Artificial Intelligence Approach to Planning the Robotic Assembly of Large Tetrahedral Truss Structures*

Luiz S. Homem-de-Mello
Jet Propulsion Laboratory
California Institute of Technology
Pasadena CA 91109-8099

Abstract

This paper presents an assembly sequence planner for tetrahedral truss structures. To overcome the difficulties due to the large number of parts, the planner exploits the simplicity and uniformity of the shapes of the parts and the regularity of their interconnection. The planning automation is based on the computational formalism known as *production system*. The global database consists of an hexagonal grid representation of the truss structure. This representation captures the regularity of tetrahedral truss structures and their multiple hierarchies. It maps into quadratic grids and can be implemented in a computer by using a two dimensional array data structure. By maintaining the multiple hierarchies explicitly in the model, the choice of a particular hierarchy is only made when needed, thus allowing a more informed decision. Furthermore, testing the preconditions of the production rules is simple because the patterned way in which the struts are interconnected is incorporated into the topology of the hexagonal grid. A directed graph representation of assembly sequences allows the use of both graph search and backtracking control strategies.

1 Introduction

Figure 1 shows a tetrahedral truss structure similar to those that will be used in future space missions [13]. The assembly, disassembly or repair of these large truss structures requires careful planning in order to guarantee that the parts are assembled, or disassembled, in a correct and efficient sequence. This planning is needed regardless of whether the assembly is executed by humans or by robots.

Because of the size and complexity of these truss structures, even trained humans may fail to detect *dead-end* sequences until a lot of work has been done and it is found that the overall assembly cannot be completed. In the case of a repair in which a faulty strut is to be replaced, an ill-planned disassembly sequence may lead to an irreparable collapse of the whole truss structure.

In addition to the difficulty humans have in guaranteeing correctness in the planning process, they often fail to notice which possibilities for the sequences are the most efficient. This difficulty is further aggravated by constant changes in the measure of the efficiency of the assembly sequence. For example, the efficiency may be measured by the total time it takes to complete the assembly in one case, and by the total energy in another case.

Moreover, humans typically are slow in generating assembly sequences. There are many situations in which the sequence planning must also be expeditious. Speed in sequence generation is particularly important in the case of a repair in which a faulty strut is to be replaced. It is virtually impossible to preplan for every conceivable repair that may be needed. Speed in sequence generation is also important in the design of the truss structures because it allows the difficulty of assembly to be considered in the design process. A designer may also want to take into account the difficulty of repairing different structures.

Therefore, there is a need to systematize and to computerize the generation of the correct assembly, disassembly, and repair sequences, as well as the selection of the best solution. Systematization is needed in order to guarantee that the sequences generated are correct and efficient. Computerization is needed in order to enable the fast generation of assembly sequences. In the case of robotic assembly and repair, the software for planning assembly, disassembly, and repair sequences will augment the array of functions that robots are able to perform autonomously.

Most previous work on assembly planning focused on electromechanical and electronic devices [6]. This paper presents an assembly sequence planner for tetrahedral truss structures. To overcome the difficulties due to the large number of parts, the planner exploits the simplicity and uniformity of the shapes of the parts and the regularity of their interconnection. The planning automation is based on the computational formalism known as *production system*. The

*This research was conducted at the Jet Propulsion Laboratory, California Institute of Technology, under contract with the National Aeronautics and Space Administration.

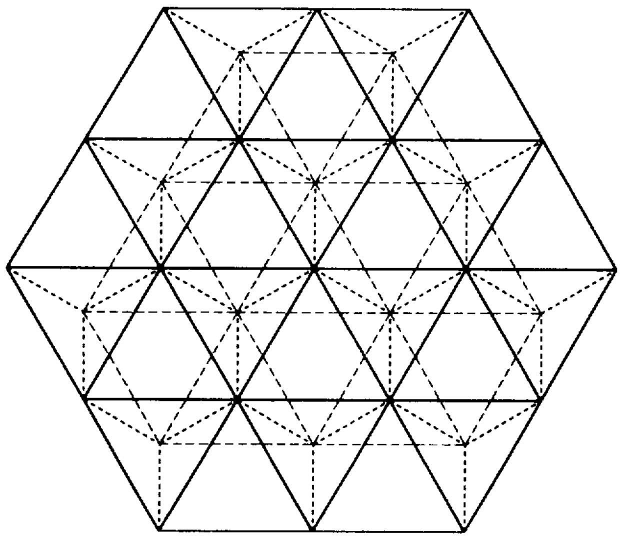

Figure 1: Tetrahedral truss structure.

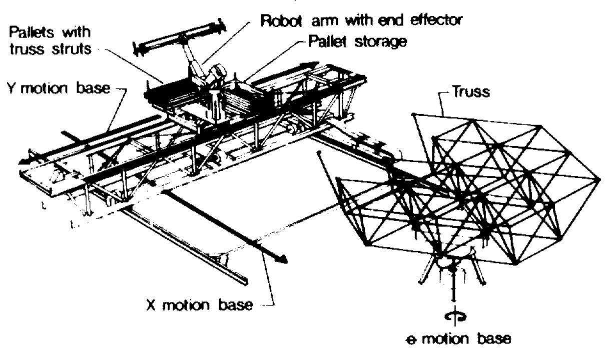

Figure 2: Robotic assembly facility at the NASA Langley Research Center.

global database consists of an hexagonal grid representation of the truss structure. This representation captures the regularity of tetrahedral truss structures and their multiple hierarchies. It maps into quadratic grids and can be implemented in a computer by using a two dimensional array data structure. By maintaining the multiple hierarchies explicitly in the model, the choice of a particular hierarchy is only made when needed, thus allowing a more informed decision. Furthermore, testing the preconditions of the production rules is simple because the patterned way in which the struts are interconnected is incorporated into the topology of the hexagonal grid. A directed graph representation of assembly sequences allows the use of both graph search and backtracking control strategies.

