CIFE CENTER FOR INTEGRATED FACILITY ENGINEERING

QStruc: An Approach for Qualitative Structural Analysis

by

Renate Fruchter, Kincho H. Law and Yumi Iwasaki

TECHNICAL REPORT
Number 63

January, 1992

Stanford University

CIFE Report December 1991

QStruc
AN APPROACH FOR QUALITATIVE STRUCTURAL ANALYSIS

by

Renate Fruchter*, Kincho H. Law and Yumi Iwasaki*****

* Center for Integrated Facility Engineering, Department of Civil Engineering,
Stanford University, Stanford, CA 94305

** Department of Civil Engineering, Stanford University, Stanford, CA 94305

*** Knowledge System Laboratory, Department of Computer Science,
Stanford University, Stanford, CA 94305

If you would like to contact the authors please write to:

c/o CIFE, Civil Engineering,
Stanford University,
Terman Engineering Center
Mail Code: 4020
Stanford, CA 95305-4020

Abstract

Understanding the behavior of a structure and its components in the preliminary design phase can have a significant effect on the quality of the final design. Accurate prediction of the behavior of a structure can reduce the number of alternative solutions and avoid costly design revisions. However, there are few tools available for modeling and qualitatively analyzing a structure.

This report describes a system, **QStruc,** for *qualitative structural analysis* which combines first principles in structural engineering and experiential knowledge of structural behavior. The purposes of QStruc are (1) to generate qualitative models from the schematics of a structure, and (2) to infer the qualitative response of the structure in terms of deflected shape, moments, and reactions. The framework consists of two main modules: the *model generation* module and the *qualitative analysis* module. The proposed qualitative analysis strategy is a "greedy," depth-first approach that tries to expand the derived response as much as possible from known parameter values. The system makes use of:

- Causal ordering mechanism, that enables the system to identify the solution path for the qualitative analysis. A causal model is used to describe the behavior of a structure in terms of physical quantities and causal interactions among them.

- Qualitative calculus, which enables the qualitative evaluation of the physical quantities of the causal model that describes the behavior of the structure.

- Quantity Lattice [7], which enables the system to reason about partial ordering among physical quantities and to reduce some of the ambiguous conclusions caused by the impreciseness of the information.

The report discusses the following aspects of QStruc: (1) *representation* of the physical model, fundamental principles, and experiential knowledge, (2) *model generation,* and (3) *qualitative analysis.* We provide a simple example to illustrate the operations of the prototype system QStruc. The report concludes by summarizing our preliminary findings and suggesting directions for future research.

ACKNOWLEDGEMENTS

This research has been the result of the collaboration of Prof. Kincho H. Law, Dr. Yumi Iwasaki and visiting post-doctoral research fellow Renate Fruchter.

Dr. Renate Fruchter would like to take this opportunity to express her gratitude to a number of people. She is grateful to Prof. Kincho H. Law for proposing this research project and inviting her to have the precious experience of research work at Stanford University. Dr. Fruchter would like to thank him for his guidance, support and patience in discussing and critiquing new ideas related to the definition of an approach for qualitative structural analysis.

Dr. Fruchter is most grateful to Dr. Yumi Iwasaki for the guidance and support throughout the project, for introducing her to the advanced research area of qualitative physics, for patiently trying to understand the structural engineering problems and the proposed approach, and for providing valuable critique of the initial ideas of this work.

Dr. Fruchter is grateful to Dr. John Kunz for the interesting discussions, useful advice and encouragement in exploring this research direction in the initial stages.

Dr. Fruchter would like to thank Mr. Pandu Nayak (Ph.D. student at the Knowledge System Laboratory) for the various interesting discussions we had.

Last, but not least, Dr. Fruchter would like to thank Mr. Jim Rice (of the Knowledge System Laboratory) for the indispensable and ever-responsive aid in understanding the hardware problems and learning the debugging tools.

The research was partially sponsored by the Center for Integrated Facilities Engineering and was conducted at the Knowledge System Laboratory, Stanford University .

CONTENTS

1. BACKGROUND AND MOTIVATION

Determining the behavior of a structure and its components in the early design stages is important because decisions made at this time strongly influence the quality of the final design. The traditional approach to the analysis of a structure has been mostly based on numerical methods. To apply a numerical method, a structural designer must supply numerical values for the loads and dimensions of the structure and its components; the method then determines the numerical results in terms of reactions, moments, deflections, etc. However, in the preliminary design stage the analyzed structure is conceived as a rough, qualitative sketch and there is not sufficient data to warrant the use of numerical analysis tools. Nevertheless, experienced engineers are able to reason about structural performance during the preliminary design stage (for example, by approximating bending moments, and deflected shapes) based on physical principles and experiential knowledge. They then use these results to guide the preliminary selection of structural components. Such qualitative analysis is an important task that can potentially reduce the number of alternative solutions and redesign efforts. One difficulty in formalizing a qualitative analysis strategy is that there is no a priori known sequence of steps in the qualitative approach. The most effective solution may emerge from different information types for different problems. Thus, a flexible qualitative approach is needed to adjust the solution strategy as more information is obtained.

This report proposes a system for qualitative structural analysis, **QStruc**, based on fundamental principles of structural behavior and experiential knowledge. Given a specific structure in terms of the components, the relation among them, and a loading scenario, the system infers a sequence of models and qualitatively determines the response of the structure. A flexible "greedy" reasoning strategy is developed to determine the structural response from the known parameter values efficiently. This report presents the conceptual ideas, methodology of QStruc, and the results of our preliminary investigation.

In the last decade, the topic of qualitative reasoning about physical systems has received much attention in artificial intelligence research [Refs. 19, 20]. Qualitative reasoning attempts to draw useful conclusions about the behavior of a physical system using mostly qualitative knowledge about its structure and the governing physical principles. Appendix D summarizes the basic concepts and discusses the main approaches proposed in the domain of qualitative physics. The mathematical tool most often used in qualitative reasoning is qualitative calculus, in which functional relations among the variables are expressed in terms of qualitative variables and operators. Qualitative variables are variables whose values are one of a small number of predefined intervals, for example positive (+), zero (0), or negative(-). Qualitative operators such as addition, subtraction, and multiplication can be defined over such qualitative values. Appendix A shows examples of such operators over qualitative values (positive, zero, negative), which we will denote as *Q+*, *Q0*, and *Q-*. Qualitative calculus enables the reasoning process to draw interesting conclusions from imprecise information about quantities and their relations. On the other hand, because of the impreciseness of the information used, qualitative reasoning can give rise to ambiguous conclusions, that need to be resolved by using additional information.

The use of qualitative reasoning to structural engineering problems has been investigated by many researchers [1, 7, 9, 14, 15].

- Slater's [Ref. 15] work is one of the first attempts to apply a rule based approach to the prediction of deflections and moments in indeterminate beam structures. His conclusion points to the need of a mechanism for a better "understanding" of behavior that will not rely so heavily on the domain-specific carryover factor approach.
- QSEIS [Ref. 9] is a recent work that addresses the problem of deriving the qualitative behavior of a structure under earthquake loads. The knowledge base of this system consists of heuristic knowledge and knowledge compiled from physical principles. The qualitative behavior is derived using causal relationships among entities. The main drawback of such an approach is the need to exhaustively envision all the possible causal relationships among the entities describing a domain.
- Fruchter [Ref. 7] proposes to use first principles and heuristic knowledge in guiding the structural analysis and focusing on critical behavior regions. The described approach is useful when the structure is described quantitatively, but does not address the case of qualitatively described structures.
- 1stPRINCE [Ref. 1] is a system that addresses the special problem of non-routine design. It develops innovate structural designs by reasoning from first principle knowledge to discover new design prototypes. The system uses the qualitative technique of monotonicity analysis as derived from Karush-Kuhn-Tucker conditions of optimality for selecting a critical integral. The design space is then expanded by a method called the "dimensional variable expansion," which essentially divides the integral into a set of smaller integrals.
- CRACK [Ref. 14] represents an attempt to link heuristic and quantitative reasoning through qualitative reasoning in the domain of fatigue and fracture in steel highway bridges. The approach of qualitative simulation as implemented in QSIM [Ref. 13] was selected as the representation scheme for the bridge cracking problem. The results of this work show that qualitative reasoning can be a useful tool for guiding quantitative analysis. However, the results also identify current limitations of the qualitative reasoning technology. The qualitative level was found to be difficult to control and to fully utilize. The main difficulties that arose in this work, by using QSIM, are that (1) in a QSIM model landmarks within a given quantity space cannot be influenced by other quantities; (2) testing to see whether two quantities are equal requires developing fictitious relations, which can lead to undesired behaviors; (3) the value of a quantity cannot be fixed throughout a simulation by statements in the initial conditions; and (4) spatial reasoning, extending the model from one dimension to two and three dimensions, needs to be addressed by future research.

