N89 - 25161

STRUTEX
A PROTOTYPE KNOWLEDGE-BASED SYSTEM FOR INITIALLY CONFIGURING A STRUCTURE TO SUPPORT POINT LOADS IN TWO DIMENSIONS

James L. Rogers
NASA Langley Research Center
Hampton, Virginia

Jaroslaw Sobieszczanski-Sobieski
NASA Langley Research Center
Hampton, Virginia

INTRODUCTION

Engineers and managers are always concerned about reducing the costs and time involved in completing a design project. Therefore, many hours of research have been devoted to speed and sensitivity improvements in the area of structural analysis. Additional research effort has been applied to the improvement of optimization algorithms. From a numerical standpoint, these areas are nearing a point of diminishing returns when using conventional computer hardware. However, one area which shows a potential for reducing design cost and time, but has had little research, is the determining and refining of an initial configuration before beginning the analysis and optimization process (figure 1). One reason is because this is a problem that is not easily solved numerically, but one that seems to require using heuristics from experienced designers.

Only recently have engineers begun making use of Artificial Intelligence (AI) tools in the area of conceptual design (refs. 1,2). To continue filling this void in the design process, a prototype knowledge-based system, called STRUTEX (ref. 3), has been developed to initially configure a structure to support point loads in two dimensions. This prototype was developed for testing the application of AI tools to conceptual design as opposed to being a testbed for new methods for improving structural analysis and optimization. This system combines numerical and symbolic processing by the computer with interactive problem solving aided by the vision of the user.

This paper describes how the system is constructed to interact with the user. Of special interest is the information flow between the knowledge base and the data base under control of the algorithmic main program. Examples of computed and refined structures are presented during the explanation of the system.

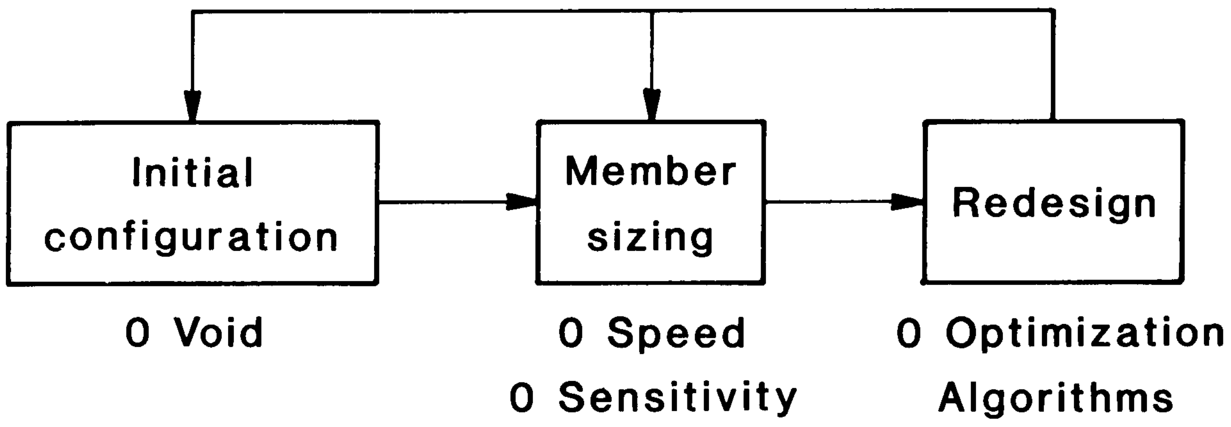

Figure 1

COMPONENTS OF THE SYSTEM

The main driver program for STRUTEX is written entirely in FORTRAN. Other components were added by linking existing software - DI-3000 (ref. 4) for the graphics, RIM (Relational Information Management, ref. 5) for the relational data base management, and CLIPS (C Language Production System, ref. 6) for the inference engine - to the main driver program. The data for RIM and the knowledge base (rules) for CLIPS are maintained in different files separated from STRUTEX. EAL (Engineering Analysis Language, ref. 7) for the structural analysis, and CONMIN (Constraint Minimization, ref. 8) for the optimization are coupled in PROSSS (Programming System for Structural Synthesis, ref. 9) to perform the analysis and optimization (figure 2).