2 Scenario

A truss structure is a composition of struts interconnected at nodes and forming a stable and rigid unit. All struts are identical, and so are all nodes. The struts are attached to the nodes through joint connectors. A strut s_i is said to belong to a node n_j if one of s_i's ends is attached to n_j. Similarly, node n_j is said to belong to strut s_i.

In this paper, the robotic assembly facility of the NASA Langley Research Center [12, 13, 15] will be used as the reference scenario. Figure 2 shows that facility in schematic form. The robot arm is mounted on a base that is mounted on a carriage that can translate along one direction. The base where the robot is mounted can translate along a direction orthogonal to the carriage translations. These two motions allow the positioning of the robot arm in a Cartesian coordinate system. The truss structure is mounted on a base that can rotate. If necessary, before a strut is assembled, the structure is turned and both the base of the robot arm and the carriage are translated.

The assembly process consists of a succession of tasks, each of which is the addition of one strut. The nodes are preattached to their first strut to be assembled. Whenever a strut is added, it is attached to its two nodes, except when the nodes have been preattached. The process starts with all struts stored in pallets that are stacked on the same base where the robot arm is mounted. The assembly process ends with all struts properly joined to form the whole structure. Ideally, after struts have been added, they are not removed until the end of the assembly process.

An assembly task is said to be feasible if there is a collision-free path to bring the strut to its position in the structure from a situation in which it is far apart, and if it is possible to lock the joints that attach the strut to its nodes. Of course, the path should also avoid collisions between the robot arm or the carriage and the truss structure.

It is desired to create a computer system that will generate a sequence of assembly tasks for any given tetrahedral truss structure. Of course, the input to this system includes a description of the desired structure. In addition to containing only feasible tasks and achieving the assembly of the whole structure, the sequence produced should minimize a given cost function. Although the definition of the cost functions is part of the problem, it will probably include a weighed combination of reliability, safety, energy and total assembly time.

3 Background on assembly sequence planning

Most previous work on assembly sequence planning [2, 3, 4, 5] focused on electromechanical and electronic devices such as gearboxes, alternators and disk drives. The difficulty in planning the assembly sequence for those products stems, in some degree, from the variety of part shapes and from the lack of regularity in the way the pieces are interconnected. To overcome this difficulty, previous approaches used elaborate representations of mechanical assemblies and complex geometric and symbolic reasoning techniques [1, 4, 16].

Another difficulty in the automation of assembly sequence planning for electromechanical and electronic devices comes from the fast growth in the required computation with the increase in the number of parts. Previous approaches have overcome this problem by clustering components into subassemblies [8], thereby artificially reducing the number of parts. Many large products have natural subassemblies that arise as a result of modular design as well as of manufacturing advantages. Clustering components into subassemblies sacrifices completeness since sequences that interleave the assembly of parts of different subassemblies can not be generated. But for most large products this loss of completeness is not a serious limitation because those natural subassemblies are assembled independently anyway. In practice, a hierarchical model of the assembly [14] is used to implement the clustering of parts. At the highest level, each subassembly is treated as a part.

There has been substantial progress in assembly planning in recent years, and several new approaches and techniques have been reported[6]. Nevertheless, because of the complexity of the reasoning involved and the large size of the solution space, it is still impracticable to use existing planners to generate an assembly sequence for assemblies containing a large number of parts, such as the structure shown in Figure 1, which is made of 102 struts.

One way to reduce the computation, when planning the assembly of tetrahedral truss structures, is to cluster the struts into subassemblies as in the case of electromechanical and electronic devices. A truss structure such as the one shown in Figure 1 can be viewed as the composition of tetrahedral and pentahedral units, much like a solid that has a complex shape can be treated as the composition of simple solids that have faces in contact, one against the other. Two adjacent units share the struts and nodes of their "contacting" faces. For example, the small truss structure shown in Figure 3a can be regarded as the composition of the two pentahedral units shown in Figure 3b. Those two pentahedrons have one face "in contact" and they share the struts and nodes of that face. Similarly, the structure shown in Figure 1 can be viewed as a composition of tetrahedral and pentahedral units with faces "in contact."

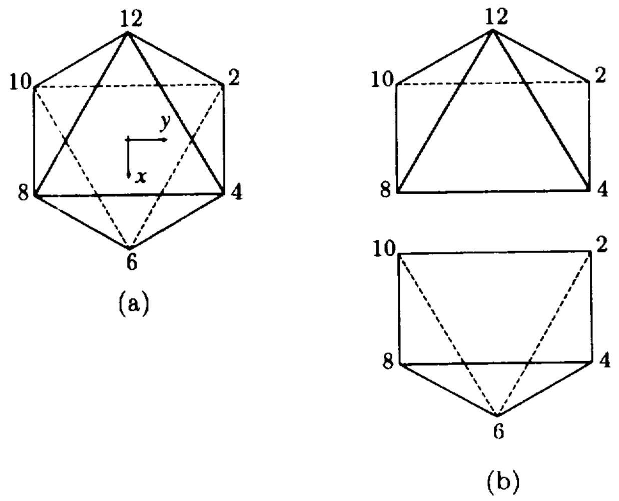

Figure 3: A small truss structure and its subdivision into two pentahedral units.