Although these studies have achieved some preliminary successes in developing methodologies for using qualitative reasoning in the recent years [Refs. 1, 7, 9, 14, 15], they do not address the need to develop an approach that can model the flexible reasoning strategies that a human designer uses in the initial design stage.

The motivation for the present work is the desire to provide a useful tool for the complex task of structural modeling, analysis, and interpretation at an early stage of the design process, one which can be integrated into a larger CAE framework. Our research has lead us to conclude that the two major phases of the design process - conceptual design and detailed design - require different kinds of tools.

- In detailed design, even though the analysis tools do not play an active role in the process of synthesis, they have an important role in verifying the synthesized design and are used to evaluate the design decisions. Efforts in developing CAD tools have primarily focused on automating the complex numeric analysis tasks used in this design phase.
- In conceptual stage, on the other hand, designs are incomplete and imprecise to warrant the use of traditional numeric analysis tools. In this phase, qualitative reasoning tools should play the role that numeric analysis tools play in the detailed design. In a larger CAE framework, such tools would provide assistance in qualitative evaluation, verification, and interpretation of preliminary alternative designs.

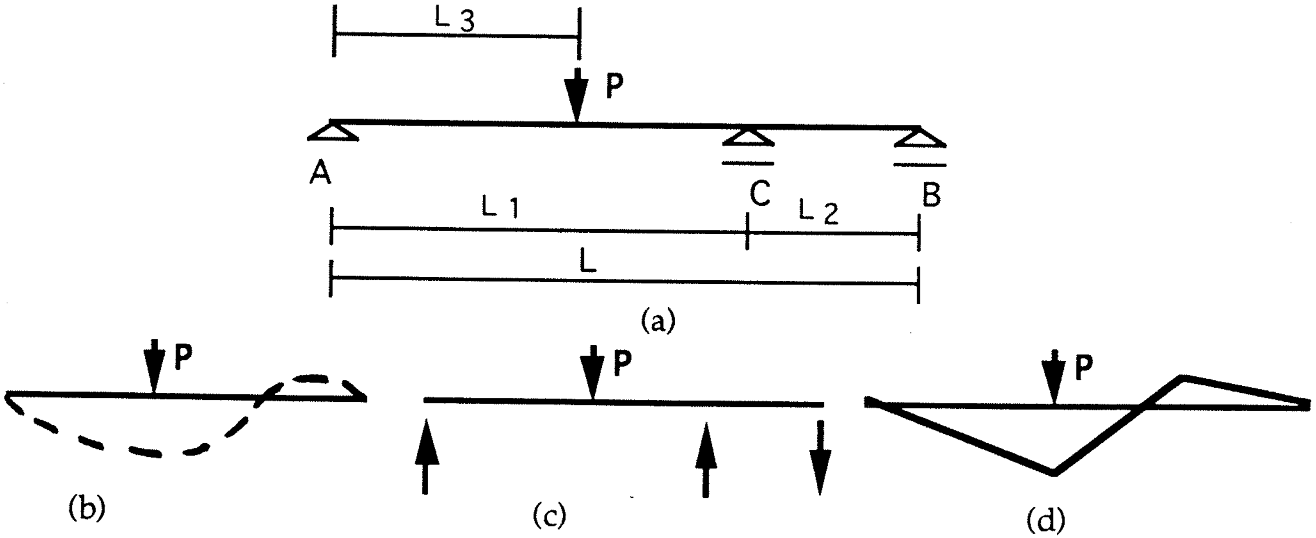
Conventional knowledge-based systems (KBSs), which rely on task-specific heuristic knowledge, tend to fail ungracefully when confronted with a problem even slightly outside their narrow domain of expertise. One reason for such "brittleness" is their lack of fundamental knowledge of the domain to fall back on when the heuristic knowledge fails. This lack of fundamental knowledge also results in the poor quality of explanations that KBS systems can provide. *Model-based reasoning* attempts to overcome these problems of conventional KBSs by providing the system an explicit domain model, consisting of fundamental principles of the domain and an ability to reason from such knowledge. In the work this report describes, we use explicit models and first principles, as well as heuristic knowledge, to analyze the behavior of a structure. Some of the heuristic knowledge has been demonstrated to be useful in guiding structural analysis [3] and in focusing on relevant problems for a given model.

Model-based reasoning about the possible behavior of a structure has three main stages: *construction of an appropriate model* for reasoning about the particular type of behavior in question, *analysis* of the model, and *interpretation* of the results. Many powerful CAE tools exist for analyzing a model numerically when the model has enough quantitative details to render such analysis possible. In contrast, few tools are available to help with model construction and interpretation. Likewise, there are no analysis tools that can be used when there is insufficient quantitative information. This work describes a framework to assist the analyst in all three stages (model construction, analysis, and interpretation) at the initial design phase when only incomplete and qualitative information about the design is available.

This technical report is organized as follows: Section 2 presents an overview of the QStruc system. Section 3 describes the representation of the domain. Section 4 describes our method for generating qualitative models. Section 5 proposes a greedy algorithm for qualitative analysis and describes the techniques that we employ for qualitative reasoning. Section 6 presents the results of the qualitative analysis for the continuous beam example which has been used throughout the discussion. To further demonstrate the capability of QStruc we present two 2-D frame structure examples. Finally, Section 7 summarizes the preliminary findings of this research and discusses future work. Appendix A gives examples of qualitative operators. Appendix B shows an example of using Quantity Lattice to reduce the ambiguity in a qualitative calculus problem. Appendix C details the reasoning steps and the results of a sample run of QStruc. Appendix D presents the basic principles of and related work in qualitative physics.

2 SYSTEM OVERVIEW

The kinds of information that a structural engineer often uses in the conceptual design phase include the qualitative values of internal forces caused by external loads, the qualitative moment diagrams, and the deflected shape diagrams. Figure 1 illustrates an example of the class of problems that QStruc addresses. Figure 1a shows a continuous beam consisting of components, connectivity, and supports. It also gives the qualitative information about the direction of a concentrated load. As shown in Figure 1b, in reasoning about the behavior of the structure, the engineer infers that: (1) the left span of the beam will bend downwards, (2) there will be an inflection point near the support along the beam, and (3) the right span will bend upward. The reactions and moments at the supports will be as indicated in Figure 1c and 1d. Deriving such a description of behavior involves reasoning about the structure at various levels of detail (e.g., considering the support reactions, focusing on the different spans and sections of the beam). Furthermore, the reasoning strategy involves deciding whether to start the qualitative analysis by deriving the moments, reactions, or deflected shape, and how to proceed from the derived information.



(a)

(b)

(c)

(d)

Figure 1: A Continuous Beam Example

Hence, a qualitative reasoning tool for structural analysis, such as QStruc, must be able to

(1) generate models necessary for deriving qualitative deflected shape tendencies, reactions, and moments,
(2) perform qualitative analysis, given an incomplete description of a structure.

The top-level conceptual structure of the **QStruc** system is shown in Figure 2. The input to the system is a schematic description of the structure, as a designer would first sketch it, in terms of structural objects (e.g., topology, supports, information about loads) and their relations (e.g., connectivity). The output is the qualitative response of the given structure, in terms of deflected shape, moments, and reactions. The domain knowledge is represented by a knowledge base, containing physical laws (first principles), and experiential knowledge about known relations among physical quantities, as well as the knowledge necessary for focusing on relevant regions with respect to the structural behavior. QStruc has two main modules: the *model generation* and the *qualitative analysis* module. The purpose of the model generation module is to take the input description of the structure and to transform it into refined models. The purpose of the qualitative analysis module is to identify a solution path and to evaluate the qualitative response of the structure. The following sections discuss the various modules of the system. QStruc has been implemented using an object-oriented knowledge engineering tool, KEE[1].