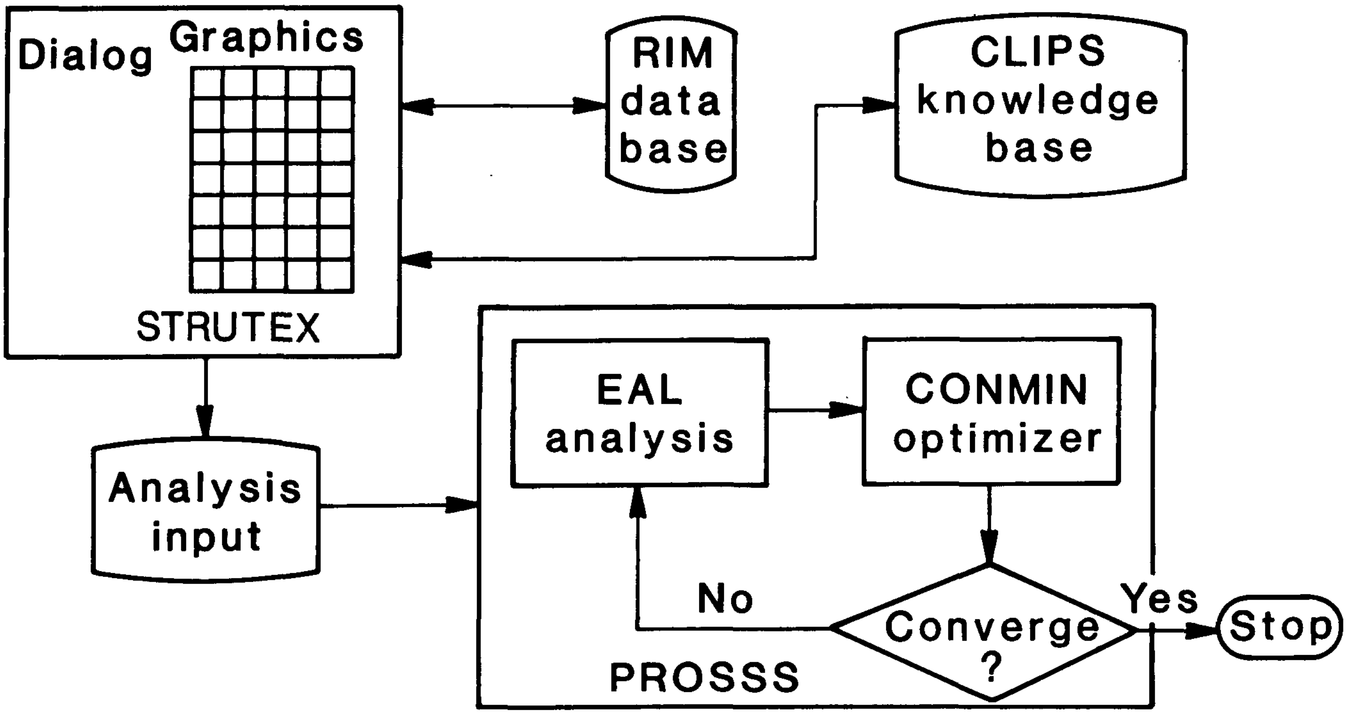

Figure 2

A PRODUCTION RULE KNOWLEDGE BASE/INFERENCE ENGINE

CLIPS is a knowledge-based system tool developed at NASA Johnson Space Center. It is written in C, performs forward chaining based on the Rete pattern matching algorithm, and has a FORTRAN interface. The knowledge base is composed of rules which are defined by the "defrule" construct. A rule states specific actions, the Right-hand side (RHS), that are to be taken if certain conditions, the Left-hand side (LHS) are met. An "=>" separates the LHS and the RHS. If, and only if, all of the conditions on the LHS are satisfied, then the actions on the RHS are performed sequentially. Each rule must contain at least one condition and one action.

Pieces of information represented by facts, the basic form of data in CLIPS, are contained in a facts-list. A fact can contain a number, a word, or a string. Facts are asserted into the facts-list before execution by the "deffacts" construct or by an assert command in the calling program, or during execution as the action caused by executing a rule. A rule executes based on the existence or non-existence of facts in the facts-list. For example, the rule for selecting a string as the type of support is shown in figure 3. It is read: If the location of the support surface is above the load and the load is a gravity type of load, then the support type is a string. This rule will execute when the two facts (SURFLC ABOVE) and (PLOADT GL) are asserted from STRUTEX and placed into the facts list. The actions, based upon a match on the two facts (conditions), are to return to STRUTEX via the KBANS1 parameter the fact that the support is a string, and to assert the fact that the support is a string into the facts-list. The KBANS1 parameter, discussed below in more detail, is the name of the subroutine in STRUTEX. In this example, only the parameters SUPPORT and STRING are needed. The 0.0 is a dummy parameter.

