N90-29866

PRECEDENCE RELATIONSHIP REPRESENTATIONS OF MECHANICAL ASSEMBLY SEQUENCES

L. S. Homem de Mello
The Robotics Institute
Carnegie Mellon University
Pittsburgh, Pennsylvania 15213-3890

A. C. Sanderson
Electrical, Computer, and Systems Engineering Dept.
Rensselaer Polytechnic Institute
Troy, New York 12180-3590

Abstract

This paper presents two types of precedence relationship representations for mechanical assembly sequences: precedence relationships between the establishment of one connection between two parts and the establishment of another connection, and precedence relationships between the establishment of one connection and states of the assembly process. Precedence relationship representations have the advantage of being very compact. The problem with these representations was how to guarantee their correctness and completeness. Two theorems are presented each of which leads to the generation of one type of precedence relationship representation guaranteeing its correctness and completeness for a class of assemblies.

1. Introduction

The generation of assembly sequences is an important capability for both autonomous and telerobotic systems for space applications. Assembly, repair, servicing, and sample acquisition are examples of tasks that are envisioned for space robotic systems. In each case, a plan or sequence of operations must be generated, usually off-line, based on prior knowledge. In real-time, it may be necessary to modify the plan based on monitoring and sensing of the execution. The desirable representation of the alternative plans for an off-line planning system may be quite different from the desirable representation for the real-time control system. The understanding of alternative representations of such plans is fundamental to their integration into a useful system.

Several methodologies for representing assembly sequences have been utilized. These include representations based on directed graphs [3], on AND/OR graphs [8], on establishment conditions [2], and on precedence relationships [3, 6]. Those based on directed graphs and on AND/OR graphs are explicit representations since there is a mapping from the assembly tasks into the elements of the representations. Those based on establishment conditions and on precedence relationships are implicit representations because they consist of conditions that must be satisfied by the assembly sequences.

In previous work [9] we have described a correct and complete algorithm for the generation of mechanical assembly sequences. This algorithm yields the AND/OR graph representation of assembly sequences. The correspondence between the AND/OR graph and the directed graph representations has also been established [10].

In this paper we address the precedence relationship representations of assembly sequences. These representations have the advantage that they are very compact and therefore might be preferred in real-time planning of assembly sequences. The problem with precedence relationship representations was the assessment of their correctness and completeness. By correctness of the representation we mean that only feasible sequences satisfy the precedence relationships. By completeness we mean that all the feasible sequences satisfy the precedence relationships.

Two types of precedence relationship representations can be used to represent mechanical assembly sequences: precedence relationships between the establishment of one connection between two parts and the establishment of another connection, and precedence relationships between the establishment of one connection and states of the assembly process. This paper describes these two representations and shows two theorems that can be used to guarantee their correctness and completeness for a class of assemblies.

2. Background

A mechanical assembly is a composition of parts interconnected forming a stable unit. Each part is a solid object. Parts are interconnected whenever they have one or more surfaces in contact. Surface contacts between parts reduce the degrees of freedom for relative motion. A cylindrical contact, for example, prevents any relative motion that is not a translation along the axis or a rotation around the axis. Attachments may act on surface contacts and eliminate all degrees of freedom for relative motion. For example, if a cylindrical contact has a pressure-fit attachment, then no relative motion between the parts is possible.

A subassembly is a nonempty subset of parts that either has only one element (i.e. only one part), or is such that every part has at least one surface contact with another part in the subset. Although there are cases in which it is possible to join the same pair of parts in more than one way, a unique assembly geometry will be assumed for each pair of parts. This geometry corresponds to their relative location in the whole assembly. A subassembly is said to be stable if its parts maintain their relative position and do not break contact spontaneously. All one-part subassemblies are stable.

The assembly process consists of a succession of tasks, each of which consists of joining subassemblies to form a larger subassembly. The process starts with all parts separated and ends with all parts properly joined to form the whole assembly. For the current analysis, it is assumed that exactly two subassemblies are joined at each assembly task, and that after parts have been put together, they remain together until the end of the assembly process.

It is also assumed that whenever two parts are joined all contacts between them are established. Due to this assumption, an assembly can be represented by a simple undirected graph $\langle P, C \rangle$ in which $P = \{ p_1, p_2, \cdots, p_N \}$ is the set of nodes, and $C = \{ c_1, c_2, \cdots, c_L \}$ is the set of edges. Each node in P corresponds to a part in the assembly, and there is one edge in C connecting every pair of nodes whose corresponding parts have at least one surface contact. The elements of C are referred to as *connections*, and the graph $\langle P, C \rangle$ is referred to as the *assembly's graph of connections*. A connection encompasses all contacts between two parts. Figure 1 shows an assembly in exploded view, and figure 2 shows its corresponding graph of connections.

• Assembly states

The state of the assembly process is the configuration of the parts at the beginning (or at the end) of an assembly task. The configuration of parts is given by the contacts that have been established. Since whenever two parts are joined all contacts between them are established, the configuration of parts is given by the connections that have been established. Therefore, a state of the assembly process can be represented by an L-dimensional binary vector $\underline{x} = [x_1, x_2, \cdots, x_L]$ in which the i^{th} component x_i is true or false respectively if the i^{th} connection is established in that state or not.