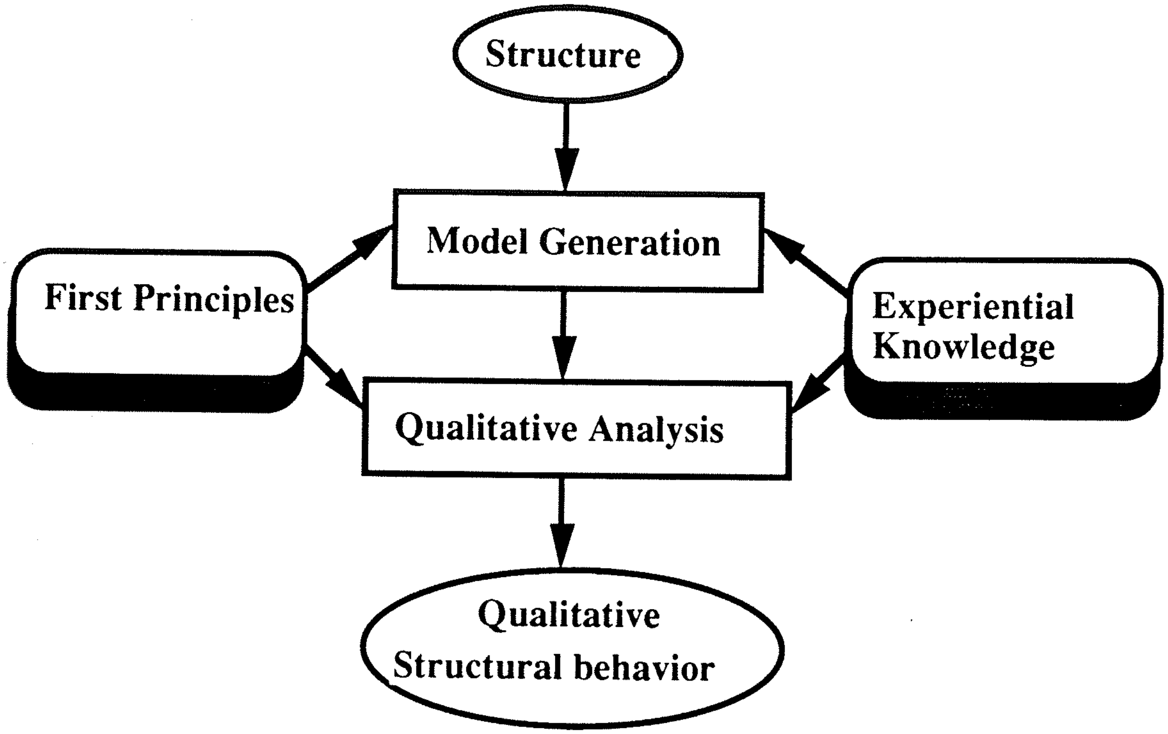


Figure 2: System Overview of QStruc

---

[1] KEE is a trademark of Intellicorp Inc.

3 REPRESENTATION OF STRUCTURAL MODEL AND KNOWLEDGE

3.1 Representation of structural model

Structural models are represented by instances of the following classes:

- *Components:* This class has two subclasses - *beam* and *column*. Components can be combined to create more complex structures or decomposed into subcomponents to highlight a relevant region from the behavior perspective [Ref. 7].
- *Terminals:* This class contains the information about the end sections of a component, such as geometrical (e.g., *location_x*) and behavioral (e.g., moment, shear force, displacement) information.
- *Connections:* This class provides the means to relate two terminals of adjacent objects. It also represents a basic unit that facilitates the transfer of information (e.g., forces, deflections) from one object to another. It has two subclasses, *continuity-connection* and *interior-hinge-connection*.
- *Nodes:* This class is used to connect any number of terminals of any adjacent type of connection to that node. It serves as a means to transfer or redistribute information among the connected objects.
- *Supports:* This class has a set of subclasses including *fixed_support*, *pinned_support*, *free_end*, and *roller_support*. They contain information about the boundary conditions and possible resulting reactions.
- *Loads:* This class has as subclasses load types, such as *concentrated_load* and *uniformly_distributed_load*.

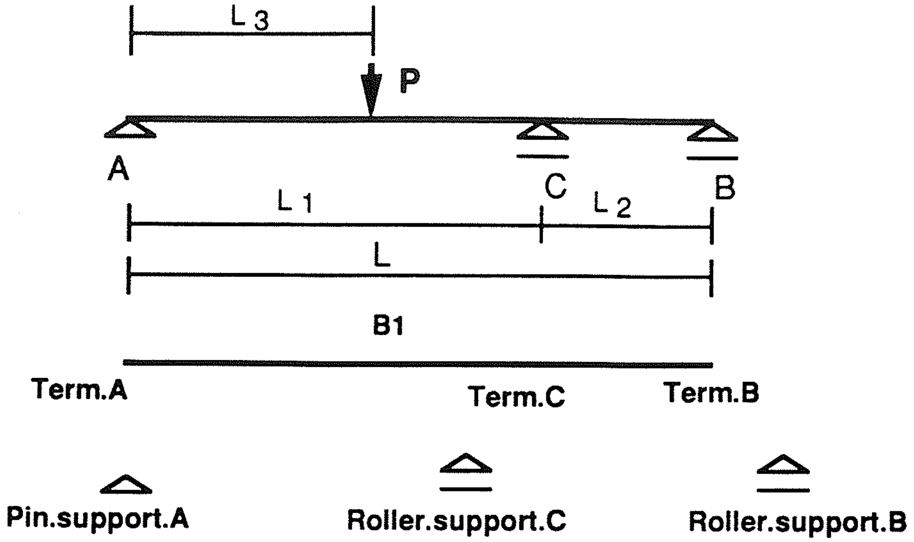
These structural objects contain slots for the following types of information:

- *Geometrical information*, which includes the location for the end sections of a component, the application point of a load, and the location of the point with respect to which the qualitative equation of moment equilibrium is derived.
- *Contextual information* , which describes the loading situation.
- *Behavioral information*, which includes existing boundary conditions (e.g., rotation, displacement) in the case of a connection or support object, possible existence of reactions or internal forces, values derived from the qualitative analysis, information about processes (e.g., bending, compression), laws of information processing and transfer (e.g., behavior equations such as laws of equilibrium), and a list of active forces.

It may be worth noting that for this representation scheme, if numerical data is available for all the objects, traditional numerical analysis tools can be used to derive the structural response.

An example of the representation of a continuous beam is shown in Figure 3. In the example, *B1* is an instance of *beam*. *Term.A*, *Term.B*, etc. are instances of *terminals.*

Pin.support.A is an instance of the subclass *pin_support,* etc. *P* is an instance of *concentrated_load.*



Beam.1
Location_i: ***Term.A***
Location_j: ***Term.B***
Load: P
Possible_bending_processes: unknown
.
.
Possible_rotation_at_i: unknown
.
.
moment_equilibrium_equation: unknown
.
.
.

Pin.support.A
Location: ***Term.A***
Supports: ***B1***
N_reaction: ***C+***
S_reaction: ***C+***
M_reaction: ***C-***
Q_N_reaction: unknown
.
.
.
Allows_x_displacement: ***C-***
Allows_y_displacement: ***C-***
Allows_rotation: ***C+***
Q_x_displacement: unknown
.

Term.A
Internal_force_N: unknown
Internal_force_S: unknown
Internal_force_M: unknown
Location_x: 1
Location_y: 0
Point_location: unknown
Q_x_displacement: unknown
Q_y_displacement: unknown
Q_rotation: unknown
.
.
.

P
Load_direction: ***Q-***
Load_magnitude_qlattice: unknown
Load_location_x: 3
Load_location_y: 0
Point_location: unknown
Force_moment_direction: unknown
.
.
.
Component_on _which_the_load_acts: ***B1***
.
.
.

Figure 3: Example of the Representation of a Continuous Beam

Each parameter (e.g., moment, force, rotation) associated with an object is represented by a slot whose value is an instance of the class *qualitative_values*. *Qualitative_values* has instances, *Q-*, *Q0*, and *Q+*, representing a negative value, a zero, and a positive value respectively. These three qualitative values are used uniformly as values for all variable types, but are interpreted differently depending on the parameter type. For example, the value *Q+* of a force variable means the direction of the force is upward, while it means clockwise rotation for a moment variable. Also in the class of *qualitative_values* are two qualitative identifiers, C+ and C-, indicating respectively the existence and non-existence of a constraint.

3.2 Representation of domain knowledge

QStruc uses fundamental domain knowledge as well as heuristic knowledge. Knowledge is represented explicitly as object classes:

- *Equations and relations:* This class has subclasses, *equations* and *relations*. *Equations* represent the various physical laws (e.g., *Moment-Equil-Eq*), and *relations* are experiential knowledge (e.g., *Load-Direction-Displacement-Direction relation*). Instances of these subclasses contain pointers to the structural component to which they belong and to the physical quantities (parameters) involved.

- *Physical.Quantities*: This class has subclasses representing the different possible parameters (e.g., *Vertical-Force, Displacement*). Instances of these subclasses have slots such as: *causal-dependents, causal-dependent-on, causal-exogenous*. They also have pointers to the structural objects to which they belong and to equation(s) and/or relation(s) that they are involved in.