Currently there are only thirteen rules in the knowledge base. There are three rules for determining the type of support, beam, truss, or string. The rules for choosing the beam or truss are more complex than that of the string and use an explicit "or" coupled with three or four explicit "and"s. Another rule in the knowledge base explains the choice of support type when executed. The remainder of the rules determine whether or not bracing is required and the type of bracing that is required in a truss. Other rules determine if there are any angles formed which are not within a given range. If so, a recommendation is made to correct the problem. The action parts of the bracing rules are more complex than those of the rules for choosing the type of support. Some of the bracing actions are based on mathematical computations within the rule, while others have choices of actions within a single rule with the choice being determined by the facts.

The inference engine in CLIPS applies the knowledge (rules) to the data (facts). Pattern matching occurs on the LHS. The basic execution cycle begins by examining the knowledge base to see if the conditions of any rules have been met. All rules with currently met conditions are pushed onto the "agenda" which is essentially a pushdown stack. Once the agenda is complete the top rule is selected and the RHS is executed. As a result of these actions, new rules can be placed on the agenda and rules on the agenda may be removed. This cycle is repeated until all rules that can execute have done so.

```
(defrule string
  (SURFLC ABOVE)
  (PLOADT GL) =>
     (assert (SUPPORT STRING))
     (KBANS1 SUPPORT STRING 0.0))
```

Figure 3

FLOW OF DATA BETWEEN THE KNOWLEDGE BASE AND THE DATA BASE

Data base systems typically have little knowledge, much data, and rely on fast secondary storage techniques. Knowledge-based systems, on the other hand, have much knowledge, little data, and work within main memory. If knowledge-based systems are to be integrated into the design process, new methods must be developed so that they can handle the large amounts of data typically created during a design project.

There are three approaches (figure 4) in current long-term research efforts which are trying to determine the best way to couple these two types of systems into a single system with the best features of both (refs. 10,11,12). One approach is to begin with an existing data base system and add knowledge base features to it. A second approach is to begin with an existing knowledge-based system and add data base features to it. The third approach, and probably the most promising, is to start from scratch and develop a completely new system combining the best features of both data and knowledge-based systems. The best short-term solution appears to be taking an existing knowledge-based system and an existing data base system and loosely coupling them with an interface such as the one used by Feyock (ref. 13). This is the approach taken with STRUTEX using CLIPS.

The rules for STRUTEX are very simple and could probably have just as easily been incorporated into the main driver program with IF-THEN statements. However, one of the objectives of this project was to investigate methods for passing data between a data base and a knowledge base. Therefore the knowledge base and the data base are maintained in different files separate from the main program. The interface is made through STRUTEX by linking a RIM interface library and the CLIPS knowledge base interface library with STRUTEX. If data is needed from the knowledge base, a rule or rules must be executed. If the data is not available within STRUTEX, it is retrieved from the data base by calling RIM interface subroutines. That data is then asserted into the knowledge base again using interface subroutines. Data is returned from the knowledge base to STRUTEX which, in turn, stores the data in the RIM data base if it is necessary.

STRUTEX has three subroutines which interface with CLIPS by calling subroutines in the CLIPS FORTRAN interface library. Subroutine KBXEC initializes CLIPS, loads the rule base, and is called by other subroutines to assert facts into the knowledge base and execute the inference engine. Once all of the rules on the CLIPS agenda have been executed, control is returned to this subroutine which allows STRUTEX to continue processing. The other two routines, KBANS1 and KBANS2, receive data from CLIPS after the appropriate rule (or rules) has been executed. These two subroutines are called as an action in CLIPS. KBANS1 has three parameters, two alphanumeric and one numeric, while KBANS2 has three numeric parameters. The data returned from CLIPS are stored in these parameters for later use in other subroutines in STRUTEX.

