Selective Disassembly: Representation and Comparative Analysis of Wave Propagation Abstractions in Sequence Planning

Hari Srinivasan\textsuperscript{1} and Rajit Gadh\textsuperscript{2}

\textsuperscript{1} Doctoral Candidate, hari\textsubscript{\textregistered}smartcad.me.wisc.edu
\textsuperscript{2} Associate Professor, gadh\textsubscript{\textregistered}engr.wisc.edu

Department of Mechanical Engineering,
1513 University Avenue, #347
University of Wisconsin-Madison,
Madison, WI - 53706, USA.

ABSTRACT

This paper analyzes the problem of removing one or more components of an assembly, defined as selective disassembly, and presents geometric abstractions and representations for automated selective disassembly analysis of geometric models.

Based on the number of target components to be disassembled from an assembly ($A$) of $n$ components, the selective disassembly ($SD$) problem is categorized into three classes. (I) Single component disassembly (1-SD): the motivation is that the disassembly analysis can be localized with respect to the target component and analysis of all components for disassembly may not be required. (II) Disassembling $s$ ($1 < s < n$) components from $A$ (defined as $s$-SD): the motivation is that a better solution may be obtained if two or more components are disassembled along a common sequence. (III) Disassembling $S$ ($S \rightarrow n$) components from $A$ (defined as $S$-SD): the motivation is to reduce the computational complexity, at the same time determining a locally optimal solution.

Some preliminary results of a prototypical system, Assembly Disassembly in Three Dimensions (A3D), implementing the SD abstractions, and analysis of the abstractions for different classes of SD problems are presented.

INTRODUCTION

Selective Disassembly (SD) involves disassembling a subset of components ($C$) from an assembly ($A$) to obtain a SD sequence ($S$). For example, to disassemble $C = \{C_4, C_5\}$ from $A$ in Figure 1a, $S = \{C_6, C_4, C_5\}$, as shown in Figure 1b.

\begin{figure}[h]
  \centering
  \includegraphics[width=0.5\textwidth]{Figure1a.png}
  \caption{Test Assembly to illustrate SD}
  \label{fig:1a}
\end{figure}

\begin{figure}[h]
  \centering
  \includegraphics[width=0.5\textwidth]{Figure1b.png}
  \caption{$S = \{C_6, C_4, C_5\}$ for $C = \{C_4, C_5\}$}
  \label{fig:1b}
\end{figure}

Applications such as maintenance, recycling, and reuse usually require removal of a subset of components of $A$, and not the entire assembly, hence providing a need for SD analysis [11]. For example, aircraft engine maintenance requires the SD of the engine and not the disassembly of the entire aircraft. Another application of SD is in reuse application requiring removal of some high-valued components such as SD of an instrument panel (for its reuse) from a car assembly.

Selective Disassembly Planning

This research focuses on automatic determination of $S$ for $C$ from $A$. The SD problem is formulated as follows:

- **Given**: An assembly $A$ and a selected set of components $C$ to be disassembled.
- **Requirement**: Determine automatically $S$ for $C$ with minimal component removals.

The objective of minimal component removals is appropriate, since for 1-disassembleable components (a 1-disassembleable component requires a single linear motion to be removed from $A$ [15]) the objective becomes minimizing the disassembly motions (operations), which is a measure of difficulty of disassembling [6]. Moreover, the product design for manufacture suggests simple motions and easier separation of components [1, 3, 4] for maintenance/recycling. Also, the above objective is consistent with earlier research [15] in SD analysis. Therefore, $S$ with minimal component removals is defined as an Optimal Sequence ($OS$).
EFFICIENT SELECTIVE DISASSEMBLY

One potential approach to determining OS is an exhaustive enumeration of all the possible sequences and the selection of OS with minimal removals. However, this analysis is computationally expensive (typically exponential with respect to the number of components in A), and is not recommended.

Several researchers (e.g., [7, 8, 13, 14]) have proposed automated Complete Disassembly CD algorithms, which involves disassembling all the components in A. An application for CD is assembling, since reversing a CD sequence can potentially yield an assembly sequence (e.g. [6, 13, 15]). An extensive amount of research in CD sequencing and assembly planning exists. Although S can be obtained from a CD, it may not give an optimal solution [10].