A method that represents a specific first principle is associated with structural objects. For example, when the moment equilibrium slot of a component object receives a message from a behavior process (e.g., *bending*) this will cause the *Moment-Equil-Eq* to be generated for the specific component together with the corresponding instances of the involved physical quantities.

Other physical principles, such as the definition of a moment generated by a force with respect to a location, are represented as "If-Then" rules, which can be activated in a forward or backward chaining manner. Heuristic knowledge about model generation is represented as methods for transforming and updating the qualitative information of the structure. Knowledge about experiential relations among parameters is encoded as rules.

4 MODEL GENERATION

The objective of the **model generation** module is to transform a given structure, which we will call *schematic model*, into several models, each of which is used to analyze a particular behavioral aspect of the structure. Since our goal is to emulate the reasoning strategy of a human designer, we have identified which relevant types of models are used in reasoning about structural behavior, and what types of knowledge are necessary for generating such models. Figure 4 illustrates the models that are prior to qualitative analysis and the sequence of transformation. These models are:

- A *behavior model*, that consists of an explicit representation of the existing interactions among the structural objects and a definitional behavior pattern. The knowledge used in defining this model consists of first principles regarding the definition of supports and connections in terms of existent or non-existent constraints (e.g., a hinge connection does not enable moment transfer) and experiential knowledge regarding possible behavioral patterns of structures (e.g., simple supported beam).

- A *specific model*, that contains additional information about regions of interest regarding the structural behavior, that are identified by applying experiential focusing knowledge (e.g., if there is a concentrated load on a component, then focus on the region where this force is applied by decomposing the component and updating the representation of the structure).

- A *process model* that consists of qualitative equations and relation obtained by identifying the applicable first principles and experiential knowledge.

In the following, we describe each transformation step using the continuous beam example of Figure 3 and illustrate graphically the results generated by the system QStruc implemented in KEE. For the purpose of clarity the figures were manually drawn based on the symbolic results of the system.

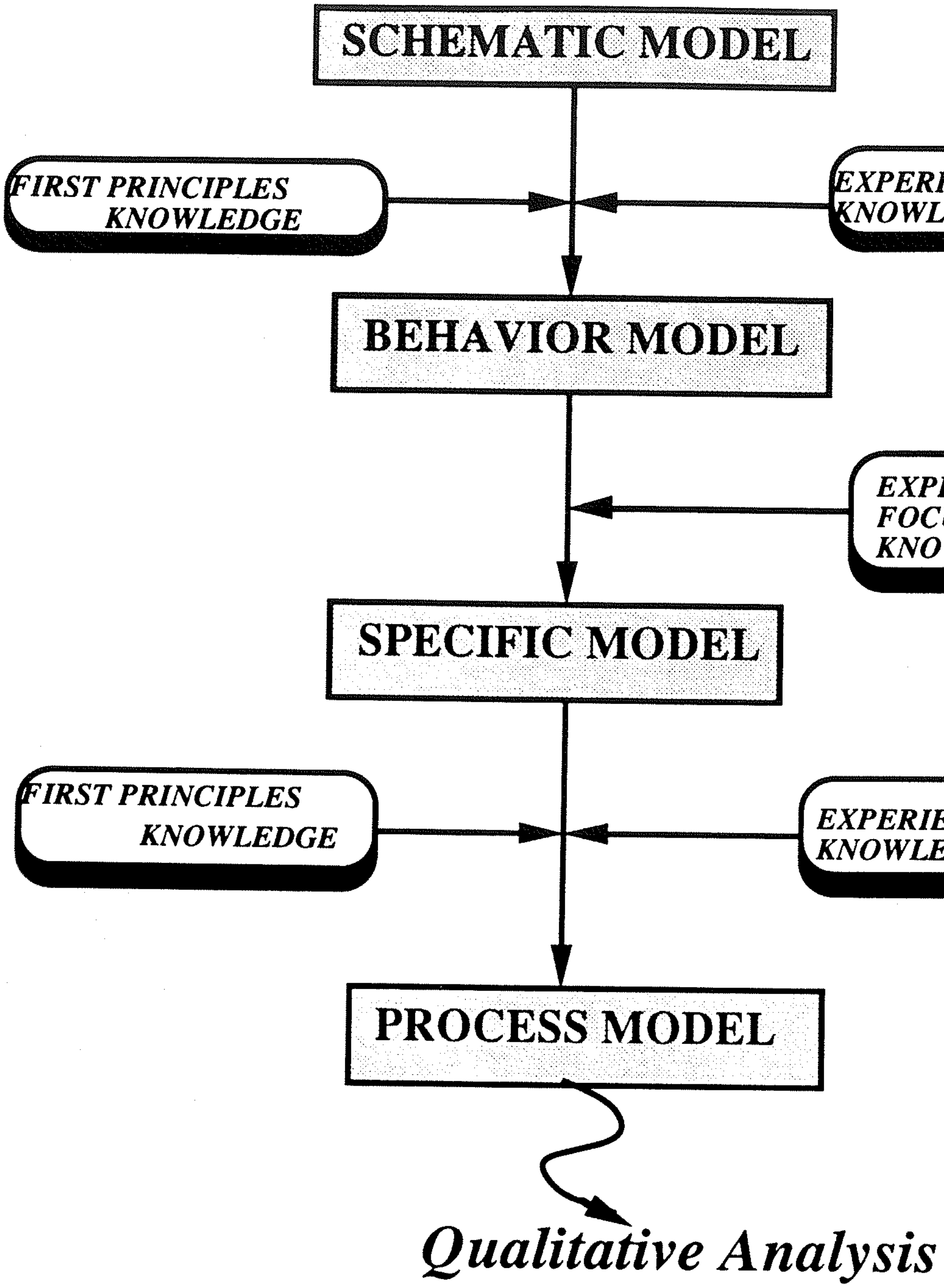


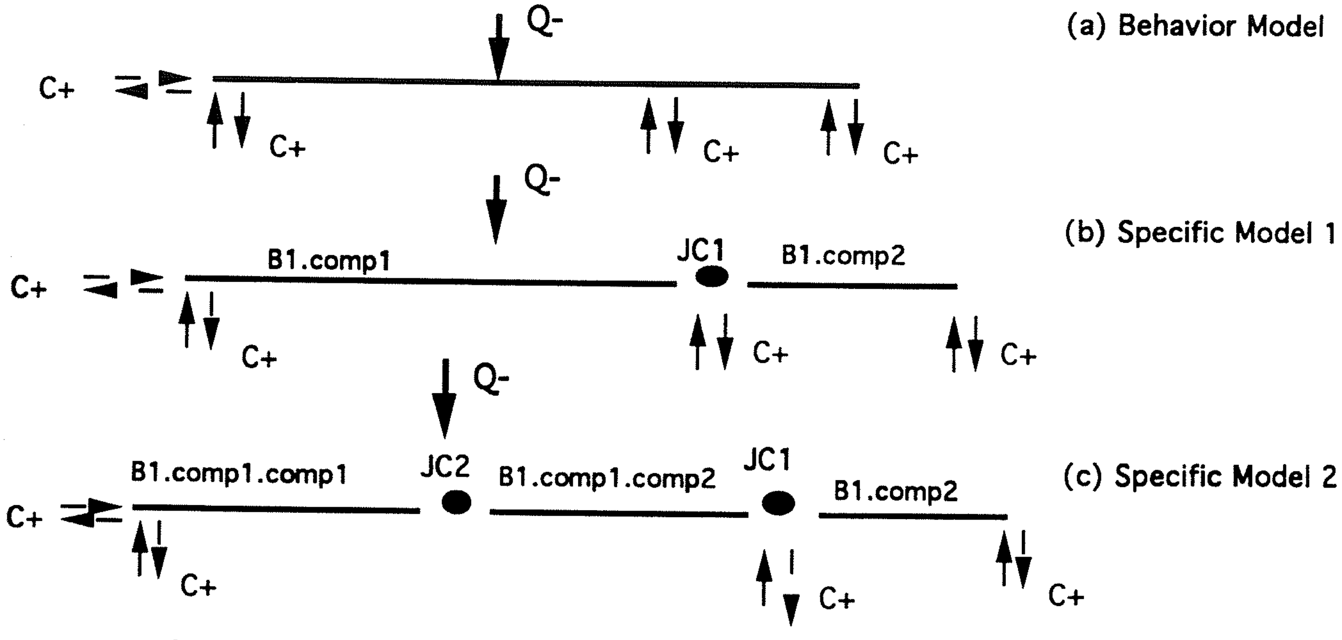
Figure 4: Model Generation Sequence

4.1 Behavior model

The first transformation derives a *behavior model* from the *schematic model.* A behavior model describes the existing interactions (e.g., forces) identified among structural objects. To generate a behavior model, each structural object is represented by its free body diagram. With the knowledge of how information is transferred by the support and connection objects, the system infers the existence of possible reactions and boundary conditions. An interaction map of the whole structure is generated by combining the free body diagrams of components showing the information flow (load transfer) among them. From this interaction map, the behavior pattern of the whole structure is identified.