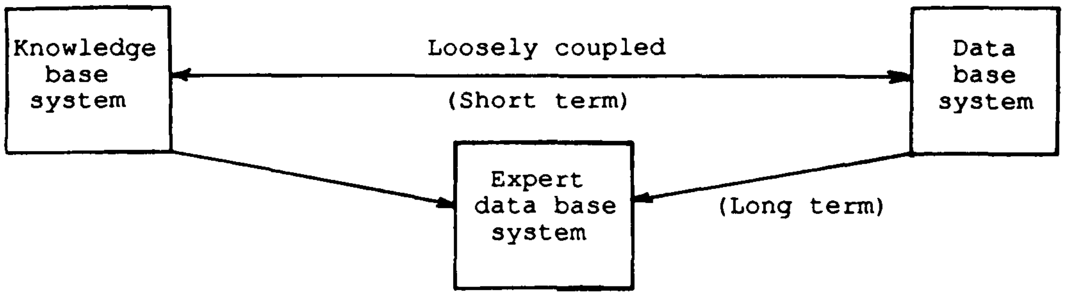

Figure 4

USER INTERACTION WITH STRUTEX

When STRUTEX begins execution, the user first answers questions about the loads. The number of load points is the first input. The next input is the type of load which can be a gravity load, vertical load, sideways load, or a combination of gravity or vertical and sideways. This is followed by an iterative process through the load points where the user inputs the vertical and horizontal magnitudes of each load and then uses the mouse to locate the load point on the graphics window. Since no units are required by STRUTEX, the user must determine the correct units for the distances and loads.

The second stage of user input concerns the support surface. The user inputs where the support surface is in relation to the load points - above, below,or to the side. The user then uses the mouse to place the midpoint of the support surface on the graph. The distance from the support surface to the first load is input without units. The distances from the remaining loads to the support surface are computed. The user inputs whether or not the support surface is a point. If it is not, then the length of the support surface is input, again with no units.

The final piece of data that is needed before the system can determine the type of support is whether it is important for the support to be inexpensive or lightweight. Once these data are known, facts are asserted into the knowledge base and the inference engine executes the rules. The type of support is returned from the rules and stored into the data base. The choice and explanation of that choice are displayed on the dialog screen as shown in figure 5.

```
An explanation from STRUTEX

*************************************

 A truss is the choice for a support.

*************************************

 Reasons:
  The support surface location is to
       the side of the loads.
  The support surface is not a point.
  The support must be lightweight.
```

Figure 5

A TRUSS WITH A SINGLE LOAD POINT

If the choice is a truss and there is only one load point, a triangular structure is drawn (figure 6a). The system then determines whether or not bracing is needed by checking the ratio of the forces in the members against the length of the members. The forces are computed from an equation representing static equilibrium of the loaded point. Facts are asserted into the knowledge base. The inference engine executes the rules and the choice is returned to the main program. If bracing is needed, the two side members are divided, and either a "Z" brace (figure 6b) if Delta is greater than 40 degrees or a "V" brace (figure 6c) if Delta is less than or equal to 40 degrees is chosen by the knowledge base. An input file is created for the analysis program and the program ends.