Another approach is the construction of a Disassembly Tree [15] for CD and a single component disassembly. The tree is designed to model the ‘Onion Peeling’ algorithm—recursively disassembling removable components, starting from the boundary of A and proceeding inwards. The Disassembly Tree approach [15] is proposed for 2.5 dimension objects and the analysis is based on the contact geometry. However, the algorithm is only applicable for assemblies in which every component is disassembled by removing none or one of its mating adjacent components. Therefore, the above approach is restrictive for our use.

In the SD problem, the requirement is to identify S to disassemble C. However, apart from the objective that the SD analysis should be automatic and analyze 3D geometric models, there are two other important issues: (i) Computationally efficient algorithms and (ii) Optimum SD solution. Efficiency and optimality are related, and one is usually achieved at the cost of the other. For example, if efficiency is the only issue, then any of the CD solution can be extended for SD. However, this results in a non-optimum solution. On the other hand, if optimality is the only issue, then exhaustive enumeration will give an optimum solution; however, this approach is computationally inefficient.

Therefore, the current research attempts to provide algorithms that balance the requirements of computationally efficient and optimal solution; i.e., determining a SD solution with fewer component removals that can be computed in a feasible computation time.

Based on the number of components to be disassembled, the SD problem of disassembling C from A of n components is categorized in to three classes:
1. SD of one component--defined as Single SD (1-SD),
2. SD of s < n components, defined as Multiple SD (s-SD), and
3. SD of S > n components, defined as Large SD (S-SD).

Each class of SD problem is studied, and analysis of some methods for automated SD is presented in subsequent sections. Prior to the analysis, some terms related to SD algorithms are defined.

DEFINITIONS

- Disassemblability: $\Delta_i$ is a binary value that indicates if $C_i \in A$ is removable [9]. For example, in Figure 2, $\Delta_4 = \text{TRUE}$ for $C_4$ and $\Delta_3 = \text{FALSE}$ for $C_3$.
- Removal Influence: $RI_i$ is a binary value that indicates if $C_i \in A$ is removable after the removal of $C_j \in A$ [9]. For example, in Figure 2, $RI_3^4 = \text{TRUE}$, since $\Delta_3$ is TRUE with removal of $C_4$ in $A$. Similarly, $RI_3^5 = \text{FALSE}$.
- 1-dependent component: A $C_i$ requiring disassembly of one component or none of its adjacents for infinitesimal translation in A is defined as 1-dependent component [9]. For example, in Figure 2, $C_3$ is 1-dependent; i.e., infinitesimal translation of $C_3$ is possible by disassembling $C_4$.

Assumptions of the Current Research

The assumptions for the current research are:
1. The relative motions of the components are determined without considering the tools, fixtures or robots required to achieve these motions.
2. Assemblies are assumed to be rigid, frictionless and defined by nominal geometry.
3. Components are 1-disassemblable (single linear motion to be removed from A) and 1-dependent.
4. Disassembly sequences are sequential, monotonic, and non-destructive (no component is destroyed).

Assumptions 1-4 are standard assumptions followed by different researchers (e.g., [7, 13, 15]) in automated assembly/disassembly analysis. The 1-disassemblable assumption is utilized by several researchers (see e.g., [6, 8, 15]), since automatic generation of disassembly sequences allowing general disassembly motion is computationally expensive. Moreover, the 1-disassemblable assumption is realistic for some real world examples, as indicated by some existing assembly planning systems (e.g., [8]).
SINGLE SELECTIVE DISASSEMBLY

The motivation for 1-SD problem is that the disassembly analysis can be localized with respect to the target component \( C_t \), and analysis of all components in \( A \) for disassembly may not be required.

In Srinivasan and Gadgil [9], an algorithm called Single Wave Propagation (SWP) has been introduced for single component disassembly. The SWP algorithm defines a wave to topologically arrange components in \( A \) to denote the disassembly order such that a component in one wavefront, \( \tau_{n+1} \) (\( n \)th wavefront), can be disassembled by removing its adjacent component(s) in \( \tau_n \) (\( n \)th wavefront).