The explicit representation of interactions among the structural objects provides a way to model the structural behavior that is often implicitly used in the reasoning process of a designer. This representation gives the user an opportunity to inspect the interactions of the various structural objects. If the user finds a discrepancy between the interaction map and the intended interactions, he/she will be able to rectify it at this initial stage by changing the schematic model.

Figure 5a illustrates the interaction map of the beam example (Appendix C - Figure C-3 illustrates the change in the values of the slots that contain the interaction information). Based on the component and support types (and also connection types if any), the system identifies the behavior pattern of the structure as a *continuous_beam.*



(⇌ illustrates the existence of a force with unknown direction)

Figure 5: Model Generation Example

4.2 Specific model

Next, the *behavior model* is transformed into a *specific model* by identifying the regions of interest with respect to the behavior of the structure. Examples of such regions of interest are the location of a concentrated load or a support along a component. The purpose of this transformation is to enable subsequent qualitative analysis at these regions. The system uses experiential knowledge to focus attention on such relevant regions. Identifying a region of interest results in the decomposition of the component into subcomponents. The model description is updated by adding new objects as specified by such rules.

The system searches for focusing rules relevant to the behavior pattern of the behavior model. In our example, this step results in the introduction of a *continuity-connection JC1* to the interaction map at the location of *Roller.support.C* as shown in Figure 5b. Then, *B1.comp1* and *B1.comp2* are added as substructures of beam *B1*. The structural representation is also updated in terms of new terminal instances corresponding to the new beam substructure instances (e.g., *Term.B1.comp1, Term.B1.comp2).* Next, the system considers the implication of the given loading context, and searches for a focusing rule to refine the interaction map in the region where the load applies. This will result in further decomposing of *B1.comp1* into *B1.comp1.comp1* and *B1.comp1.comp2* and adding the *continuity-connection JC2* to the interaction map at the location where the load *P* acts. The structural representation is also updated in terms of new terminal instances corresponding to the new beam substructure instances (e.g., *Term.B1.comp1.comp1, Term.B1.comp1.comp2,* etc.). The final result of the *specific model* generation is illustrated in Figure 5c (see also Figure C-4 and Figure C-5 of Appendix C).

4.3 Process model

A *process model* is defined as the set of applicable qualitative equations and relations, together with the set of physical parameters involved. The purpose of transforming the *specific model* into a *process model* is to identify active processes of the overall structure and the components. A process is defined as the possible behavior (e.g., bending, compression) of a structure in the context of the loading and boundary conditions. For a given specific model the system checks the preconditions of possible processes associated with each component to identify the active processes. QStruc represents processes as objects. Each process has references to applicable equations and relations.

In our example, *bending_process* is identified as an active process in the beam component. As a result, the applicable experiential knowledge and first principles are instantiated for each of the subcomponents of the specific model. The objects in the process model generated in terms of *equations, relations,* and *physical quantities* is shown in Figure 6. In Appendix C we give examples of KEE units representing *equations, relations,* and *physical quantities.*