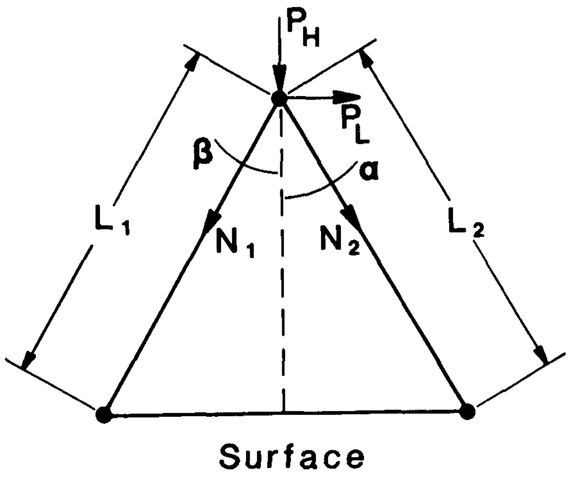

Figure 6a

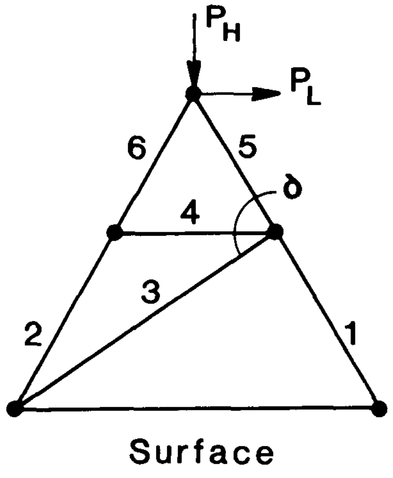

Figure 6b

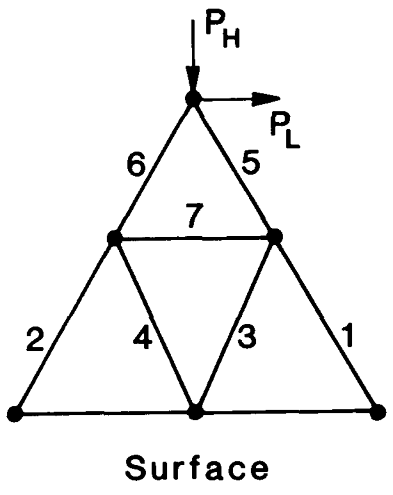

Figure 6c

A TRUSS WITH MULTIPLE LOAD POINTS

If the choice is a truss and there is more than one load point, then the user must build the truss guided by recommendations from the system. The user begins with the load points and the support surface. A recommendation is made to first connect all the load points forming triangles whenever possible, but not connect the load points to the support surface. Members are added by placing the mouse on the end points. Once this has been completed, there is a second recommendation for the user to connect the load points to the support surface without having a new member intersect an existing member. (The reason for using two recommendations to build the truss is discussed below.) Figure 7 displays the structure at this point in the design.