A disassembly wave is represented by a Removal influence Graph (RG) [9], whose nodes correspond to components in the disassembly wave and arcs correspond to the removal influence between the components. An edge \( C_i \rightarrow C_j \) indicating that \( C_i \) is disassemblable after removing \( C_j \) represents a WP from \( C_i \) to \( C_j \). For example, Figure 4 illustrates a WP from \( \tau_n \) to \( \tau_{n+1} \) where \( C_i \in \tau_n, C_j \in \tau_{n+1}, A_i = \text{FALSE} \) and \( R_{ij} = \text{TRUE} \). The WP from \( C_i \) to \( C_j \) implies that \( C_i \) is disassemblable after removing \( C_j \) (similarly for \( C_j \rightarrow C_i \)).

SWP algorithm defines \( C_s \) as a wave source, with the waves propagating outwards until a removable component (defined as the boundary component \( C_b \)) is reached. The wavefronts provide a hierarchy for component removal. \( OS = \{ C_s \rightarrow C_1 \} \) is derivable from RG: where \( C_s \rightarrow C_1 \) denotes the shortest path from \( C_s \) to \( C_1 \) in the graph RG. To illustrate this, consider \( A \) in Figure 3 with \( C = \{ C_1 \} = \{ C_4 \} \), Figure 5 shows the WP from \( C_2 \). \( C_3 \) is disassemblable by removing \( C_2 \) or \( C_4 \) in \( \tau_1 \) \( (RI_{12} = \text{TRUE}, RI_{14} = \text{TRUE}) \). \( C_4 \) is disassemblable after removing \( C_5 \) in \( \tau_2 \) \( (RI_{45} = \text{TRUE}) \) and \( C_2 \) is disassemblable after removing \( C_1 \) in \( \tau_2 \) \( (RI_{12} = \text{TRUE}) \). Since \( A_1 = \text{TRUE} \) for \( C_1 \) (a boundary component), \( OS = \{ C_5, C_2, C_3 \} \) for \( C_3 \).

MULTIPLE SELECTIVE DISASSEMBLY

One potential algorithm to perform s-SD analysis is by applying SWP for every component in \( C \). Although this approach may determine \( OS \) for individual target components, the resultant \( S \) (aggregation of all disjoint \( OS \)) is not necessarily \( OS \). To illustrate the s-SD problem, consider \( A \) in Figure 3 with the requirement to disassemble \( C = \{ C_5, C_19 \} \). Let \( n_B \) = number of components in \( S \). Applying 1-SD algorithm results in \( S = \{ C_1, C_2, C_3, C_4, C_5, C_23, C_22, C_21, C_20, C_19 \} \) with \( n_B = 10 \). But better sequences \{ \{ C_5, C_1, C_2, C_3, C_4, C_5, C_6, C_19 \} \} and \{ \{ C_23, C_22, C_21, C_20, C_19, C_6, C_5 \} \} exist with \( n_B = 7 \). Therefore, a better solution may be obtained if two or more components are disassembled along a common sequence. This motivates the need for s-SD analysis.

In Srinivasan and Gadgil [10], an algorithm called Multiple Wave Propagation (MWP) has been introduced for multiple components disassembly. The MWP algorithm defines two types of disassembly waves: \( \tau \) and \( \beta \) waves that determine the disassembly ordering with
respect to \( C \) and the \( \beta \) boundary of \( A \), respectively. For example, Figure 6a illustrates a \( \tau \) wave from \( \tau_{a} \) to \( \tau_{b} \), where \( C_{1} \in \tau_{a} \), \( C_{2} \in \tau_{b} \). \( \Delta_{a} = \text{FALSE} \) and \( \text{RI}_{a} = \text{TRUE} \). The \( \tau \) wave from \( C_{1} \) to \( C_{2} \) implies that \( C_{1} \) is disassemblable after removing \( C_{2} \). Similarly, the \( \beta \) wave from \( C_{2} \) in \( \beta_{a} \) to \( C_{1} \) in \( \beta_{a} \) is represented as \( C_{1} \rightarrow C_{2} \) denoting that \( C_{1} \) is disassemblable after disassembling \( C_{2} \). Figure 6b shows a \( \beta \) wave from \( C_{1} \) in \( \beta_{a} \) to \( C_{2} \) in \( \beta_{a} \) implying the minimum number of components to be removed to disassemble \( C_{j} \) from \( A \) is (a-1) and \( C_{1} \) is a.

\[
\text{Figure 6. (a) \( \tau \) Wave (b) \( \beta \) Wave}
\]

Based on the intersection event (IE) between \( \tau \) and \( \beta \) waves, a sequence to disassemble \( C \) is determined. The importance of the intersection waves lies in the determination of the component at which the waves intersect (therefore the shape of the wave in the geometry space is irrelevant).