Since in practice it is preferred to finish the assembly of one unit before beginning the assembly of another [11], seeing the structure as a composition of units should not be a problem in the assembly sequence planning. Furthermore, this approach should reduce the computation required to create the assembly sequence. Mathur and Sanderson [9] describe a hierarchical planner for truss structures. For the structure shown in Figure 1, for example, using this approach corresponds to reducing the number of parts from 102, which is the number of struts, to 37, which is the number of units.

The use of a hierarchical approach for planning the assembly requires that the tetrahedral truss structure be subdivided into pentahedral and tetrahedral units. But there are several ways to cluster the struts into those kinds of unit. For example, in addition to the subdivision shown in Figure 3b, there are two other ways in which the small truss structure shown in Figure 3a can be subdivided into two pentahedral units, and these are shown in Figure 4. Unlike the electromechanical and electronic devices studied previously, in the case of truss structures, there is no manufacturing advantage in choosing one subdivision over the others. Instead of having one natural hierarchy of the parts, tetrahedral truss structures have several hierarchical models, none of which is "more natural" than the others. When the small structure shown in Figure 3 is part of a large structure (e.g. Figure 1), unless the best assembly sequence is known in advance, choosing one of its subdivisions to create a hierarchical model will likely preclude the generation of the best assembly sequence. Therefore, it is important that the representation of the problem captures the multiple hierarchies that occur in a tetrahedral

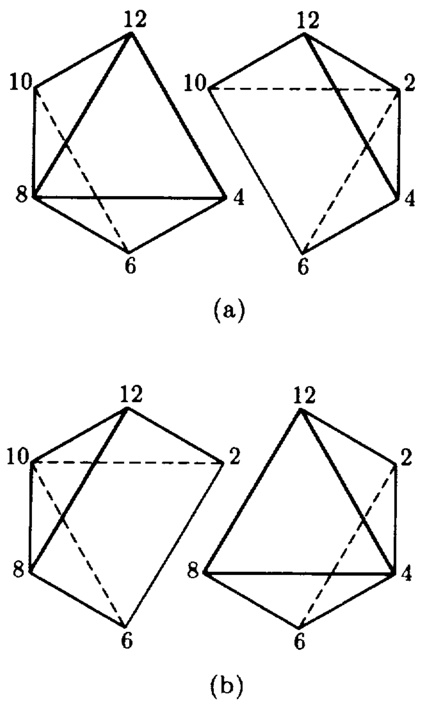

Figure 4: Two additional subdivisions of the small truss structure (Figure 3a) into two pentahedral units.

truss structure.

A second way to reduce the computation, when planning the assembly of tetrahedral truss structures, is to take advantage of the simplicity and uniformity of the shapes of the parts and the regularity of their interconnection. Unlike the electromechanical and electronic devices studied previously, the tetrahedral truss structures are made of struts, all of which have the same cylindrical shape. Moreover, those struts are interconnected in a regular fashion. Because the parts have the same shape and are interconnected in a patterned way, the model of a truss structure can incorporate the geometry of the set of parts in a more explicit way than the models used for electromechanical and electronic devices.

The models of assemblies that have been used in previous work describe the shapes of all parts and the geometric and mechanical relationships between parts. Typically, assembly models can be associated with graphs in which the vertices correspond to the parts and the edges to the geometric relationships between parts [14]. The topology of the graph corresponds to the topology of the parts in the assembly. But there is no relation between the geometry of the set of parts in the assembly and the topology of the graph. Figure 5 shows three simple assemblies made up of the same set of parts. Those assemblies are associated with the same graph, also shown in Figure 5, since the topology of the parts is the same. But the

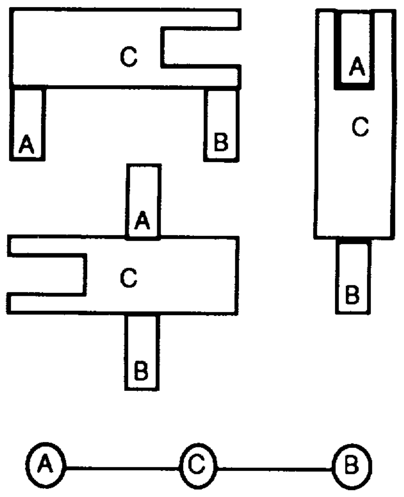

Figure 5: Three assemblies with different geometry but same topology.

geometry of the parts in each assembly is very different from the geometry of the parts in the others.

The next section presents an assembly planner that uses a multihierarchical representation for the truss structures and that takes advantage of the simplicity and uniformity of the shapes of the parts and the regularity of their interconnection.

4 Planning the assembly of tetrahedral truss structures

The computational formalism known as production system [10] has been used for the automatic generation of assembly sequences for tetrahedral truss structures. There are three major elements in a production system: the global database, the set of production rules, and the control scheme. This section describes these three elements. Subsection 4.1 presents a multihierarchical representation of tetrahedral truss structures that constitutes the global database; subsection 4.2 discusses the control scheme; and subsection 4.3 introduces the production rules that act on the global database.