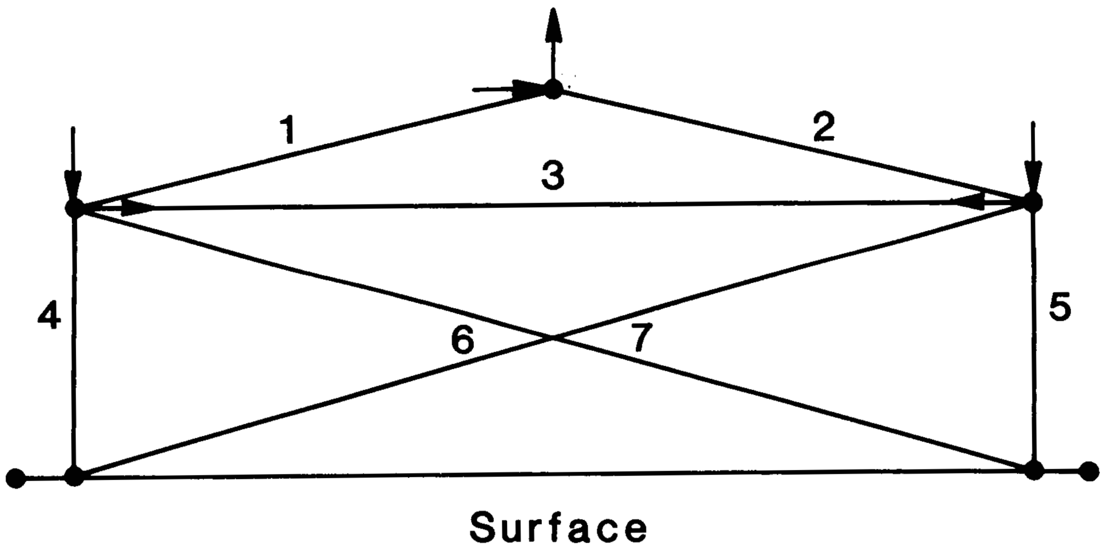

Figure 7

A RECOMMENDATION FOR IMPROVING THE MODEL BASED ON TRIANGLES

After this step is complete, the system determines all the triangles formed by the members and checks their angles. If the knowledge base determines that there are angles in the model outside a given range, a recommendation is made to correct the problem. The limiting values for the angles are judgmental and can be changed based on the experience of the user. An example of such a recommendation based on the angles is shown in figure 8.

```
A recommendation from STRUTEX

        ******RECOMMENDATIONS******

        The following triangles contain angles that
        are less than 15 degrees, therefore
        a modification may be required.

        TRIANGLE        ANGLE        OPPOSITE MEMBER
         1 2 3           13.7              1
         1 2 3           12.4              2

                      N1
                      /\
                     /  \
                  N2/    \N3

        If two external members form the angle
        then to expand the angle
           (1) Remove the two members N1-N2 and N1-N3
               that form the angle.
           (2) Add two new members N1-N4 and N1-N5
               to form a larger angle.
           (3) Add a new member to connect N4 and N5.
           (4) Add two members to connect
               N2-N4 and N3-N5.

                      N1
                      /\
                     /  \
                  N4/____\N5
                     |    |
                  N2 |    | N3

        If this recommendation can be implemented in
        more than one way, choose the way that will
        contract the structure rather than expand it.

        *********************************************
        The following triangles contain angles that
        are greater than 120 degrees; therefore,
        a modification may be required, such as
        adding a new member to divide the angle
        into two smaller angles.

        TRIANGLE        ANGLE        OPPOSITE MEMBER
         1 2 3          154.               3
        *********************************************
```

Figure 8

IMPROVED MODEL

The user then removes all members which contribute to the problem and adds new members to satisfy the recommendation. If the user desires, the angles in the model can again be checked for problems. This is repeated until the user is satisfied (figure 9). The input file for the structural analysis program is written and the program ends.

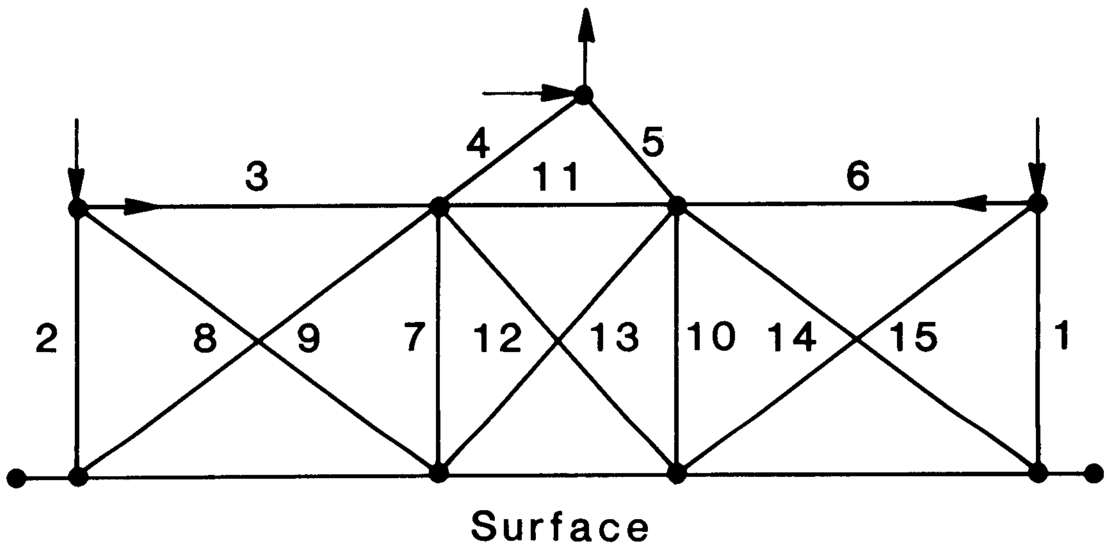

Figure 9

MODIFYING THE BRACING OF A TRUSS

For the truss with multiple load points, there are two recommendations instead of one to allow the knowledge base to determine the bracing required between two members connecting a load point to the support surface. When two members connecting a load point to the support surface form a quadrilateral, the knowledge base is given the lengths of the two members. If the two lengths are the same, an "X" bracing is added. If the two lengths are different, a brace is added from the bottom of the longer member to the top of the shorter member (figure 10).