As mentioned above, it is assumed that whenever a subassembly is formed all connections between its parts are established. Therefore, any subassembly can be characterized by its set of parts. In the rest of this paper, references to subsets of parts should be understood as references to the subassemblies made up of those parts. It will always be clear from context what the whole assembly is. Because of this assumption, any state of the assembly process can also be represented by a partition of the set of parts of the whole assembly.

Given an assembly's graph of connections and one of the two representations of assembly states described above (binary vector or partition), it is straight forward to obtain the other representation.

There are partitions of the set of parts of the whole assembly that cannot characterize a state of the assembly process. For example, the partition { {CAP, HANDLE}, {RECEPTACLE}, {STICK} } cannot characterize a state of the assembly process for the assembly shown in figure 1 because the subset {CAP, HANDLE} does not characterize a subassembly. Partitions that can characterize a state of the assembly process will be referred to as *state partitions*, and partitions that cannot characterize a state will be referred to as *nonstate partitions*.

Similarly, not all L-dimensional binary vectors can characterize a state. For example, for the assembly shown in figure 1 the 5-dimensional binary vector [true, true, false, false, false] does not correspond to a state because if connections c_1 and c_2 are established then connection c_3 should also be established. L-dimensional binary vectors that can characterize a state will be referred to as *state vectors* whereas L-dimensional binary vectors that cannot characterize a state will be referred to as *nonstate vectors*.

Any state of the assembly process can be associated to a simple undirected graph $\langle P, C_k \rangle$ in which P is the set of nodes of the assembly's graph of connections, and C_k is the subset of connections ($C_k \subseteq C$) that is established in that state. This graph is referred to as the *state's graph of connections*. Except for the final state of the assembly process, a state's graph of connections has more than one component.

We will use the subassembly predicate *sa* to determine whether a subset of parts makes up a subassembly. The argument to this predicate is a subset of parts, and its value is either true or false depending on whether that subset of parts corresponds to a subassembly. For example, for the assembly shown in figure 1, *sa*({ RECEPTACLE, HANDLE }) = true, whereas *sa*({ CAP, HANDLE }) = false. From the assembly's graph of connections it is straight forward to compute *sa* for any given subset of parts.

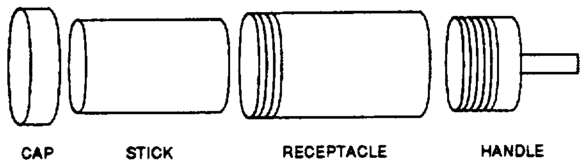


Figure 1: A simple product in exploded view.

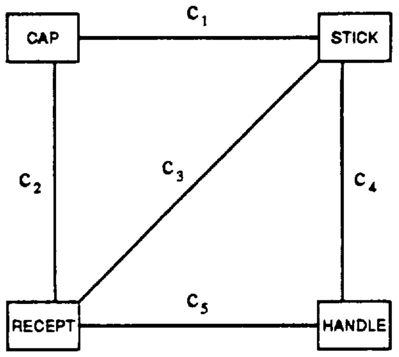


Figure 2: The graph of connections for the product shown in Figure 1

In this paper, a partition of the set of parts whose elements all satisfy the subassembly predicate is an assembly state representation, regardless of whether that state actually occurs in any of the different ways the assembly can be assembled. The corresponding L-dimensional binary vector is also an assembly state representation. And the corresponding configuration of parts is an assembly state. For example, for the assembly shown in figure 1, the partition { {CAP, RECEPTACLE, HANDLE}, {STICK} } as well as the corresponding L-dimensional binary vector [false, true, false, false, true] are assembly state representations. Yet, since it was assumed that once parts are put together they remain together, the configuration of parts (i.e. the state) corresponding to these representations cannot occur in any assembly process. Once the cap and the handle have been joined to the receptacle, it is no longer possible to join the stick.

In this paper, an assembly state representation for which all subassemblies satisfy the stability predicate is said to be a *stable assembly state representation.* For example, for the assembly shown in figure 1, the partition { {CAP, RECEPTACLE, HANDLE}, {STICK} } as well as the corresponding binary vector [false, true, false, false, true] are stable assembly state representations.

• **Assembly tasks**

Given two subassemblies characterized by their sets of parts θ_i and θ_j, we say that joining θ_i and θ_j is an assembly task if the set $\theta_k = \theta_i \cup \theta_j$ characterizes a subassembly. For example, for the assembly shown in figure 1, if θ_i = { RECEPTACLE } and θ_j = { HANDLE } then joining θ_i and θ_j is an assembly task, whereas if θ_i = { CAP } and θ_j = { HANDLE } then joining θ_i and θ_j is not an assembly task. The subassemblies θ_i and θ_j are the *input* subassemblies of the assembly task, and θ_k is the *output* subassembly of the assembly task. Due to the assumption of unique geometry, an assembly task can be characterized by its input subassemblies only and it can be represented by a set of two subsets of parts. For example, for the assembly shown in figure 1, the joining of the cap to the receptacle is represented by { {CAP}, {RECEPTACLE} }.

An assembly task is said to be *geometrically* feasible if there is a collision-free path to bring the two subassemblies into contact from a situation in which they are far apart. And an assembly task is said to be *mechanically* feasible if it is feasible to establish the attachments that act on the contacts between the two subassemblies.