Every occurrence of an IE for \( m > 0 \) \( \tau \) wave(s) \((\tau^{1}, \tau^{2}, \ldots, \tau^{m})\) determines \( S \) for \( C = (C_{11}, C_{12}, \ldots, C_{1m}) \), where \( C_{1} \subseteq C \). For example, Figure 7 shows the RG for \( C = (C_{5}, C_{19}) \) for the Gear Reducer assembly in Figure 3. An IE occurs at \( C_{a} \): \( \tau_{1} \) of \( C_{5} \cap \beta_{a} \), \( C_{19} \) at \( C_{6} \) which determines \( S_{1} = (C_{1}, C_{2}, C_{3}, C_{4}, C_{5}, C_{6}, C_{19}) \), \( S_{2} = (C_{23}, C_{22}, C_{21}, C_{20}, C_{19}, C_{6}, C_{3}) \). Similarly, all other IE’s and the corresponding sequences are determined to evaluate an OS for \( C \).

\[
\text{Figure 7. RG for } C = (C_{5}, C_{19}) \quad \begin{align*} S_{1} &= (C_{1}, C_{2}, C_{3}, C_{4}, C_{5}, C_{6}, C_{19}), \\ S_{2} &= (C_{23}, C_{22}, C_{21}, C_{20}, C_{19}, C_{6}, C_{3}) \end{align*}
\]

LARGE SELECTIVE DISASSEMBLY

A MWP algorithm is computationally feasible for \( s \) (<<n). However, for \( S \) (= n) components, it is computationally prohibitive, which provides motivation for a new algorithm for S-SD.

In Srinivasan and Gadh [12], an algorithm called Priority Intersection Event (PIE) algorithm has been introduced for multiple components disassembly. The PIE algorithm defines prioritized intersection events between \( \tau \) and \( \beta \) waves to determine locally minimum component removal sequences. The \( \tau \) and \( \beta \) wave propagation’s are as illustrated in Figure 6, however, for every time step \( T \) the \( \tau \) wave advances by one wavefront and \( \beta \) wave by \( T \) wavefronts. For S-SD, let \( \tau^{n} = \text{wave of } C_{n} \), \( \tau^{m} = \text{wave of } C_{m} \); \( \beta_{n} = \text{wavefront of } C_{m} \); \( \beta^{m} = \beta_{1} \cap \beta_{2} \cap \cdots \cap \beta_{m} \cap \tau_{n} \), where \( 1 \leq m \leq S \). For example, a priority event is determined: Let \( m = \text{number of } \tau \text{ waves intersect } C_{m} \in A \). An IE between the \( m \) \( \tau \) waves and \( \beta^{m} \) at \( C_{n} \) with the number of components in the sequence being less than the corresponding disjoint sequence is a priority event. Figure 8 shows the RG for the Gear Reducer assembly in Figure 3 with \( C = (C_{2}, C_{3}, C_{4}) \) at \( T = 2 \). At \( C_{2} \), there are five IE’s: \( \tau_{2} \cap \tau_{1} \cap \beta_{2} \cap \beta_{1}, \tau_{1} \cap \beta_{2} \cap \beta_{1}, \tau_{1} \cap \beta_{1}, \tau_{1} \cap \beta_{2}, \tau_{1} \cap \beta_{1} \). However, a priority event at \( C_{2} \) is \( \tau_{1} \cap \tau_{2} \cap \beta_{2} \cap \beta_{1} \) determines a common sequence with locally minimum component removals.

\[
\text{Figure 8. RG at } T = 2 \text{ for } C = (C_{2}, C_{3}, C_{4}): \text{IE at } C_{2}, \quad S_{2} = (C_{1}, C_{2}, C_{3}, C_{4})
\]

Different classes of priority events, their characteristics, and evaluation of an OS based on the order of priority events are defined [12]. For the Gear Reducer assembly in Figure 3 with \( C = (C_{2}, C_{3}, C_{4}, C_{12}, C_{13}, C_{20}, C_{21}, C_{22}) \) an OS from PIE algorithm is \( (C_{1}, C_{2}, C_{3}, C_{4}, C_{23}, C_{22}, C_{21}, C_{20}, C_{14}, C_{13}, C_{12}) \).