4.1 A multihierarchical representation of tetrahedral truss structures

It was mentioned in section 3 that, unlike the electromechanical and electronic devices studied previously, the tetrahedral truss structures are made of simple struts all of which have the same shape. In addition, the struts are interconnected in a very regular

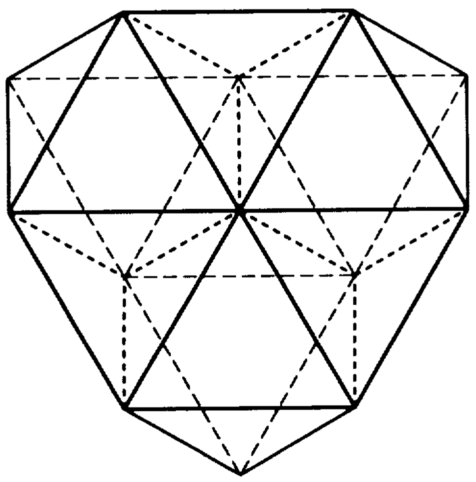

Figure 6: A subset of the structure shown in Figure 1.

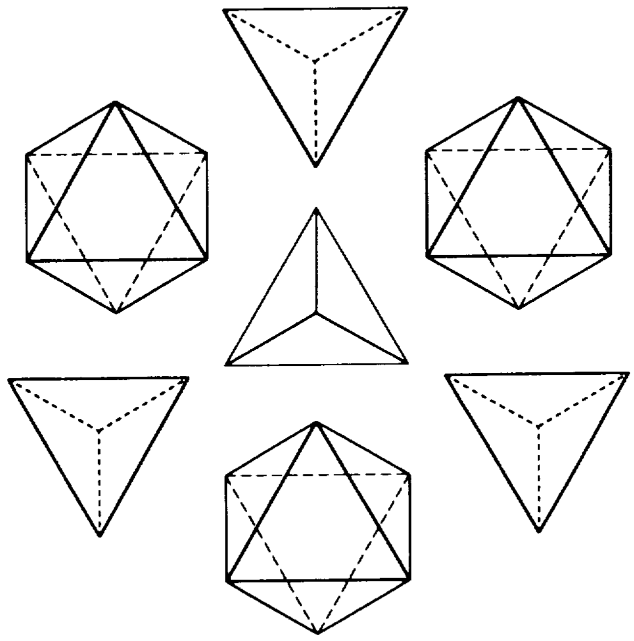

Figure 7: The building blocks of the structure shown in Figure 6.

fashion. The method for planning assembly sequences of tetrahedral truss structures can take advantage of these facts to reduce the computation needed to generate assembly sequences.

The representation to be introduced next is based on viewing tetrahedrons and octahedrons as the building blocks of a tetrahedral truss structure. A pentahedron will be considered to be a building block only when it is not embedded in any octahedron[1]. Figure 6 shows a subset of the structure depicted in Figure 1, and Figure 7 shows its building blocks.

As discussed above, there are six embedded pentahedrons in an octahedron. The nodes of the octahedron shown in Figure 3a have been numbered by analogy with the numbers in a clock face. The embedded pentahedrons are designated by the number of the vertex corresponding to their apex. Therefore, the pentahedrons shown in Figure 3b are referred to as P12 (top) and P6 (bottom); the pentahedrons shown in Figure 4a are referred to as P8 (left) and P2 (right); and the pentahedrons shown in Figure 4b are referred to as P10 (left) and P4 (right).

Figure 3a shows a coordinate frame associated with an octahedron. The x–y plane contains nodes 2, 6, and 10. The z axis points out of the figure[2]. The octahedrons in a truss structure are all parallel to each other. Therefore, the transformation between the coordinate frames of two octahedrons is a pure translation.

Some tetrahedrons have three nodes on the top plane and one node on the bottom plane. These are referred to as *tetrahedron-down* because they can be viewed as a pyramid pointing down. The other tetrahedrons, which have three nodes on the bottom plane and one node on the top plane, are referred to as *tetrahedron-up*. All tetrahedrons-up in a truss structure are parallel to each other, and all tetrahedrons-down are parallel to each other. Therefore, the transformation between the coordinate frames of two tetrahedrons-up (or two tetrahedrons-down) is a pure translation.

A tetrahedral truss structure can be represented by a graph in which the vertices correspond to volumetric units, and the edges correspond to "face-contacts" between adjacent units. Figure 8 shows a graph representation for the 102-strut truss structure shown in Figure 1.

The geometry of this graph parallels that of the truss structure. Because of the regularity of the structure, its graph representation constitutes an hexagonal grid. In addition, the hexagonal grid can be mapped into a rectangular grid as shown in Figure 8, where the lines and columns are labeled with their indices. Furthermore, a coordinate frame can be associated with the graph shown in Figure 8: the x axis points down, and the y axis points right.

There are three types of vertices, represented, respectively, by hexagons, triangles, and half-hexagons. Hexagon vertices correspond to octahedrons like the one shown in Figure 3a. Triangles correspond to tetrahedrons; triangles pointing down in the figure correspond to *tetrahedrons-down*, and triangles pointing up correspond to *tetrahedrons-up*.

Unlike regular graphs, in this representation the position and the orientation of the vertices are important. A coordinate frame is associated with the

[1]Pentahedrons not contained in any octahedron occur at the periphery of a structure. See Figure 8.

[2]Nodes 4, 8, and 12 have positive z coordinate.