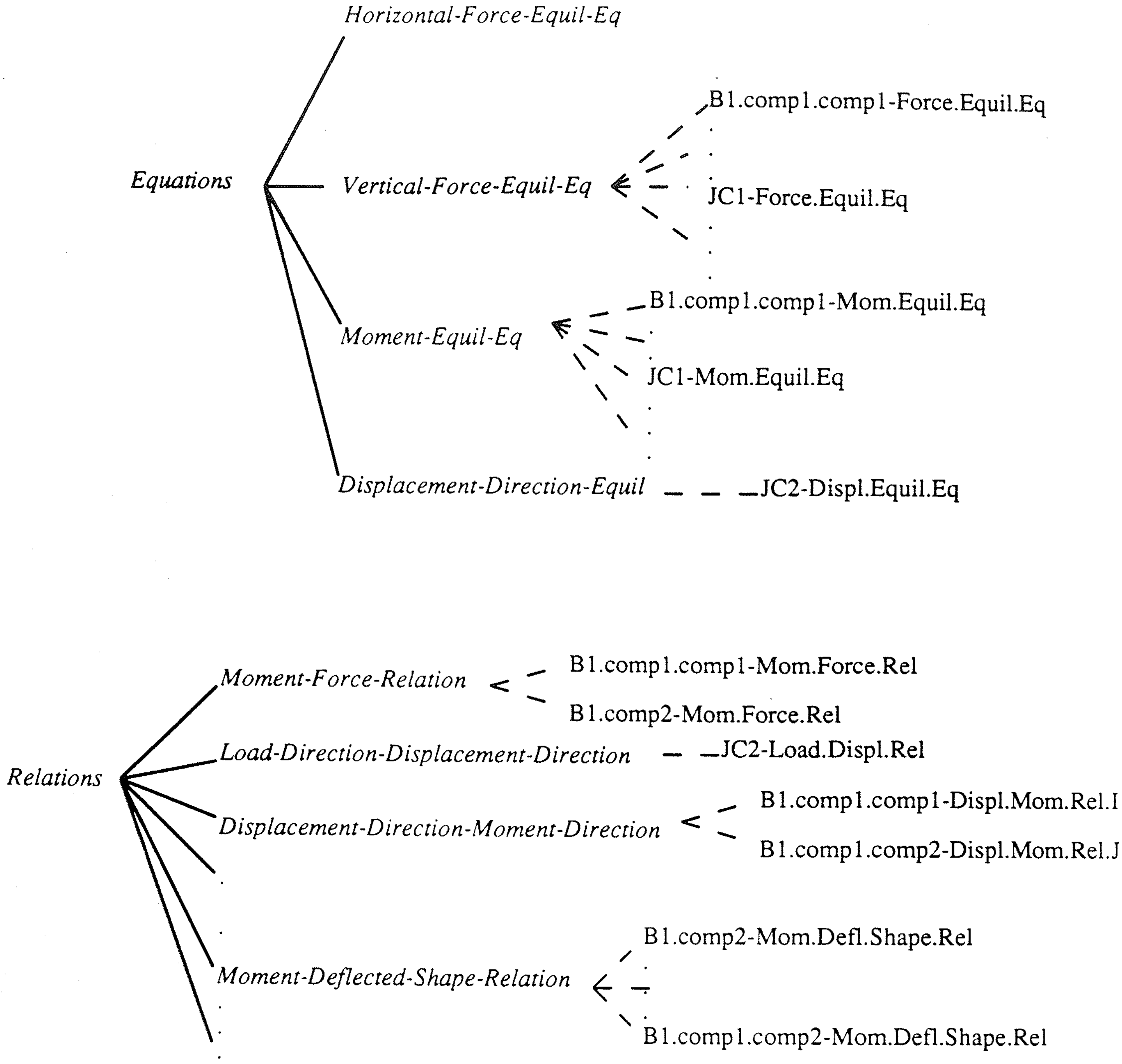


Fig. 5 Process model instantiation - equations, relations, physical quantities

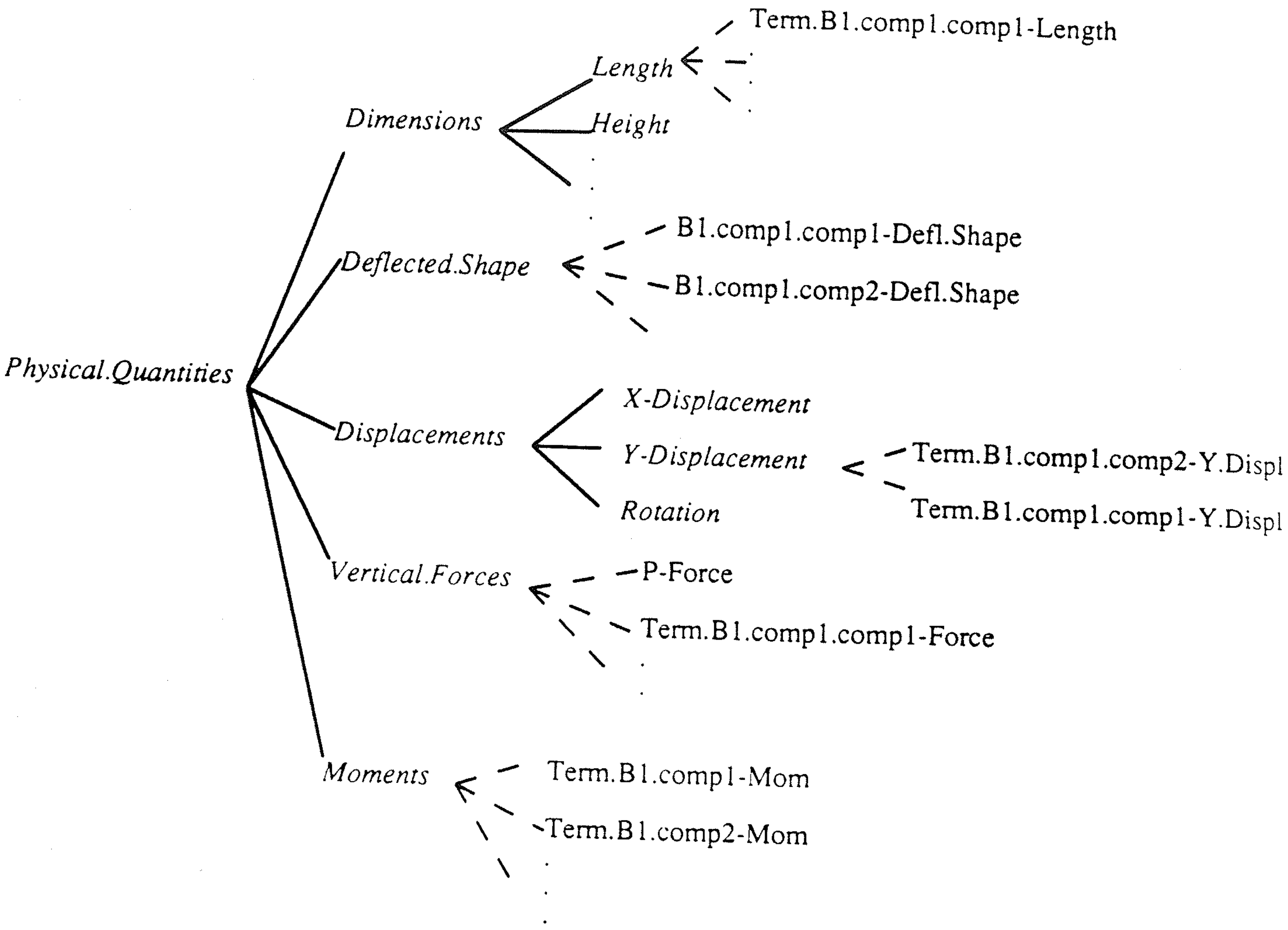


Fig. 5 Process model instantiation - equations, relations, physical quantities (continued)

5 QUALITATIVE ANALYSIS

A significant difference between qualitative analysis strategies employed by human engineers and quantitative analysis methods is that there is no fixed sequence of steps in the qualitative approach. In the numerical approach the sequence of solution steps is often predefined. In contrast, when analyzing a preliminary design, it is not obvious from what point the solution will emerge most effectively and in what way the solution process will proceed. It may be best to start with the deflected shape in one case, but with the bending moment in another case. Also, depending on the amount and type of information available, it may be more effective to apply first principles in some situations and experiential knowledge in others. Since we do not know in advance what types of information will be available, we need a flexible solution strategy that can infer as much as possible from whatever types and amount of information available.

To achieve such a flexible analysis strategy we employed the following mechanisms:

- a causal reasoning mechanism [Ref.10] which is used to identify a solution path from an arbitrary set of relations and equations that compose the process model;
- an agenda mechanism that consists of a sequence of tasks for qualitative analysis, where each task represents one reasoning step expressed as a rule, a method or a qualitative evaluation of an equation;
- qualitative calculus [Ref. 10] which is used to derive the structural response;
- Quantity Lattice [Ref. 18], which is used as a mechanism to order qualitative magnitudes of physical quantities, given additional first principle knowledge about them. This information is further used to reduce some of the ambiguities in the qualitative reasoning.

In the following subsections, we discuss the approach of representing behavior of structures by causal dependency networks, and deriving the qualitative behavior by qualitative evaluation. We then present the proposed solution algorithm.

5.1 Representing behavior of structures by causal dependency networks

The behavior of a structure is described by a causal model in terms of physical quantities and causal interactions among them. The physical parameters (e.g., moments, forces, displacements) can be represented as nodes in a graph. The causal interactions (e.g., moment equilibrium equation, load displacement relation) can be represented as arcs in such a graph (Figure 7). Interaction arcs are directional and denote strict causal precedence (e.g., *"P-Force* ⟶ *Term.B1.comp1.comp2-Y.Displ"* means that *Term.B1.comp1.comp2-Y.Displ* depends on *P-Force).* Analyzing the behavior of a given process model involves: (1) identifying the initial conditions of input (in our case the parameters with exogenous known values), (2) inferring the *causal dependency network* among the physical parameters, and (3) propagating the information across the causal interaction arcs. A causal dependency network for the example problem under consideration is shown in Figure 7.

The explicit representation of the behavior of a structure by the causal dependencies among the physical parameters enables the user to focus on the influences and interactions among specific components, processes and/or parameters of interest. This may further enable the user to focus the qualitative evaluation to the subgraph of the causal dependency network concerning the object(s) of interest. An example of focusing on the physical parameter *Term.B1.comp2--Mom* and its dependents is shown in Figure C-9 Appendix C.

5.2 Deriving the qualitative behavior by qualitative evaluation

Having generated the causal dependency network, the system transforms the information into a sequence of qualitative evaluation tasks to determine the structural response. For this purpose we have defined the following concepts:

- a *cluster* as the subgraph of the causally connected physical parameters belonging to the same structural component,
- a *cluster root* as the physical parameter that is the root node of a cluster and is shared with another cluster,
- a *priority of a cluster* as the number of new parameter values that can be derived in a cluster,
- a *rank of a cluster* as the longest path from cluster C_i to the root cluster C,
- a *task* as any causal link (equation or relation) between two nodes (parameters) that has to be executed in the qualitative evaluation step to derive the qualitative value of the dependent parameter. The naming convention for the tasks was chosen such that each name is self explanatory. The first part of the name denotes what structural component is considered and the second part of the name describes the task.
- an *agenda of tasks* as a list of tasks generated from the cluster hierarchy ordered in ascending order of priority.

Figure 7 shows an example to illustrate these concepts.

After the agenda of tasks has been generated, the system proceeds to execute the tasks. Each task involves qualitative evaluation to determine the deflected shape tendency, qualitative reactions, internal forces and moments. Since all such attributes are represented as slots of objects in the specific and process models, performing qualitative evaluation means filling as many of these slots as possible with qualitative values. The system infers unknown parameter values from the known ones by means of equations, relations, and methods.