• **Assembly sequences**

Given an assembly that has N parts, an ordered set of $N-1$ assembly tasks $\tau_1, \tau_2, \cdots, \tau_{N-1}$ is an assembly sequence if there are no two tasks that have a common input subassembly, the output subassembly of the last task is the whole assembly, and the input subassemblies to any task τ_i is either a one-part subassembly or the output subassembly of a task that precedes τ_i. To any assembly sequence $\tau_1, \tau_2, \cdots, \tau_{N-1}$ there corresponds an ordered sequence $s_1, s_2, \cdots, s_N$ of N assembly states of the assembly process. The state s_1 is the state in which all parts are separated. The state s_N is the state in which all parts are joined forming the whole assembly. And any two consecutive states s_i and s_{i+1} are such that only the two input subassemblies of task τ_i are in s_i and not in s_{i+1}, and only the output subassembly of task τ_i is in s_{i+1} and not in s_i. Therefore, an assembly sequence can also be characterized by an ordered sequence of states.

An example of an assembly sequence for the assembly shown in figure 1 is:

1. The first task (τ_1) consists of joining the cap to the receptacle.
2. The second task (τ_2) consists of joining the stick to the subassembly made up of the cap and the receptacle.
3. The third task (τ_3) consists of joining the handle to the subassembly made up of the cap, the stick, and the receptacle.

An assembly sequence is said to be feasible if all its assembly tasks are geometrically and mechanically feasible, and the input subassemblies of all tasks are stable. The assembly sequence described above is feasible. An example of an unfeasible assembly sequence for the assembly shown in figure 1 is:

1. The first task (τ_1) consists of joining the cap to the receptacle.
2. The second task (τ_2) consists of joining the handle to the subassembly made up of the cap and the receptacle.
3. The third task (τ_3) consists of joining the stick to the subassembly made up of the cap, the stick, and the receptacle.

This assembly sequence is unfeasible because the third task (τ_3) is not geometrically feasible since there is no collision free path to bring the stick into the receptacle, once both the cap and the handle have been joined to the receptacle.

An assembly sequence (not necessarily feasible) can be represented in different ways. We will use the following representations:

- An ordered list of task representations. The number of elements in this list is equal to the number of parts minus one.
- An ordered list of binary vectors. Each vector must correspond to a state (not necessarily stable). The number of elements in this list is the equal to the number of parts.
- An ordered list of partitions of the set of parts. Each partition must correspond to a state (not necessarily stable). The number of elements in this list is equal to the number of parts.
- An ordered list of subsets of connections. The number of elements in this list is equal to the number of parts minus one.

Given the assembly's graph of connections and an assembly sequence in any of these four representations, it is straight forward to obtain the other three representations.

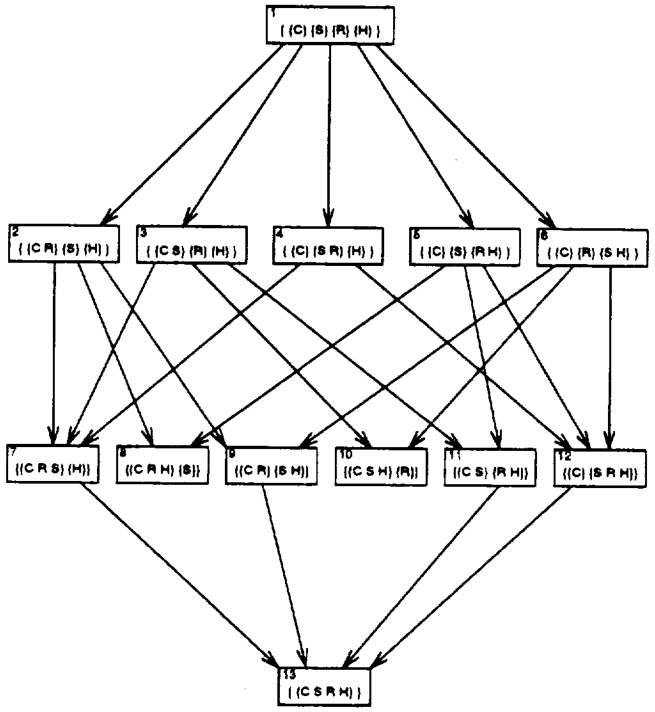
Since each assembly sequence can be represented by ordered lists, it is possible to represent the set of all assembly sequences by a set of lists, each corresponding to a different assembly sequence. It is also possible to use directed graphs, and AND/OR graphs to represent the set of all assembly sequences. Figure 3 shows the direct graph of feasible assembly sequences for the assembly shown in figure 1. The AND/OR graph of assembly sequences for the assembly shown in figure 1 has been presented elsewhere [8]. Alternatively, the set of all feasible assembly sequences can be represented by sets of precedence relationships. Sections 3 and 4 below present two types of precedence relationship representations of feasible assembly sequences.

3. Precedence relationships between the establishment of one connection and the establishment of another connection

We will use the notation $c_i < c_j$ to indicate the fact that the establishment of connection c_i must precede the establishment of connection c_j. And we will use the notation $c_i \le c_j$ to indicate the fact that the establishment of connection c_i must precede or be simultaneous with the establishment of connection c_j. Furthermore, we will use a compact notation for logical combinations of precedence relationships; for example, we will write[1] $c_i < c_j \cdot c_k$ when we mean $(c_i < c_j) \wedge (c_i < c_k)$, and we will write $c_i + c_j < c_k$ when we mean $(c_i < c_k) \vee (c_j < c_k)$.