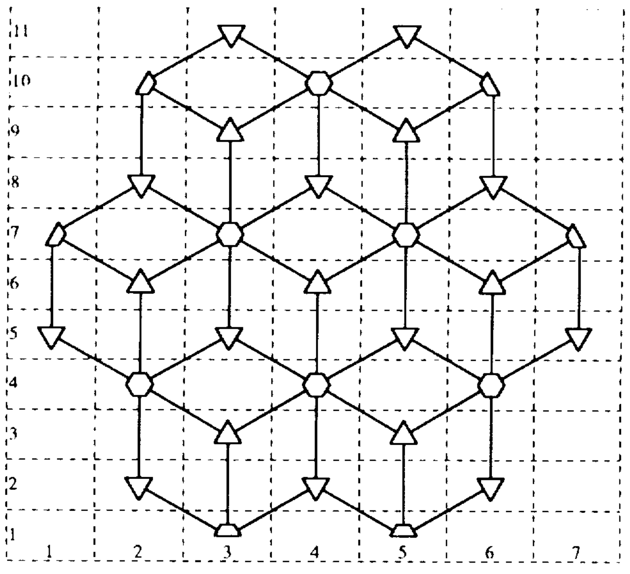

Figure 8: Graph representation for a tetrahedral truss structure with 102 struts and its mapping into a rectangular grid.

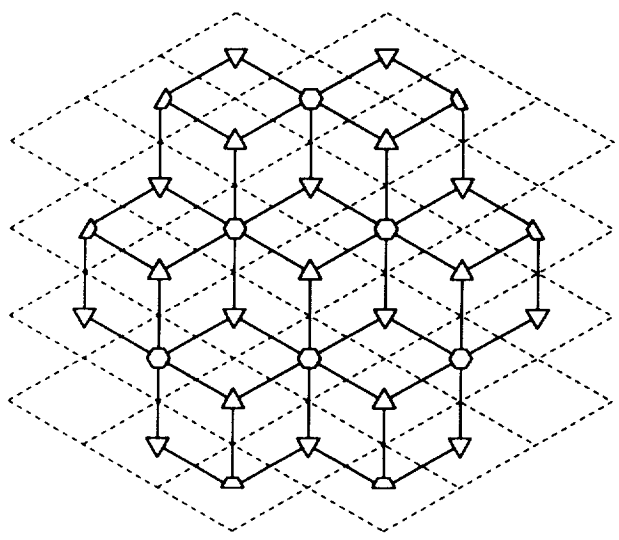

Figure 10: The mapping of the graph representation of tetrahedral truss structures into a quadratic (not rectangular) grid.

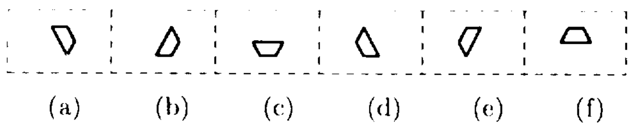

Figure 9: The six orientations in which half-hexagons may occur: (a) P2; (b) P4; (c) P6; (d) P8; (e) P10; (f) P12.

graph. Its axes are parallel to those of the frames associated with the octahedrons. Each vertex is oriented as its unit's parallel projection on the bottom plane of the structure.

The half-hexagons correspond to pentahedrons such as those shown in Figures 3b, 4a, and 4b. The half-hexagons are used only when there is no octahedron that includes the corresponding pentahedron. Since there are six pentahedrons embedded in each octahedron, there are six orientations in which the half-hexagons may occur, and they are shown in Figure 9.

The edges in the graph representation of a truss structure correspond to those "faces" that are common to two adjacent volumetric units, that is, those sets of three struts that "belong" to both volumetric units.

The mapping of the graph representation into a rectangular grid gives rise to a data-structure for a computer implementation: a two-dimensional array in which each element may contain information about one building block of the truss structure. The indices of the array element indicate the position of the building block.

The edges in the graph shown in Figure 8 are only implicitly encoded into the two-dimensional array data structure. In addition, the contacts between units that share only one strut, or only one node, are also implicitly encoded into the array. For example, a tetrahedron-up at cell $(i\ ,\ j)$ (i.e. line i, column j) shares one strut with the tetrahedron-down at cell $(i+2\ ,\ j)$, another strut with the tetrahedron-down at cell $(i-1\ ,\ j-1)$, and another strut with the tetrahedron-down at cell $(i-1\ ,\ j+1)$. As another example, an octahedron at cell $(i\ ,\ j)$ shares one node with the tetrahedron at cell $(i+2\ ,\ j-2)$.

It should be pointed out that Figure 8 shows one mapping from the graph representation of tetrahedral truss structure, which is an hexagonal grid, into a rectangular grid. That mapping is probably the most direct, but it leaves a number of empty cells. In a computer implementation, if the available storage space is scarce, it is straightforward to devise other mappings from the hexagonal grid into a quadratic grid, which may not be rectangular. Figure 10 shows another mapping from the graph representation of truss structures into a quadratic grid. Unlike the one shown in Figure 8, the mapping shown in Figure 10 does not leave empty cells.

As it will become clear in the following subsections, this graph representation of truss structures allows an assembly planner to exploit the regularity in which the parts are joined to improve its planning efficiency. This improvement is due, in part, to the encoding of the geometry of the truss structure in the topology of