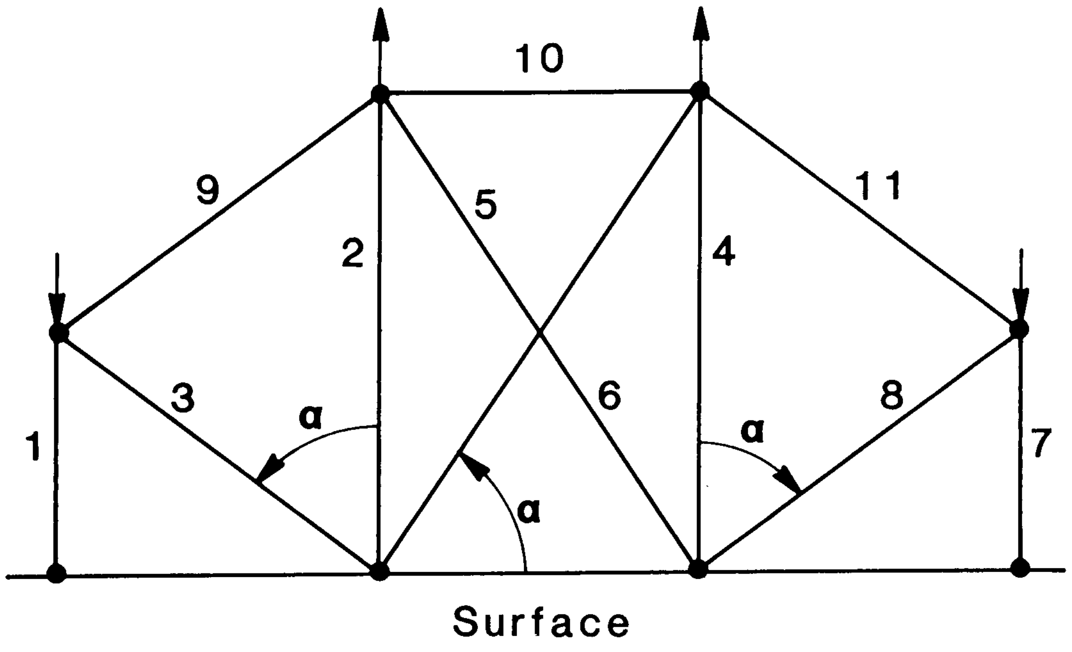

Figure 10

A RECOMMENDATION FOR IMPROVING THE BRACING

The angle, Alpha, made by the two members is passed to the knowledge base, and a recommendation is made if the angle is not within the proper range. An example of such a recommendation is shown in figure 11.

A recommendation from STRUTEX

```
******RECOMMENDATIONS******

Because the angles made by the diagonals
and the support surface are greater than 75 degrees
(Angle = 76.), it is recommended that
members 5 and 6 be removed and that
members 2 and 4 be divided in two and
reconnected with an X bracing.

Because the angle made by the diagonal
and the support surface is greater than 75 degrees
(Angle = 77.5), it is recommended that
member 3 be removed and that member 2
be divided in two and connected to the ends
of member 1.

Because the angle made by the diagonal
and the support surface is greater than 75 degrees
(Angle = 77.5), it is recommended that
member 8 be removed and that member 4
be divided in two and connected to the ends
of member 7.

*************************************************
```

Figure 11

A TRUSS WITH IMPROVED BRACING

The truss in figure 12 reflects the refinements made from this recommendation. It is possible, especially in a very complex truss, that this recommendation might come out in addition to the recommendation about the triangles.