An assembly sequence whose representation as an ordered sequence of binary vectors is $(\underline{x}_1, \underline{x}_2, \cdots, \underline{x}_N)$ and whose representation as an ordered sequence of subsets of connections is $(\gamma_1, \gamma_2, \cdots, \gamma_{N-1})$ satisfies the precedence relationship $c_i < c_j$ if $c_i \in \gamma_a$, $c_j \in \gamma_b$, and $a < b$. Similarly, the sequence satisfies $c_i \le c_j$ if $c_i \in \gamma_a$,

[1]The logical operation AND will be denoted either by the symbol "$\wedge$" or by the product of the two logical variables. Similarly, the logical operation OR will be denoted either by the symbol "$\vee$" or by the sum of the two logical variables.



C = cap S = stick R = receptacle H = handle

Figure 3: Directed graph of feasible assembly sequences for the assembly shown in figure 1.

$c_j \in \gamma_b$, and $a \leq b$. For example, for the assembly shown in figure 1, the assembly sequence whose representation as an ordered sequence of binary vectors is

([false, false, false, false, false]
[true, false, false, false, false]
[true, true, true, false, false]
[true, true, true, true, true])

and whose representation as an ordered sequence of subsets of connections is ($\{c_1\}$ $\{c_2, c_3\}$ $\{c_4, c_5\}$) satisfies the precedence relationships $c_2 < c_4$ and $c_2 \leq c_3$ but does not satisfy the precedence relationships $c_2 < c_3$ and $c_2 \leq c_1$.

Each feasible assembly sequence of a given assembly can be uniquely characterized by a logical expression consisting of the conjunction of precedence relationships between the establishment of one connection and the establishment of another connection. For example, for the assembly shown in figure 1, the assembly sequence described in the previous paragraph can be uniquely characterized by the following conjunction of precedence relationships

$$(c_1 < c_2) \wedge (c_2 < c_4) \wedge (c_2 \leq c_3) \wedge (c_3 \leq c_2) \wedge (c_4 \leq c_5) \wedge (c_5 \leq c_4)$$

The set of all M feasible assembly sequences can be uniquely characterized by a disjunction of M conjunctions of precedence relationships in which each conjunction characterizes one assembly sequence. Clearly, this logical combination of precedence relationships constitutes a correct and complete representation for the set of all assembly sequences.

It is often possible to simplify this logical combination of precedence relationships using the rules of boolean algebra. Further simplification is possible if one notices that there are logical combinations of precedence relationships that cannot be satisfied by any assembly sequence. For the assembly shown in figure 1, for example, the combination $(c_1 < c_2) \wedge (c_2 < c_3) \wedge (c_3 < c_4) \wedge (c_4 < c_5)$ cannot be satisfied by any assembly sequence. These combinations can be set as don't care conditions in the simplification of the logical combination of precedence relationships.

Whenever the assembly has the two properties described below, it is possible to obtain a simple precedence relationship representation of all assembly sequences. This representation is obtained using the result of theorem 1.

The first property is:

Property 1: Given any two states s_i and s_j, not necessarily in the same assembly sequence, let γ_i and γ_j be the sets of connections that are established in assembly tasks τ_i and τ_j from s_i and s_j respectively. If

$\langle P, C_i \rangle$ is the state's graph of connections associated to s_i,
$\langle P, C_j \rangle$ is the state's graph of connections associated to s_j,
$\gamma_i \subseteq \gamma_j$,
$C_i \subseteq C_j$, and
τ_j is geometrically and mechanically feasible,

then

τ_i is geometrically and mechanically feasible.

This property corresponds to the fact that if it is geometrically and mechanically feasible to establish a set of connections (γ_j) when many other connections (C_j) have already been established, then it is also geometrically and mechanically feasible to establish fewer connections ($\gamma_i \subseteq \gamma_j$) when fewer other connections ($C_i \subseteq C_j$) have been established. Although many common assemblies have this property, there are assemblies that don't have it.

The second property is:

Property 2: If the subsets $\theta_1, \theta_2, \cdots, \theta_k$ of the set of parts P characterize stable subassemblies, then the set $\theta = \theta_1 \cup \theta_2 \cup \cdots \cup \theta_k$ also characterizes a stable subassembly.

Like in the case of property 1, many common assemblies have this second property. Yet, there are assemblies that don't have it.

Theorem 1: Given an assembly made up of N parts whose graph of connections is $\langle P, C \rangle$ (with $C = \{c_1, c_2, \cdots, c_L\}$), let

$$\{ (\gamma_{11}\ \gamma_{21} \cdots \gamma_{(N-1)1}),\ (\gamma_{12}\ \gamma_{22} \cdots \gamma_{(N-1)2}),\ \cdots,\ (\gamma_{1M}\ \gamma_{2M} \cdots \gamma_{(N-1)M}) \}$$

be a set of M ordered sequences of subsets of connections that represent feasible assembly sequences. If the assembly has properties 1 and 2, then any ordered sequence of $N-1$ subsets of connections that represents an assembly sequence corresponds to a feasible assembly sequence if it satisfies the set of $2L$ precedence relationships:

$$c_i \le \sum_{j=1}^{M} T_{ij} \quad i = 1, 2, \cdots, L \qquad \text{and} \qquad \sum_{j=1}^{M} H_{ij} \le c_i \quad i = 1, 2, \cdots, L$$

where

$$T_{ij} = \prod_{k=1}^{L} \lambda_{ik} \quad \text{with} \quad \lambda_{ik} = \begin{cases} c_k \text{ if } c_k \in \gamma_{lj} \text{ and } l \ge i \\ \text{true otherwise} \end{cases}$$

$$H_{ij} = \prod_{k=1}^{L} \lambda_{ik} \quad \text{with} \quad \lambda_{ik} = \begin{cases} c_k \text{ if } c_k \in \gamma_{lj} \text{ and } l \le i \\ \text{true otherwise.} \end{cases}$$