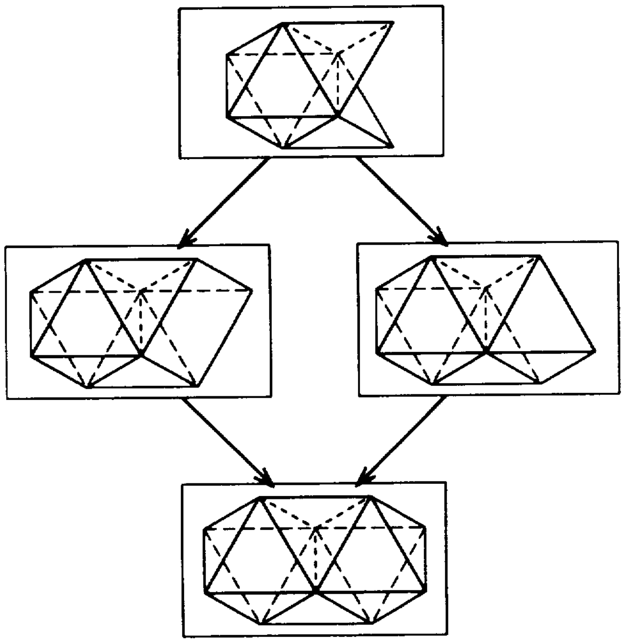

Figure 11: A portion of the directed graph of assembly sequences, which is also shown in Figure 12. The vertices have been labeled by the top view of the partial truss structure at each state of the assembly process.

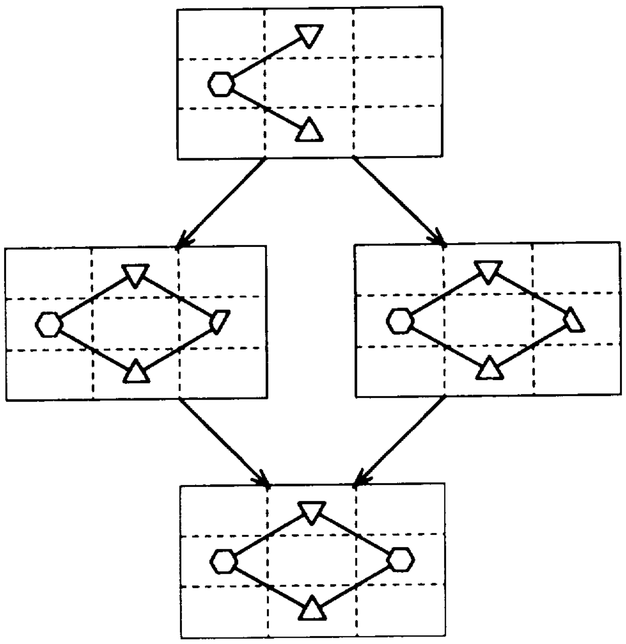

Figure 12: A portion of the directed graph of assembly sequences, which is also shown in Figure 11. The vertices have been labeled by the graph representation of the partial truss structure at each state of the assembly process.

the graph. Moreover, the graph representation also allows the planner to take advantage of the multiple hierarchies that exist in tetrahedral truss structures. The decision of which hierarchy to choose does not have to be made until it is needed. Being able to delay the selection of the hierarchy, the planner will have more information available to decide which hierarchy is more advantageous, and therefore will be able to make a better choice.

4.2 Control strategy

Several methodologies for representing assembly sequences have been utilized [7], including representations based on directed graphs and AND/OR graphs.

As mentioned in section 4, it is preferred to complete the assembly of a tetrahedral or pentahedral unit before beginning the assembly of another unit [11]. Therefore, the assembly task can be redefined as the assembly of one tetrahedron or one pentahedron. In this definition, each assembly task consists of a sequence of subtasks, each being the assembly of one strut.

Since each assembly task is the addition of exactly one volumetric unit, both the directed graph and the AND/OR graph will have the same size. The directed graph representation has been used in this work because it is simpler and easier to understand and implement. The vertices in this directed graph correspond to the states of the assembly process that can be characterized by the description of the substructure already assembled. The edges in this directed graph represent the assembly tasks, each corresponding to the addition of one volumetric unit.

Figures 11 and 12 show a portion of the directed graph of assembly sequences. In Figure 11 the vertices have been labeled by the top view of the partial truss structure at each state of the assembly process. This labeling is better for displaying the assembly sequences for humans. In Figure 12 the vertices have been labeled by the graph representation of the partial truss structure at each state of the assembly process. This labeling reflects more closely the computer internal representation of the assembly sequences. In both Figures, the vertex at the top corresponds to a state in which one octahedron and two tetrahedrons are already assembled. The two vertices in the middle corresponds to states in which an additional pentahedron is already assembled. In the left vertex, the additional pentahedron is P10, and in the right vertex, the additional pentahedron is P8. The vertex at the bottom corresponds to a state in which two octahedrons and two tetrahedrons are already assembled.

Figure 12 also illustrates the advantage of using the multihierarchical representation of tetrahedral truss structures introduced in section 4.1. Because the building blocks are tetrahedrons and octahedrons, it is possible to generate sequences that use different sets of pentahedrons as assembly tasks. As pointed out above, the additional pentahedron in the left middle vertex is not the same as the one in the right middle vertex. By using the representation in Figure 8, the three possibilities in which an octahedron can be subdivided can be considered. In the scenario described in section 2, the two possibilities corresponding to the subdivision of the right octahedron into P6 and P12 are not considered valid. If the structure had been viewed as a composition of pentahedrons and tetrahedrons, only one alternative would be considered.

Each assembly sequence corresponds to a path in the directed graph of assembly sequences, starting in the vertex that has no label (i.e., no strut has been assembled) and ending in the vertex that is labeled by the whole truss structure. By construction, the directed graph of assembly sequences has no cycle. A measure that reflects the quality of an assembly sequence can be computed by assigning costs to the vertices (i.e., the states of the assembly process) and to the edges (i.e., the assembly tasks). The cost of a path p can be defined recursively as:

$$cost(p) = \begin{cases} C_S(s_p) & \text{if the path has only one node} \\ C_S(s_p) + C_T(t_p) + cost(r_p) & \text{otherwise} \end{cases}$$

where s_p is the initial vertex (state) of p, t_p is the initial edge (task) of p, and r_p is the tail of p, that is, what is left of p after s_p and t_p are removed. The function C_S gives an assessment of the quality of a state of the assembly process. Better (e.g., more stable) states correspond to smaller values of C_S. The function C_T gives an assessment of the quality of a task in the assembly process. Better (e.g., less complex or less time consuming) tasks correspond to smaller values of C_S.

