8 |
|  | 3 | 1,2,13 | 1 | 3 | 2 | 25334.69 |
| 5 | 5 | 2,4 | 0 | 2 | 2 | 804.08 | 10 | 12 | *+x, -x,*  *+y,+z, -z* | - |
|  | 6 | 2,4 | 0 | 2 | 2 | 804.08 |
|  | 10 | 1,12 | 0 | 2 | 2 | 545.8 |
|  | 3 | 1,2,13 | 1 | 3 | 2 | 25334.69 |
|  | 11 | 1,4,7,8 | 2 | 4 | 2 | 16563.72 |
| 6 | 5 | 2,4 | 0 | 2 | 2 | 804.08 | 5 | empty | *-* | - |
|  | 6 | 2,4 | 0 | 2 | 2 | 804.08 |
|  | 3 | 1,2,13 | 1 | 3 | 2 | 25334.69 |
|  | 11 | 1,4,7,8 | 2 | 4 | 2 | 16563.72 |
|  | 12 | 1,4,9,10 | 2 | 4 | 2 | 16563.72 |
| 7 | 6 | 2,4 | 0 | 2 | 2 | 804.08 | 6 | 4 | *+x,-x* | - |
|  | 3 | 1,2,13 | 1 | 3 | 2 | 25334.69 |
|  | 11 | 1,4,7,8 | 2 | 4 | 2 | 16563.72 |
|  | 12 | 1,4,9,10 | 2 | 4 | 2 | 16563.72 |
| 8 | 3 | 1,2,13 | 1 | 3 | 2 | 25334.69 | 11 | empty | - | 7,8 |
|  | 11 | 1,4,7,8 | 2 | 4 | 2 | 16563.72 |
|  | 12 | 1,4,9,10 | 2 | 4 | 2 | 16563.72 |
|  | 4 | 1,2,5,6,11,12 | 2 | 6 | 4 | 6224.54 |

Table 5 *(Continued)*

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Iteration | Candidate | Connection components | *a* | *c* | *b* | Volume (mm3) | Selected component | New candidate | Disassembly direction | Predecessors |
| 9 | 3 | 1,2,13 | 1 | 3 | 2 | 25334.69 | 12 | empty | - | 9,10 |
|  | 12 | 1,4,9,10 | 2 | 4 | 2 | 16563.72 |
|  | 4 | 1,2,5,6,11,12 | 3 | 6 | 3 | 6224.54 |
| 10 | 3 | 1,2,13 | 1 | 3 | 2 | 25334.69 | 4 | empty | - | 5,6,11,12 |
|  | 4 | 1,2,5,6,11,12 | 4 | 6 | 2 | 6224.54 |
| 11 | 3 | 1,2,13 | 1 | 3 | 2 | 25334.69 | 3 | 1 | *+y* | 13 |
|  |  |  |  |  |  |  | 2 | *-y* |
| 12 | 1 | 2,3,4,7,8,9, 10,11,12 | 8 | 9 | 1 | 169238.64 | 2 | empty | - | 3,13,4,5,6 |
|  | 2 | 1,3,4,5,6,13 | 5 | 6 | 1 | 57452.1 |
| 13 | 1 | 1,3,4,5,6,13 | 9 | 9 | 0 | 169238.64 | 1 | empty | - | 2,3,4,11,7,  8,12,9,10 |

*a* = the number of candidate connections with components that have not been disassembled yet; *b* = the number of candidate connections with components that have been disassembled; *c* = the number of connections of candidate

Predecessors of component-4 are 5, 6, 11 and 12. Predecessors of component-11 are 7 and 8, while the predecessors of component-12 are 9 and 10. Thus, predecessors of component-4 are equipped into 5, 6, 7, 8, 9, 10, 11, and 12. The predecessor list which has been improved for each component of Bench Vice is shown in Table 6, while the assembly precedence diagram of Bench Vice’s components is shown in Figure 6.



Figure 6 Assembly precedence diagram of Bench Vice’s components

**4. CONCLUSION**

This research was conducted as part of an effort to help the assembly planners of product in generating the feasible assembly sequence automatically. Assembly precedence constraints were determined through the following steps: 1) Extract component geometric data from CAD system; 2) Build component database; 3) Detect collision of components; 4) Build collision-free path database; 5) Generate the precedence constraints. This paper proposes a fully automated method for generating precedence constraints using the disassembly approach. The 3D stacked drawing in a solid model is used as input because it has more information than 2D drawing. The exploded view and the assembly line to get connections information of each component as proposed by Ariastuti et al., (1998) and Toha et al., (2004), and also the analyzing of the components motion planning that proposed by Morato et al., (2013), did not need to be performed in this proposed method. Product designer has defined connections type between components when designed the product in a CAD system. In *SolidWorks*, the connection's type is defined by the mating type, so then the information of the mating type and the volume of the component is used in this method.