The sum and the product in this theorem are the logical operations OR and AND respectively. Each term T_{ij} (for $i = 1, 2, \cdots, L$, and for $j = 1, 2, \cdots, M$) is the product of the variables corresponding to the connections that are established at the same time or after the establishment of connection c_i in the jth sequence. Similarly, each term H_{ij} (for $i = 1, 2, \cdots, L$, and for $j = 1, 2, \cdots, M$) is the product of the variables corresponding to the connections that are established before the establishment of connection c_i in the jth sequence. Precedence relationships that have "true" on either side are always satisfied. The proof of this theorem is presented elsewhere [7].

An example will illustrate the use of theorem 1. The assembly shown in figure 1 has properties 1 and 2. For that assembly, the set of feasible assembly sequences can be obtained from the directed graph shown in figure 3. There are ten feasible assembly sequences and they are:

$(\{c_1\}\{c_2,c_3\}\{c_4,c_5\})$ $(\{c_1\}\{c_5\}\{c_2,c_3,c_4\})$ $(\{c_2\}\{c_1,c_3\}\{c_4,c_5\})$
$(\{c_2\}\{c_4\}\{c_1,c_3,c_5\})$ $(\{c_3\}\{c_1,c_2\}\{c_4,c_5\})$ $(\{c_3\}\{c_4,c_5\}\{c_1,c_2\})$
$(\{c_4\}\{c_2\}\{c_1,c_3,c_5\})$ $(\{c_4\}\{c_3,c_5\}\{c_1,c_2\})$ $(\{c_5\}\{c_1\}\{c_2,c_3,c_4\})$
$(\{c_5\}\{c_3,c_4\}\{c_1,c_2\})$

Applying the result of theorem 1 to the above set of feasible sequences for the assembly shown in figure 1, the precedence relationships having connection c_1 alone on one side are:

$$c_1 \leq c_2 \cdot c_3 \cdot c_4 \cdot c_5 + c_2 \cdot c_3 \cdot c_4 \cdot c_5 + c_3 \cdot c_4 \cdot c_5 + c_3 \cdot c_5 + c_2 \cdot c_4 \cdot c_5 + c_2 + c_3 \cdot c_5 + c_2 + c_2 \cdot c_3 \cdot c_4 + c_2$$

and

$$\text{true} + \text{true} + c_2 \cdot c_3 + c_2 \cdot c_3 \cdot c_4 \cdot c_5 + c_2 \cdot c_3 + c_2 \cdot c_3 \cdot c_4 \cdot c_5 + c_2 \cdot c_3 \cdot c_4 \cdot c_5 + c_2 \cdot c_3 \cdot c_4 \cdot c_5 + c_5 + c_2 \cdot c_3 \cdot c_4 \cdot c_5 \leq c_1 .$$

Using the rules of boolean algebra, these two precedence relationships can be simplified yielding

$$c_1 \leq c_2 + c_3 \cdot c_5 \qquad \text{and} \qquad \text{true} \leq c_1 .$$

The second precedence relationship is always satisfied and can be ignored. Similarly, applying the result of theorem 1, simplifying the logical expressions, and deleting those precedence relationships that have "true" on either side, we obtain four additional precedence relationships. The resulting set of precedence relationships is:

$$c_1 \leq c_2 + c_3 \cdot c_5 \quad c_2 \leq c_1 + c_3 \cdot c_4 \quad c_3 \leq c_1 \cdot c_5 + c_2 \cdot c_4 \quad c_4 \leq c_5 + c_2 \cdot c_3 \quad c_5 \leq c_4 + c_1 \cdot c_3 . \qquad (Set\ 1)$$

Set 1 of precedence relationships still contains some redundancies and can be shown to be equivalent to

$$c_3 \leq c_1 \cdot c_5 + c_2 \cdot c_4 . \qquad (Set\ 2)$$

It should be noticed that an unfeasible assembly sequence, such as the assembly sequence whose representation as an ordered sequence of subsets of connections is $(\{c_2\}\{c_5\}\{c_1,c_3,c_4\})$, does not satisfy Set 2 of precedence relationships. It should also be noticed that there are ordered sequences of subsets of connections, such as $(\{c_3\}\{c_1,c_4\}\{c_2,c_5\})$, that do not represent an assembly sequence, but satisfy Set 2 of precedence relationships. The precedence relationships obtained using theorem 1 can only discriminate the feasible from the unfeasible assembly sequences. The information in the assembly's graph of connections allows the discrimination of assembly sequences from ordered sequences of subsets of connections that do not correspond to assembly sequences.