The directed graph representation of assembly sequences and its associated cost function allow both backtracking and graph search control regimes [10] to be implemented. The construction of the assembly sequence can proceed in backward or forward fashion. The former is easier to understand while the latter may be more efficient, since it avoids dead-end states. Subsection 4.4 describes the current implementation.

4.3 Production rules

The global database introduced in subsection 4.1 reflects the state of the truss structure at each point of the assembly process. The production rules that are introduced in this subsection contain the conditions for the execution of an assembly task and the changes that occur in the state of the truss structure when that task is executed. In the operation of the planning system, whenever a production rule is applied, the global database must be updated to reflect the changes in the state of the truss structure.

- Precondition:
  1. Cell (i,j) currently contains a pentahedron Pk.
  2. Goal is one octahedron in cell (i,j).
  3. Any cell (x,y) for which $L(x,y,i,j,k) > 0$ is empty where $L(x,y,i,j,k) = \alpha(k) \cdot x + \beta(k) \cdot y + \gamma(i,j,k)$.
- Effect:
  1. Adjust the angle of the truss structure and the x–y position of the robot arm acording to the position of cell (i,j).
  2. Install pentahedron Pk' in cell (i,j) where $k' = \text{rem}(6+k, 12)$.

Figure 13: Production rule example. See Table 1.

The simplest way to introduce the production rules is by an example. Figure 13 shows one production rule. It corresponds to the assembly task that finishes up one octahedron, starting with one of its pentrahedron halves already assembled. If the pentahedron already assembled is Pk, the pentahedron that will complete the octahedron is Pk' where $k' = \text{rem}(6+k, 12)$.

The first two preconditions simply verify that the goal is an octahedron in a cell (i,j) that currently has a pentahedron. The third precondition verifies that no collision will occur between the truss structure and the carriage where the base of the robot is mounted. It requires that all cells on the side of the line $L(x,y,i,j,k) = 0$ where Pk' is must be empty. Figure 14 shows a state in which the preconditions of the production rule in Figure 13 are satisfied for cell $(7,5)$ and $k = 10$.

The effect of this production rule is the installation of pentahedron Pk' in cell (i,j). This can be accomplished by using a precompiled sequence of subtasks, each of which is the addition of one strut. Since the base of the pentahedron is already in place, only four struts must added. This subsequence of tasks, each of which includes the motions of the robot arm, is independent of the position of the cell (i,j). Of course, the positions of the carriage and of the base, as well as the angle of the structure, must be adjusted according to the position of cell (i,j).

For each possible geometric configuration that a cell can take, there is a production rule similar to the

Table 1: Coefficients of $L(x, y, i, j, k)$ in the production rule example shown in Figure 13.

k	$\alpha(k)$	$\beta(k)$	$\gamma(i,j,k)$
2	-1	-3	$(i+3j)$
4	1	-3	$(3j-i)$
6	1	0	$-i$
8	1	3	$-(i+3j)$
10	-1	3	$(3j-i)$
12	-1	0	i

one in Figure 13. Since there are only a few geometric configurations, the total number of production rules is small.

4.4 Current implementation

The current implementation is an interactive production system that uses a backtracking control scheme. The assembly sequences are generated in a forward fashion. The first unit to be assembled is given.

At each step, a menu containing all the subunits that can be assembled next is displayed for the user. These options are obtained by testing the preconditions of the production rules. The alternatives in the menu are ranked according to the system's preference criterion. The user may accept the system's choice for the next subunit or may select another among those that are feasible. A graphical display of the truss structure allows the user to visualize the available options. At any point, the user can force the system to backtrack and to "undo" one or more assembly tasks.

This interactive production system exploits the strengths of humans and computers. Computers are better at guaranteeing that the sequence is correct and that no option is overlooked. Humans are better at assessing the quality of an assembly sequence.

The cost function that is used is a function of the translation of the carriage, the translation of the base, and the rotation of the structure. The shorter those motions, the lower the cost function. The task for which the cost function is minimal has the highest preference. Other cost functions are being investigated, and one of the goals of this project is to find good cost functions and their corresponding heuristic estimations.

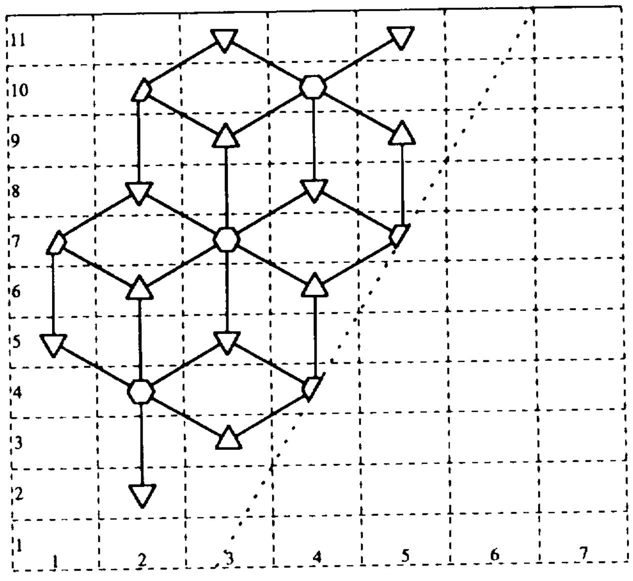

Figure 14: A state in which the preconditions of the production rule in Figure 13 are satisfied for cell (7, 5) and $k = 10$.