DISCUSSION

This section presents some of the attributes and preliminary implementation results of SD algorithms, and discusses the applicability of these algorithms for SD planning.

Attributes of WP algorithms in SD Planning

Some of the attributes of the SWP algorithm are:

- The wave denotes the disassembly order and minimal waves denote a minimal component removal sequence.
- The average computational complexity of determining an OS is \( O(n^2) \) for a 1-dependent solution [9].

132
The number of components analyzed ranges from 1 to n, depending on the geometry of \( C_j \) in \( A \). If \( C_j \) is closer to the boundary, then the number of components analyzed is significantly less than n.

Some of the primary attributes of MWP algorithm are:

- An IE for \( m=1 \) determines a disjoint sequence for \( C_j \in C \). However, for \( m>1 \), IE determines a common sequence to disassemble all the \( m \) components \( C_{j1}, C_{j2}, ..., C_{jm} \in C \).
- IE occurrence for \( C_j \) depends on the geometric configuration of \( A \), i.e., if in \( A \), there is no IE of \( m \) \((>0)\) wave(s) \((\tau^1, \tau^2, ..., \tau^m)\) with a \( \beta \) wave at \( C_{w} \in A \), then there exist no common sequence \( S \) for \( \{C_{j1}, C_{j2}, ..., C_{jm}\} \) of type \((C_b \sim C_{w}, C_{w} \sim C_{j1}, ..., C_{w} \sim C_{jm})\).
- The computational complexity of the MWP approach is of order \( O(n^22^2) \) [10]. For \( s << n \), this algorithm is computationally feasible, and is efficient compared to the enumeration approach of computational complexity \( O(2^n) \).

Some of the primary attributes of PIE algorithm are:

- The priority events are necessary candidate events in determining an \( OS \) and number of such events is polynomial in number [12].
- The order of event occurrence in \( A \) depends on the geometric configuration of components in \( A \) and every IE for \( m \) \((>0)\) wave(s) \((\tau^1, \tau^2, ..., \tau^m)\), with a \( \beta^m \) wave, determines \( S \) for \( C = \{C_{j1}, C_{j2}, ..., C_{jm}\} \) with locally minimum component removals; where \( C_j \subseteq C \).
- The computational complexity of PIE algorithm in determining an \( OS \) is \( O(sn^2) \) [12].

The SD algorithms presented for 1-SD, s-SD and S-SD problems have some attributes in common:

- A wave provides topological disassembly ordering based on the geometric attributes, such as accessibility of components.
- A wave embeds the disassemblability of components in previous wavefronts; i.e., once a component in one wavefront is disassembled, its adjacent components in the previous wavefront become disassemblable.
- A SD solution with minimal wavefronts gives a locally optimum solution.

**Preliminary Implementation Results**

Some preliminary implementation results of SD algorithms in a prototypical system A3D (Assembly Disassembly in Three Dimensions) are presented below.

Figures 9a and 9c show the result for single component disassembly. For the Engine sub-assembly shown in Figure 9a with \( C = \{C_3\} \) there are two optimal solutions: Figure 9b shows the SD with \( OS = \{C_1, C_2, C_3\} \) and Figure 9c shows the SD with \( OS = \{C_6, C_5, C_4\} \).

Figure 10b shows the result of disjoint SD sequences obtained from SWP algorithm. For the Cab sub-assembly shown in Figure 10a with \( C = \{C_2, C_3\} \); \( S = \{C_1, C_2, C_4\} \). The resultant \( S \) is obtained from two disjoint sequences: \( OS = \{C_1, C_2\} \) for \( C1 = \{C_2\} \) and \( OS = \{C_4, C_5\} \) for \( C2 = \{C_5\} \).
SWP, MWP and PIE algorithms are taken as references for comparison with respect to other algorithms.

REFERENCES