Theorem 1 is a sufficient condition. The set of precedence relationships obtained using this theorem is correct but not necessarily complete. But if the set of M ordered sequences of subsets of connections includes the representations of all feasible assembly sequences, then the resulting set of precedence relationships constitutes a correct and complete representation of the feasible assembly sequences.

4. Precedence relationships between the establishment of one connection and states of the assembly process

We will use the notation $c_i \rightarrow S(\underline{x})$ to indicate that the establishment of the i^{th} connection must precede any state s of the assembly process for which the value of the logical function $S(\underline{x})$ is true. The argument of $S(\underline{x})$ is the L-dimensional binary vector representation of the state s. We will use a compact notation for logical combinations of precedence relationships. For example, we will write $c_i + c_j \rightarrow S(\underline{x})$ when we mean $[c_i \rightarrow S(\underline{x})] \vee [c_j \rightarrow S(\underline{x})]$.

An assembly sequence whose representation as an ordered sequence of binary vectors is $(\underline{x}_1\, \underline{x}_2 \cdots \underline{x}_N)$ and whose representation as an ordered sequence of subsets of connections is $(\gamma_1\, \gamma_2 \cdots \gamma_{N-1})$ satisfies the precedence relationship $c_i \rightarrow S(\underline{x})$ if

$$S(\underline{x}_k) \Rightarrow \exists l\, [(l < k) \wedge (c_i \in \gamma_l)] \qquad \text{for } k = 1, 2, \cdots, N .$$

For example, for the assembly shown in figure 1, the assembly sequence whose representation as an ordered sequence of binary vectors is

([false, false, false, false, false]
[true, false, false, false, false]
[true, true, true, false, false]
[true, true, true, true, true])

and whose representation as an ordered sequence of subsets of connections is ($\{c_1\}$ $\{c_2, c_3\}$ $\{c_4, c_5\}$) satisfies the precedence relationship $c_1 \rightarrow x_2 \cdot x_3$ because the only states for which $S(\underline{x}) = x_2 \cdot x_3$ is true are the third and the fourth, and the establishment of connection c_1 occurs on the first assembly task. This sequence does not satisfy the precedence relationship $c_4 \rightarrow x_1 \cdot x_2 \cdot x_3$ because for the third state the value of $S(\underline{x}) = x_1 \cdot x_2 \cdot x_3$ is true but the establishment of connection c_4 occurs on the third assembly task, which occurs after the third state.

Let Ψ_S be the set of assembly states that never occur in any feasible assembly sequence. These include the unstable assembly states, the stable states from which the final state cannot be reached, and the states that cannot be reached from the initial state. Let $\Psi_X = \{\underline{x}_1, \underline{x}_2, \cdots, \underline{x}_J\}$ be the set of L-dimensional binary vectors that represent the states in Ψ_S. Every element $\underline{x}_j$ of Ψ_X is such that the value of the logical function $G(\underline{x}_j)$ is true, where

$$G(\underline{x}) = G(x_1, x_2, \cdots, x_L) = \sum_{k=1}^{K} \prod_{l=1}^{L} \lambda_{kl}. \qquad (Eq. 1)$$

The sum and the product in equation 1 are the logical operations OR and AND respectively, and λ_{kl} is either the symbol x_l if the l^{th} component of $\underline{x}_k$ is true, or the symbol $\bar{x}_l$ if the l^{th} component of $\underline{x}_k$ is false. In many cases the expression of $G(\underline{x})$ can be simplified using the rules of boolean algebra. Allowing for simplifications, but keeping the logical function as a sum of products[2], equation 1 can be rewritten as

$$G(\underline{x}) = \sum_{j=1}^{J'} g_j(\underline{x}) \qquad (Eq. 2)$$

where each term $g_j(\underline{x})$ is the product of a subset of $\{x_1, x_2, \cdots, x_L, \bar{x}_1, \bar{x}_2, \cdots, \bar{x}_L\}$ that does not include both x_i and $\bar{x}_i$ for any i. Each term $g_j(\underline{x})$ can be rewritten grouping all the nonnegated variables first and all the negated variables last, i.e., $g_j(\underline{x}) = x_a \cdot x_b \cdot \cdots \cdot x_h \cdot \bar{x}_p \cdot \bar{x}_q \cdot \cdots \cdot \bar{x}_z$.

Any assembly sequence that includes a state that is in Ψ_S is unfeasible. Therefore, a necessary condition for the feasibility of an assembly sequence whose representation as an ordered list of binary vectors is $(\underline{x}_1\ \underline{x}_2 \cdots \underline{x}_N)$ is that $G(\underline{x}_1) = G(\underline{x}_2) = \cdots = G(\underline{x}_N) =$ false. This is equivalent to $g_j(\underline{x}_i) =$ false for $i = 1, 2, \cdots, N$ and for $j = 1, 2, \cdots, J'$. If the assembly has property 1 (see section 3), this condition is also sufficient. Furthermore, if $(\underline{x}_1\ \underline{x}_2 \cdots \underline{x}_N)$ is an ordered list of binary vectors that represents an assembly sequence, the condition $g_j(\underline{x}_1) = g_j(\underline{x}_2) = \cdots = g_j(\underline{x}_N) =$ false corresponds to a precedence relationship. These facts are established by the following theorem. (The proof of this theorem is presented elsewhere [7].)