In each assembly task, a number of struts are assembled. For example, in the tasks corresponding to the effect of the production rule shown in Figure 13, four struts are assembled. By properly positioning the carriage and the base of the robot, the arm motions to install a given strut is the same regardless of the position of the octahedron that is being completed. In the current implementation, these motions were taught. Each production rule is associated with the paths to install the struts of its corresponding subunit. Therefore, the output of the planning system includes, for each strut, the positions of the carriage and the base of the robot, the angle of the truss structure, and the specific arm motion to be used.

5 Conclusion

This paper brought about a clear understanding of the regularity of the tetrahedral truss structures and their multiple hierarchies. Unlike electromechanical and electronic devices, tetrahedral truss structures can be represented by a graph whose topology corresponds to the geometry of the parts. This representation captures the regularity of the truss structure as well as all its hierarchies. It consists of an hexagonal grid that can be mapped into a two-dimensional array data structure. The relationships between units are implicitly encoded by the indexes of their corresponding cells in the array.

Using this representation and its associated data structure, a simple reasoning is sufficient to decide whether or not a candidate assembly task is feasible. Furthermore, the choice between hierarchies can be made as the plan is generated, thus allowing a better selection than if the choice were made in advance.

A prototype planning system that uses the production system paradigm has been implemented. The global database is the hexagonal grid representation of tetrahedral truss structures. There is one production rule for each possible configuration that a cell can take. Since there are only a few geometric configurations the total number of production rules is small. A directed graph representation of assembly sequences allows the use of both graph search and backtracking control strategies. The prototype uses a backtracking scheme.

This current implementation is interactive and exploits the strengths of humans and computers. Computers are better at guaranteeing that the sequence is correct and that no option is overlooked. Humans are better at assessing the quality of an assembly sequence. For the structure shown in Figure 1, the system generated an assembly sequence that significantly reduces the amount of rotation when compared to a sequence generated by hand. Future work will focus on cost functions and heuristic evaluations aimed at making the system fully autonomous.

Acknowledgments

The author would like to thank R. S. Desai for support of this research, and M. D. Rhodes and R. W. Will for valuable feedback on the planning system and for kindly providing Figure 2.

References

[1] D. F. Baldwin. *Algorithmic Methods and Software Tools for the Generation of Mechanical Assembly Sequences.* Master of science thesis, Department of Mechanical Engineering, Massachusetts Institute of Technology, February 1990. Also published as report CSDL-T-1040, The Charles Stark Draper Laboratory.

[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. Corrections *ibid* RA-4(6):705–708, December 1988.

[4] J. M. Henrioud. *Contribution a la Conceptualisation de l'Assemblage Automatisé: Nouvelle Approche en vue de Détermination des Processus d'Assemblage.* Thèse d'état, Université de Franche-Comté, Besançon, France, December 1989.

[5] L. S. Homem de Mello. *Task Sequence Planning for Robotic Assembly.* PhD thesis, Carnegie Mellon University, May 1989.

[6] L. S. Homem de Mello and S. Lee, editors. *Computer-Aided Mechanical Assembly Planning.* Kluwer Academic Publishers, 1991.

[7] L. S. Homem de Mello and A. C. Sanderson. Representations of Mechanical Assembly Sequences. *IEEE Transactions on Robotics and Automation*, 7(2):211–227, April 1991.

[8] S. Lee and Y. G. Shin. Assembly Planning Based on Subassembly Extraction. In *Proc. IEEE Int. Conf. on Robotics and Automation.* IEEE Computer Society Press, May 1990.

[9] R. K. Mathur and A. C. Sanderson. A Hierarchical Planner for Space Truss Assembly. *Cooperative Intelligent Robotics in Space*, Rui J. deFigueiredo, William E. Stoney, Editors, Proc. SPIE 1387, 47–57, 1991.

[10] N. J. Nilsson. *Principles of Artificial Intelligence.* Tioga, 1980.

[11] M. D. Rhodes. Guidelines for Development of Truss Assembly Scenario. Unpublished technical notes, March 1990.

[12] M. D. Rhodes and R. W. Will. Automated Assembly of Large Space Structures. In *41st Int. Astronautical Congress*, October 1990.

[13] M. D. Rhodes, R. W. Will, and M. A. Wise. A Telerobotic System for Automated Assembly of Large Space Structures. NASA Technical Memorandum 101518, Langley Research Center, Hampton, VA, March 1989.

[14] S. Srikanth and J. U. Turner. Toward a Unified Representation of Mechanical Assemblies. *Engineering with Computers*, 6:103–112, 1990.

[15] R. W. Will and M. D. Rhodes. An Automated Assembly System for Large Space Structures. *Cooperative Intelligent Robotics in Space*, Rui J. deFigueiredo, William E. Stoney, Editors, Proc. SPIE 1387, 60–71, 1991.

[16] R. Wilson and J. F. Rit. Maintaining Geometric Dependencies in an Assembly Planner. In *Proc. IEEE Int. Conf. on Robotics and Automation*, pages 890–895. IEEE Computer Society Press, May 1990.