Table 6 Predecessors and CFAP value of Bench Vice’s components

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Component | *+x* | *-x* | *+y* | *-y* | *+z* | *-z* | *Predecessor* |
| 1 | 0 | 0 | 1 | 0 | 0 | 0 | 2,3,4,5,6,7,8,9,10,11,12,13 |
| 2 | 0 | 0 | 0 | 1 | 0 | 0 | 3,4,5,6,7,8,9,10,11,12,13 |
| 3 | 0 | 1 | 0 | 0 | 0 | 0 | 13 |
| 4 | 1 | 1 | 1 | 0 | 0 | 0 | 5,6,7,8,9,10,11,12 |
| 5 | 0 | 0 | 1 | 0 | 0 | 0 | - |
| 6 | 0 | 0 | 1 | 0 | 0 | 0 | - |
| 7 | 0 | 0 | 1 | 0 | 0 | 0 | - |
| 8 | 0 | 0 | 1 | 0 | 0 | 0 | - |
| 9 | 0 | 0 | 1 | 0 | 0 | 0 | - |
| 10 | 0 | 0 | 1 | 0 | 0 | 0 | - |
| 11 | 1 | 1 | 1 | 0 | 1 | 1 | 7,8 |
| 12 | 1 | 1 | 1 | 0 | 1 | 1 | 9,10 |
| 13 | 0 | 0 | 0 | 1 | 0 | 0 | - |

CAD system was used in the development of the models and the algorithm. The software used in this research is *SolidWorks* 2005. This CAD system has a quite complete feature, so that can show how the components are assembled to build the final product. We proposed an automatic precedence constraint method in this paper and developed two rules and an algorithm. We build the algorithm using the rules and the information of assembly collision-free path as input. The proposed method has been tested showing disassembly precedence constraints of product correctly. We have developed the prototype software in *SolidWorks* 2005 CAD system for implementation. This research is a part of an effort to propose an automated method in generating assembly sequence which is currently being developed.

**5. ACKNOWLEDGEMENT**

Alfadhlani thanks Institute of Research and Community Service, Andalas University with grant number 01/UN.16/PL/API/2014.

**6. REFERENCES**

Alfadhlani and Toha, I.S., (2005), Product assembly sequence generation using 3D solid drawing as input, Proceedings of Seminar VII Production System, 1–13, (in Indonesian).

Alfadhlani and Toha, I.S., (2008), Assembly sequence generation for multidirectional and orthogonal of product based on mating location point and volume of components, Jurnal Teknik Industri, 10(2), (in Indonesian).

Alfadhlani, Samadhi, T.M.A.A., Ma’ruf, A., Toha, I.S., (2011), Automatic collision detection for assembly sequence planning using a three-dimensional solid model, Journal of Advanced Manufacturing Systems, 10(2), 277–291.

Ariastuti, R., Taroepratjeka, H., Toha, I.S., (1998), Automatic assembly sequence generation of products, Jurnal Teknik dan Manajemen Industri 18(1), (in Indonesian).

De Fazio, T. L. and Whitney D. E., (1987), Simplified generation of all mechanical assembly sequences, IEEE Transactions on Robotics and Automation, RA-3(6).

Delchambre, A., (1992), Computer-aided assembly planning, Chapman & Hall, London, 1992.

Lai, H.Y. and Huang, C.T., (2004), A systematic approach for automatic assembly sequence plan generation, International Journal of Advanced Manufacturing Technology, 24, 752-763.

Li, J. R., Wang, Q.H., Huang, P., Shen, H.Z., (2010), A novel connector-knowledge-based approach for disassembly precedence constraint generation, International Journal of Advanced Manufacturing Technology, 49, 293–304.

Morato, C., Kaipa, K. N., Gupta, S.K., (2013), Improving assembly precedence constraint generation by utilizing motion planning and part interaction clusters, Computer-Aided Design, 45, 1349–1364.

Tickoo, S., SolidWorks for designers release 2004, (2004), CADCIM Technologies, Schererville, USA.

Toha, I.S., Baharuddin, F.R., Ma’ruf, A., (2004), Automatic assembly planning using the assembly line, Jurnal Teknik dan Manajemen Industri, 24(2), 15-27, (in Indonesian).

Tseng, H. E. and Li, R. K., (1999), A Novel Means of Generating assembly sequences using the connector concept, Journal of Intelligent Manufacturing, 10(5), 423-435.

1. Corresponding author’s email: alfadhlani@eng.unand.ac.id, Tel. +62-0751-72497, Fax. +62-0751-72566

   Permalink/DOI: https://doi.org/10.14716/ijtech.vxix.xxxx [↑](#endnote-ref-2)