Theorem 2: Given an assembly made up of N parts whose graph of connections is $\langle P, C \rangle$ (with $C = \{c_1, c_2, \cdots, c_L\}$), let

$$G(\underline{x}) = \sum_{j=1}^{J'} g_j(\underline{x})$$

be a disjunctive normal form of the logical function that is true if and only if $\underline{x}$ is a binary-vector representation of a state that does not occur in any feasible assembly sequence. Let A_j be the set containing the indexes of the variables that are asserted in $g_j(\underline{x})$. Let N_j be the set containing the indexes of the variables that are negated in $g_j(\underline{x})$. If the assembly has property 1, and if $(\gamma_1\ \gamma_2 \cdots \gamma_{N-1})$ is an ordered sequence of subsets of connections that represents an assembly sequence, then $(\gamma_1\ \gamma_2 \cdots \gamma_{N-1})$ satisfies the set of precedence relationships

$$\sum_{k \in N_j} c_k \rightarrow \prod_{i \in A_j} x_i \qquad \text{for } j = 1, 2, \cdots, J'$$

if and only if it corresponds to a feasible assembly sequence.

[2]This form of a logical function is commonly referred to as *disjunctive normal form* [4].

An example will illustrate the use of theorem 2. For the assembly shown in figure 1, which has property 1, $\Psi_X = \{ [\text{false}, \text{true}, \text{false}, \text{false}, \text{true}], [\text{true}, \text{false}, \text{false}, \text{true}, \text{false}] \}$ (these binary vectors correspond to nodes 8 and 10 in the directed graph of assembly states shown in figure 3). Therefore,

$$G(\underline{x}) = G(x_1, x_2, x_3, x_4, x_5) = \bar{x}_1 \cdot x_2 \cdot \bar{x}_3 \cdot \bar{x}_4 \cdot x_5 + x_1 \cdot \bar{x}_2 \cdot \bar{x}_3 \cdot x_4 \cdot \bar{x}_5$$

In this case the expression of $G(\underline{x})$ cannot be further simplified and we have

$$g_1(\underline{x}) = \bar{x}_1 \cdot x_2 \cdot \bar{x}_3 \cdot \bar{x}_4 \cdot x_5 \qquad A_1 = \{2, 5\} \qquad N_1 = \{1, 3, 4\}$$
$$g_2(\underline{x}) = x_1 \cdot \bar{x}_2 \cdot \bar{x}_3 \cdot x_4 \cdot \bar{x}_5 \qquad A_2 = \{1, 4\} \qquad N_2 = \{2, 3, 5\}.$$

Therefore, the precedence relationships are:

$$c_1 + c_3 + c_4 \rightarrow x_2 \cdot x_5 \qquad c_2 + c_3 + c_5 \rightarrow x_1 \cdot x_4 \qquad (Set\ 3)$$

A simpler set of precedence relationships can be obtained if in the simplification of $G(\underline{x})$ we set the nonstate vectors as don't care conditions. For the assembly shown in figure 1, the set of precedence relationships

$$c_1 \Rightarrow x_2 \cdot x_5 \qquad c_2 \Rightarrow x_1 \cdot x_4 \qquad (Set\ 4)$$

was obtained in the same fashion as Set 3, except for setting the nonstate vectors as don't care conditions in the simplification of $G(\underline{x})$. Set 4 is simpler and yet equivalent to Set 3.

It should be noticed that an unfeasible assembly sequence, such as the assembly sequence whose representation as an ordered sequence of subsets of connections is $(\{c_2\}\ \{c_5\}\ \{c_1, c_3, c_4\})$, does not satisfy both sets of precedence relationships above (i.e. Sets 3 and 4). It should also be noticed that there are ordered sequences of $N-1$ subsets of connections and their corresponding ordered sequence of binary vectors, such as $(\{c_1\}\ \{c_2\}\ \{c_3, c_4, c_5\})$, and

([false, false, false, false, false]
[true, false, false, false, false]
[true, true, false, false, false]
[true, true, true, true, true]),

that do not represent an assembly sequence, but satisfy Sets 3 and 4 of precedence relationships. The precedence relationships obtained using the result of theorem 2 can only discriminate the feasible from the unfeasible assembly sequences. The information in the assembly's graph of connections allows the discrimination of assembly sequences from ordered sequences of subsets of connections that do not correspond to assembly sequences.

In order to be able to discriminate the representations of feasible assembly sequences from any sequence of $N-1$ subsets of connections, the set Ψ_X must also include all nonstate vectors. For the assembly shown in figure 1 there are 13 distinct assembly states, two of which do not occur in any feasible assembly sequence. Since there are 32 5-dimensional binary vectors, there are 19 5-dimensional nonstate vectors for the assembly shown in figure 1. Let $G(\underline{x})$ be the logical function that is true if and only if $\underline{x}$ is one of these 21 $(2+19)$ 5-dimensional vectors. Simplifying this function, we obtain

$$G(\underline{x}) = \bar{x}_1 \cdot x_2 \cdot x_5 + \bar{x}_1 \cdot x_2 \cdot x_3 + x_1 \cdot \bar{x}_2 \cdot x_4 + x_1 \cdot \bar{x}_2 \cdot x_3 + x_1 \cdot x_2 \cdot \bar{x}_3 + \bar{x}_3 \cdot x_4 \cdot x_5 + x_3 \cdot \bar{x}_4 \cdot x_5 + x_3 \cdot x_4 \cdot \bar{x}_5$$