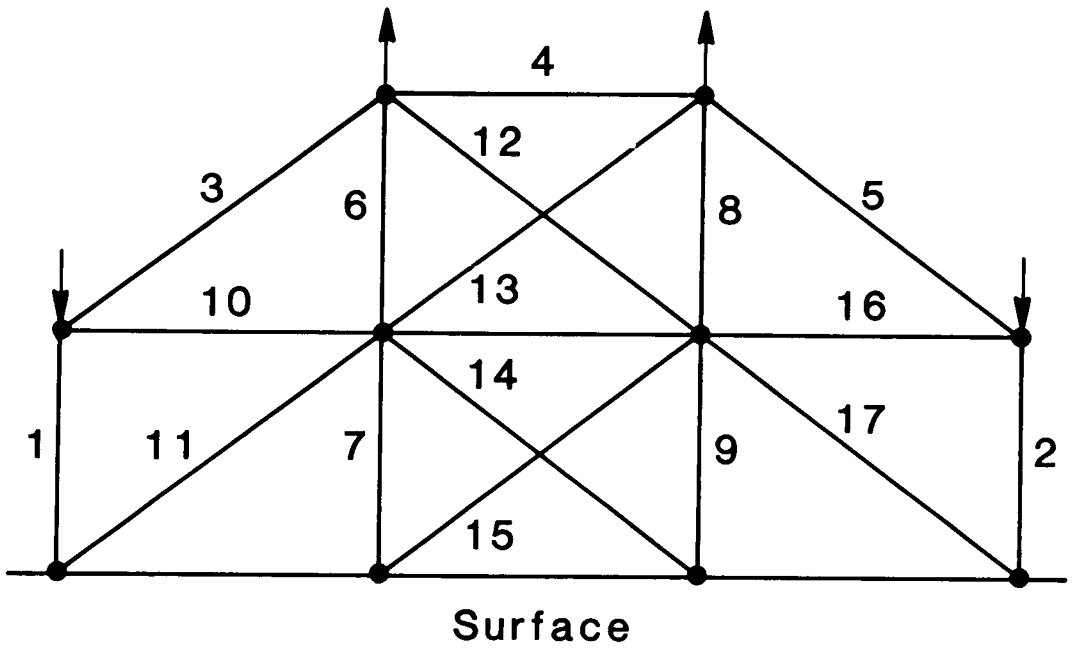

Figure 12

CONCLUSIONS

A prototype knowledge-based system has been developed to initially configure a structure to support point loads in two dimensions. There were two primary objectives for this project. The first objective was to investigate methods for passing data between a data base and a knowledge base. This was accomplished by integrating an inference engine into the system and determining the effects on the flow of data between the knowledge base and the data base. No significant problems were encountered in integrating the inference engine.

The second objective was to determine if an initial conceptual model could be improved by using symbolic processing instead of numeric processing. By applying the knowledge base, significant improvements were made to the trusses shown in the examples.. If more rules were added to the knowledge base then more improvements could be made to the model.

REFERENCES

1. MacCallum, K. J.; Duffy, A.; and Green, S.: "An Intelligent Concept Design Assistant." Proceedings of the IFIP W.G.5.2 Working Conference on Design Theory in CAD, 1985, pp.233-249.

2. Shah, J.: "Development of a Knowledge Base for an Expert System for the Design of Structural Parts." Proceedings of the 1985 29thComputers in Engineering Conference, 1985, Vol. 2, pp. 131-136.

3. Rogers, J., Feyock, S. and Sobieszczanski-Sobieski, J.: STRUTEX A Prototype Knowledge Based System for Initially Configuring a Structure to Support Point Loads in Two Dimensions. "Artificial Intelligence in Engineering: Design", Computational Mechanics Publications, Southampton, PP. 315-335.

4. DI-3000 User's Guide, Precision Visuals Inc. Document Number DI3817, Release Number 4, March 1984.

5. Erickson, W. J.: "User Guide: Relational Information Management (RIM)", Report Number D6-IPAD-70023-M, Boeing Commercial Airplane Company, Seattle Washington, 1981.

6. Riley, G.; Culbert, C.; and Savely, R. T.: "CLIPS: An Expert System Tool for Delivery and Training." Proceedings of the Third Conference on AI for Space Applications, November 1987.

7. Whetstone, W. D.: "EISI-EAL: Engineering Analysis Language", Proceedings of the Second Conference on Computing in Civil Engineering, ASCE, 1980, PP. 276-285.

8. Vanderplaats, G. N.: CONMIN _ A FORTRAN Program for Constrained Function Minimization. User's Manual. NASA TM X-62282, 1973.

9. Rogers, J. L., Jr.; Sobieski-Sobieszczanski, J.; Bhat, R. "An Implementation of the Programming Structural Synthesis System (PROSSS)", NASA TM 83180, December 1981.

10. Ceri, S.; Gottlob, G.; and Wiederhold, G.: "Interfacing Relational Databases and Prolog Efficiently." Proceedings of the First International Conference on Expert Database Systems, April 1-4, 1986, pp.141-153.

11. Nguyen,G.T.: "Prototypes and Database Samples for Expert Database Systems." Proceedings of the First International Conference on Expert Database Systems, April 1-4, 1986, pp.3-14.

12. van Biema, M.; Miranker, D. P.; and Stolfo, S.J.:" The Do-Loop Considered Harmful in Production System Programming." Proceedings of the First International Conference on Expert Database Systems, April 1-4, 1986, pp. 125-138.

13. Feyock, S.; and Rogers, J. L.: "Embedded AI for Structural Optimization". Presented at the International Conference on Computational Engineering Science, April 10-14, 1988, Atlanta, GA.