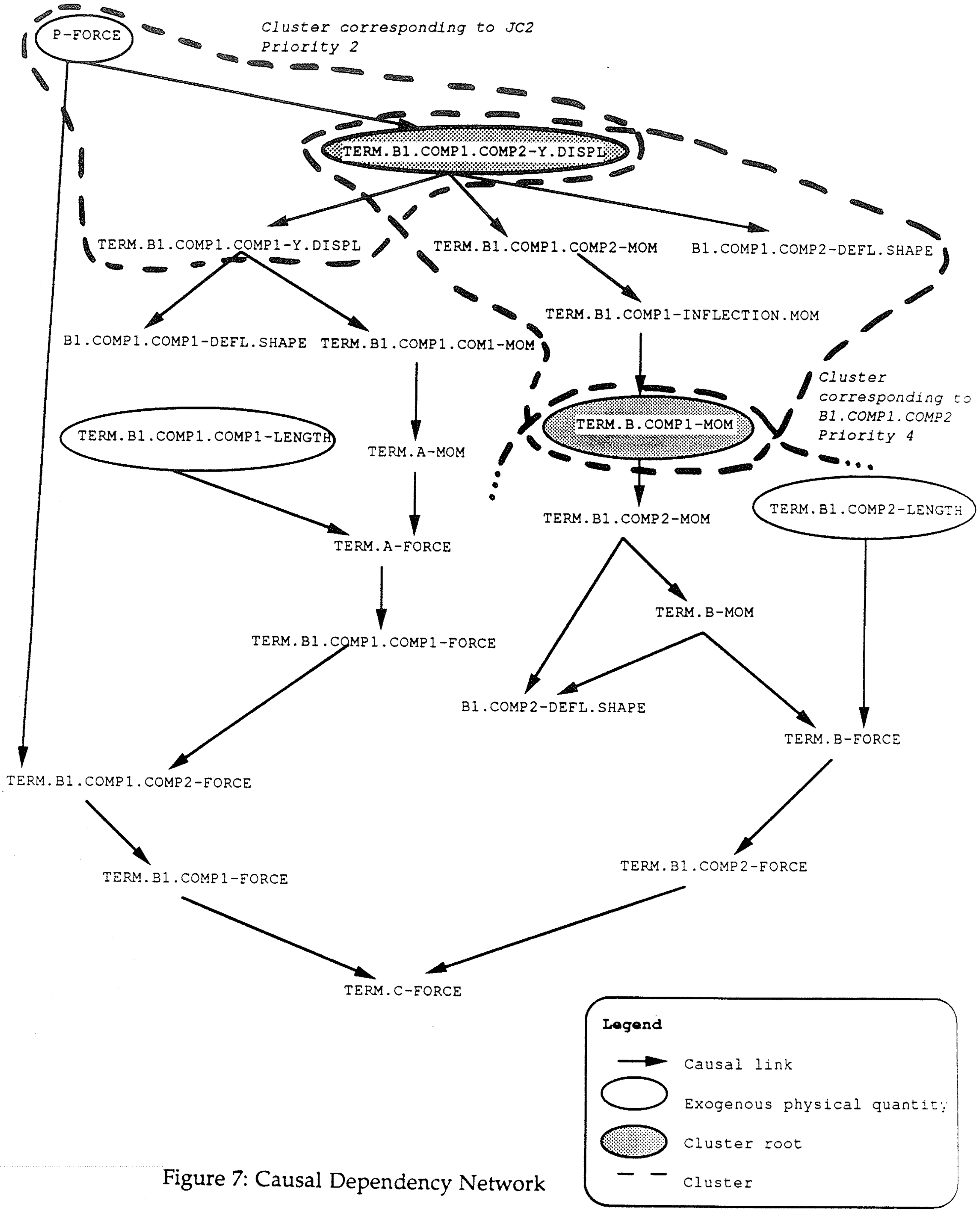


Figure 7: Causal Dependency Network

Explanatory example:

P-Force ⟶ *Term.B1.comp1.comp2-Y.Displ*

The two nodes (*P-Force, Term.B1.comp1.comp2-Y.Displ*) represent the physical quantities that are linked by the directed arc representing the relation *JC1-Load.Displ.Rel.* The relation states that the terminal displaces in the same direction as the force. This causal link *JC1-Load.Displ.Rel* represents a qualitative evaluation task in the future agenda of tasks. The execution of this task will result in the qualitative value of the *Y* displacement of *Term.B1.comp1.comp2-Y.Displ.*

Figure 7 Causal Dependency Network (continued)

The way qualitative equations are used to infer unknown values can be illustrated as follows: Suppose we have in the agenda a task *B1.comp1.comp1-Force.Equil.Eq* that will trigger the force equilibrium equation of *B1.comp1:*

Term.A-Force + *Term.B1.comp1.comp1-Force* = *0*

The substitution of the parameters with their current qualitative values gives:

$(Q+) + (C+) = 0$

Recall the value C+ indicates that a constraint exists but the value is unknown. The unknown physical quantity *Term.B1.comp1.comp1-Force* can be evaluated by using the definition of qualitative negation operator (see Appendix A), which gives:

Term.B1.comp1.comp1-Force = *Q-*

In general, if we know the values of all but one of the variables in an equation, we can attempt to determine the value of the unknown variable from the known values based on the definitions of qualitative arithmetic operators. However, we cannot always uniquely determine the qualitative value of the remaining variable due to the inherent ambiguity of qualitative calculus. In Qstruc, we use a program called Quantity Lattice [Ref. 18], which maintains information about the partial ordering relations among the quantities. By consulting Quantity Lattice, our system can reduce some of the ambiguities in qualitative reasoning. An example of using Quantity Lattice for resolving ambiguity is given in Appendix B.

5.4 Solution algorithm

We employ a "greedy" algorithm for inferring a causal solution path and executing the tasks in the agenda. The tasks are executed on a last-in-first-out basis. Processing a task (such as firing a rule, calling a method, or qualitatively evaluating an equation) produces a new parameter value. In this algorithm the system first picks a task that is most promising based on existing information about the parameters (exogenous parameters and/or cluster root), applicable knowledge and causal ordering among the parameters. This task is executed to produce as many additional conclusions as possible that may result from the exogenous parameters and/or cluster root. The proposed *"greedy" depth-first algorithm* can be summarized as follows:

(1) Identify all the parameters of the models such that their values are already known.
(2) For each such parameter, identify the possible analysis tasks (i.e. equations and relations that are associated with the parameter and that can be used to derive the value of other parameters). Perform a causal ordering analysis for the generated process model and represent the dependency among parameters explicitly in a *causal dependency network.*
(3) Cluster the derived causal dependency network and identify the cluster roots together with the priority of each cluster.
(4) Clusters naturally represent a hierarchy that reflects the original causal ordering. Order the clusters according to their rank and priority, such that the highest rank comes before the lowest rank and lower priority comes before highest priority.
(5) Tasks in the agenda are ordered in such a way, that the task that computes the value of a parameter p is pushed onto the agenda before the tasks that compute the parameters on which p depends.
(6) Derive the qualitative behavior of the given structure by executing these tasks in a last-in-first-out manner until the agenda is empty.

6 Examples

In this section, applications of the Qstruc system to a continuous beam and 2D frame problems are presented.

6.1 Analysis of Continuous Beam

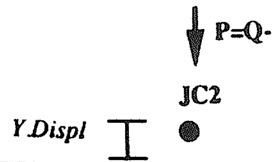
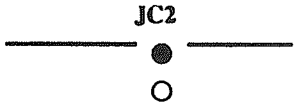
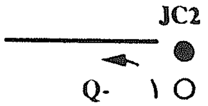
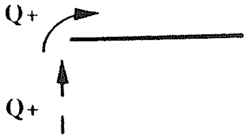
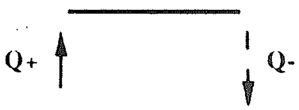
In this section, we describe the solution strategy with the example shown in Figure 3. Figure 6 illustrates part of the causal dependency network of the process model derived in the model generation step. The nodes represent the physical parameters that describe the behavior of the structure. The directed arcs represent the equations or relations among the connected nodes. The leftmost physical quantities represent the identified exogenous parameters.

After a solution path is identified by the causal reasoning mechanism, the system generates the agenda of tasks from the hierarchy of clusters that are ordered in ascending order of priority. Each task is carried out by applying qualitative calculus and/or Quantity Lattice. Consider the tasks of the qualitative evaluation steps in the agenda illustrated in Figure 8. Processing some of these tasks is explained in Table 1, where the qualitative evaluation results are illustrated schematically. In the rightmost column the dashed arrows represent unknown parameters derived at that step. Details of the reasoning steps are given in Appendix C.

JC2-Load.Displ.Rel
JC2-Displ.Equil.Eq
B1.comp1.comp1-Displ.Mom.Rel.I
B1.comp1.comp1-Displ.Defl.Shape.Rel
B1.comp1.comp1-Mom.Equil.Eq
B1.comp1.comp1-Mom.Force.Rel
B1.comp1.comp1-Force.Equil.Eq
B1.comp1.comp2-Displ.Mom.Rel.J
B1.comp1.comp2-Displ.Defl.Shape.Rel
B1.comp1.comp2-Inflection.Rel.I
B1.comp1.comp2-Inflection.Rel.J
JC1-Mom.Equil.Eq
B1.comp2-Mom.Equil.Eq
B1.comp2-Mom.Defl.Shape.Rel
B1.comp2-Mom.Force.Rel
B1.comp2-Force.Equil.Eq
JC2-Force.Equil.Eq
B1.comp1.comp2-Force.Equil.Eq
JC1-Force.Equil.Eq