Therefore, the precedence relationships are:

$$\begin{array}{llll} c_1 \rightarrow x_2 \cdot x_5 & c_1 \rightarrow x_2 \cdot x_3 & c_2 \rightarrow x_1 \cdot x_4 & c_2 \rightarrow x_1 \cdot x_3 \\ c_3 \rightarrow x_1 \cdot x_2 & c_3 \rightarrow x_4 \cdot x_5 & c_4 \rightarrow x_3 \cdot x_5 & c_5 \rightarrow x_3 \cdot x_4 \end{array} \qquad (Set\ 5)$$

The ordered sequences of subsets of connections $(\{c_1\}\ \{c_2\}\ \{c_3, c_4, c_5\})$, which does not correspond to an assembly sequence but satisfies Sets 3 and 4 of precedence relationships does not satisfy Set 5. But it should be noticed that Set 5 of precedence relationships will be "satisfied" for ordered sequences of subsets of connections containing less than $N-1$ subsets. For example, the sequence $(\{c_1, c_2, c_3\}\ \{c_4, c_5\})$ "satisfies" Set 5 of precedence relationships. Yet, this sequence does not correspond to a feasible assembly sequence because it does not contain exactly $N-1$ subsets of connections.

5. Conclusion

Two types of precedence relationships that can be used to represent assembly sequences were addressed: precedence relationships between the establishment of one connection and the establishment of another connection, and precedence relationships between the establishment of one connection and states of the assembly process.

The problem of guaranteeing the correctness and completeness of precedence relationship representations of assembly sequences was solved for the class of assemblies that have properties 1 and 2 described in section 3.

In previous work [9] we have described the generation of the correct and complete AND/OR graph representation of assembly sequences. The correspondence between the AND/OR graph and the directed graph representations has also been established [10]. The results presented in this paper can be used to generate correct and complete precedence relationship representations of assembly sequences from the AND/OR graph. These results can also be used in proving the correctness and completeness of algorithms for the generation of mechanical assembly sequences that yield precedence relationship representations.

Acknowledgements

We thank Thomas L. De Fazio for helpful discussions and James Daniell for making the software for simplification of boolean functions available to us. The responsibility for the paper, of course, remains with the authors.

This research was supported by the Conselho Nacional de Desenvolvimento Científico e Tecnológico (Brazil), the Jet Propulsion Laboratory of the California Institute of Technology, and The Robotics Institute of Carnegie Mellon University.

References

[1] N. Boneschanscher et al. Subassembly Stability. In *Proceedings of AAAI-88*, pages 780-785. Morgan Kaufman, August, 1988.

[2] A. Bourjault. *Contribution a une Approche Méthodologique de L'Assemblage Automatisé: Elaboration Automatique des Séquences Opératoires.* Thèse d'État, Université de Franche-Comté, Besançon, France, November, 1984.

[3] T. L. De Fazio and D. E. Whitney. Simplified Generation of All Mechanical Assembly Sequences. *IEEE Journal of Robotics and Automation* RA-3(6):640-658, December, 1987. See corrections on same journal, RA-4(6):705-708, December, 1988.

[4] H. D. Ebbinghaus et al. *Mathematical Logic*. Springer Verlag, 1984.

[5] B. R. Fox. *A Representation for Serial Robotic Tasks.* PhD thesis, Computer Science, University of Missouri-Rolla, 1987.

[6] B. R. Fox and K. G. Kempf. Opportunistic Scheduling for Robotics Assembly. In *1985 IEEE International Conference on Robotics and Automation*, pages 880-889. IEEE Computer Society, 1985.

[7] L. S. Homem de Mello. Forthcoming PhD Thesis. Department of Electrical and Computer Engineering, Carnegie Mellon University.

[8] L. S. Homem de Mello and A. C. Sanderson. AND/OR Graph Representation of Assembly Plans. In *Proceedings of AAAI-86*, pages 1113-1119. Morgan Kaufmann, 1986.

[9] L. S. Homem de Mello and A. C. Sanderson. *Automatic Generation of Mechanical Assembly Sequences.* Technical Report CMU-RI-TR-88-19, The Robotics Institute - Carnegie Mellon University, December, 1988.

[10] L. S. Homem de Mello and A. C. Sanderson. Task Sequence Planning for Assembly. In *IMACS World Congress '88 on Scientific Computation*. Paris, July, 1988.

[11] L. S. Homem de Mello and A. C. Sanderson. Planning Repair Sequences using the AND/OR Graph Representation of Assembly Plans. In *Proceedings of the 1988 IEEE International Conference on Robotics and Automation*, pages 1861-1862. Computer Society Press of the IEEE, April, 1988.

[12] M. M. Lui. *Generation and Evaluation of Mechanical Assembly Sequences Using the Liaison-Sequence Method.* Master's thesis, Department of Mechanical Engineering, Massachusetts Institute of Technology, May, 1988. Also published as Report CSDL-T-990, The Charles Stark Draper Laboratory, Inc.

[13] A. C. Sanderson, M. A. Peshkin, and L. S. Homem de Mello. Task Planning for Robotic Manipulation in Space Applications. *IEEE Transactions on Aerospace and Electronic Systems* 24(5), September, 1988.