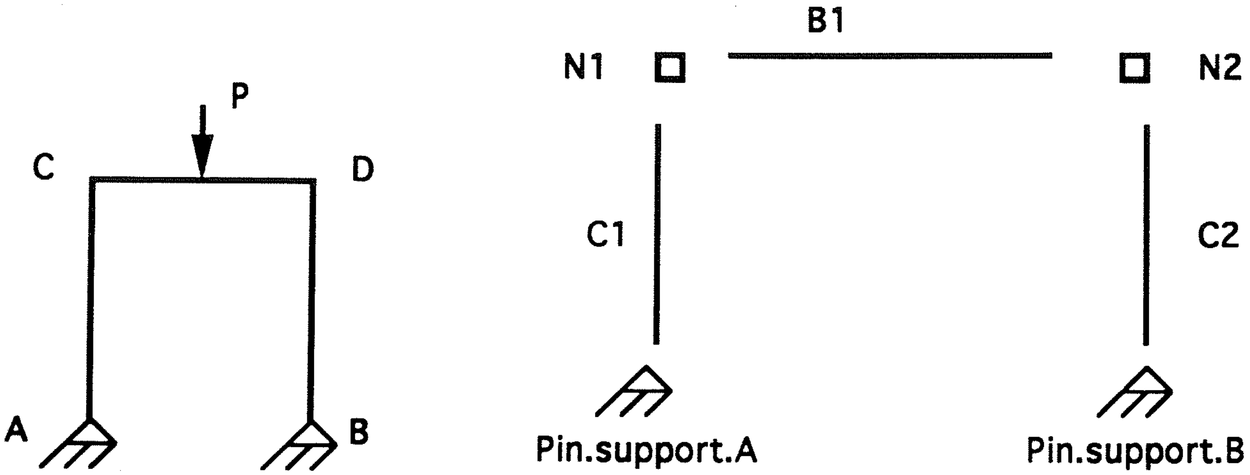
Figure 8: Agenda of Tasks

Table 1 Examples from the Qualitative Evaluation Results

Task	Applied Knowledge and Result	Qualitative evaluation
JC2-Load.Displ.Rel	Represents the relation between load direction and displacement direction at a *continuity-connection.* Derives the displacement direction *(Q-)* at *Term.B1.comp1.comp2* of *JC2.*	
JC2-Displ.Equil.Eq	Represents the displacement equilibrium principle of the two adjacent terminals to a *continuity-connection.* Derives the displacement direction at *Term.B1.comp1.comp1* of *JC2.*	
B1.comp1.comp1-Displ.Mom.Rel.I	Represents the relation between displacement direction at a *continuity-connection* and moment direction. Derives the moment direction at *Term.B1.comp1.comp1* of *B1.comp1.comp1.*	
B1.comp1.comp1-Displ.Defl.Shape.Rel	Represents the relation between deflected shape, support condition and load context. Derives the deflected shape of *B1.comp1.comp1.*	
B1.comp1.comp1-Mom.Equil.Eq	Represents the moment equilibrium principle on a component. Derives the moment *Term.A-Mom.Force* of the vertical force at *Term.A* of *B1.comp1.comp1.*	
B1.comp1.comp1-Mom.Force.Rel	Represents the definition of a moment given by a force with respect to a point. Derives the direction of the force at *Term.A* of *B1.comp1.comp1.*	
B1.comp1.comp1-Force.Equil.Eq	Represents the force equilibrium principle on a component. Derives the force at *Term.B1.comp1.comp1* of *B1.comp1.comp1.*	
. . .	. . .	. . .

6.2 Analysis of Frame Structures

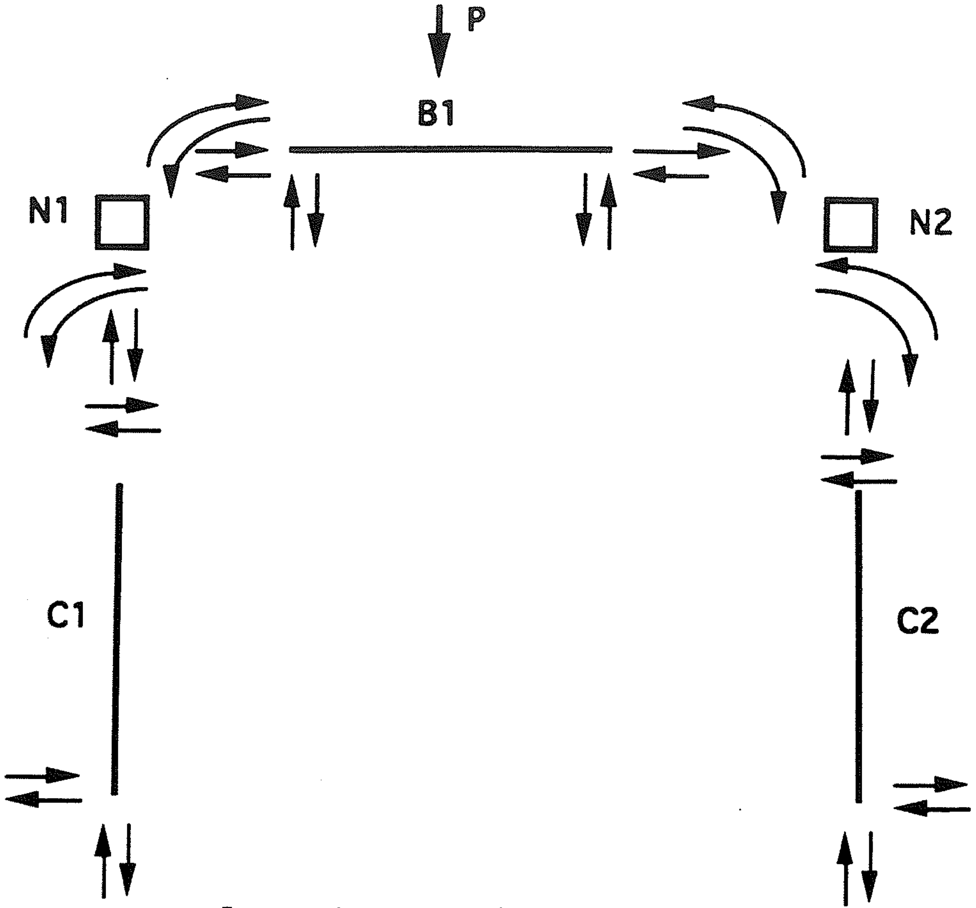
To further demonstrate the capability of QStruc we will present the analysis of two 2-D frame structures. In the first example a simple frame structure shown in Figure 9 is employed. It has pin supports at points A and B and rigid joints at nodes C and D, and a concentrated load acts at the center of the horizontal beam CD. The representation of this frame is also shown in this figure, where *B1* is the instance representing the beam CD, *C1* and *C2* represent the two columns, *N1* and *N2* represent the rigid nodes C and D, and *Pin.support.A* and *Pin.Support.B* represent the two supports of the frame. Figure 10 shows the result of the model generation steps that transform the schematic model into a behavior model (Figure 10a) and a specific model (Figure 10b). The causal dependency network derived for this frame example is shown in Figure 11. The resulting agenda of tasks is given in Figure 12. Furthermore, the qualitative behavior of this frame and selective qualitative values inferred by Qstruc are illustrated in Figure 13. Note that the qualitative values *Q-Q0Q+* and *Q+Q0Q-* indicate an inflection point in the deflected shape of the two subcomponents of beam B1 (Figure 13 - Deflected Shape).



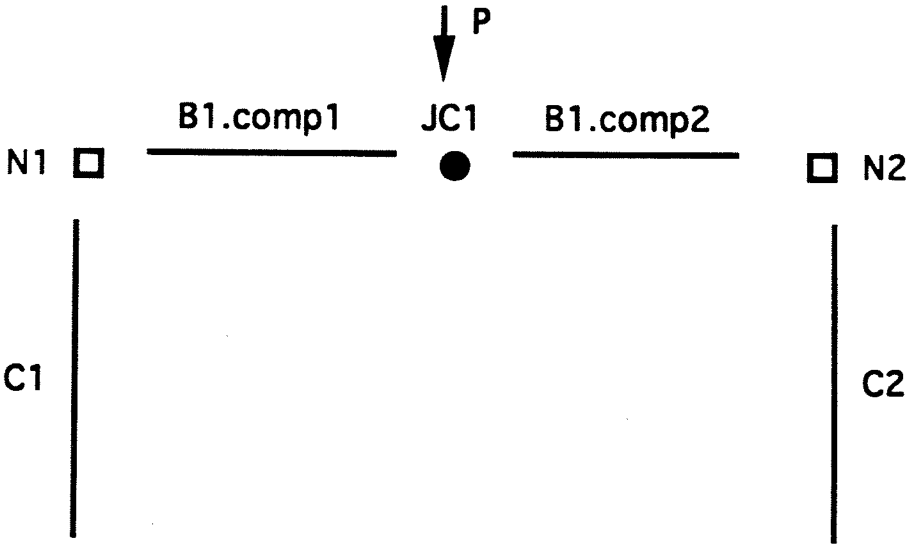
Schematic Model

QStruc Representation of Frame Example

Figure 9: Frame Example



Interaction Map of Behavior Model



Specific Model

Figure 10: Model Generation Results

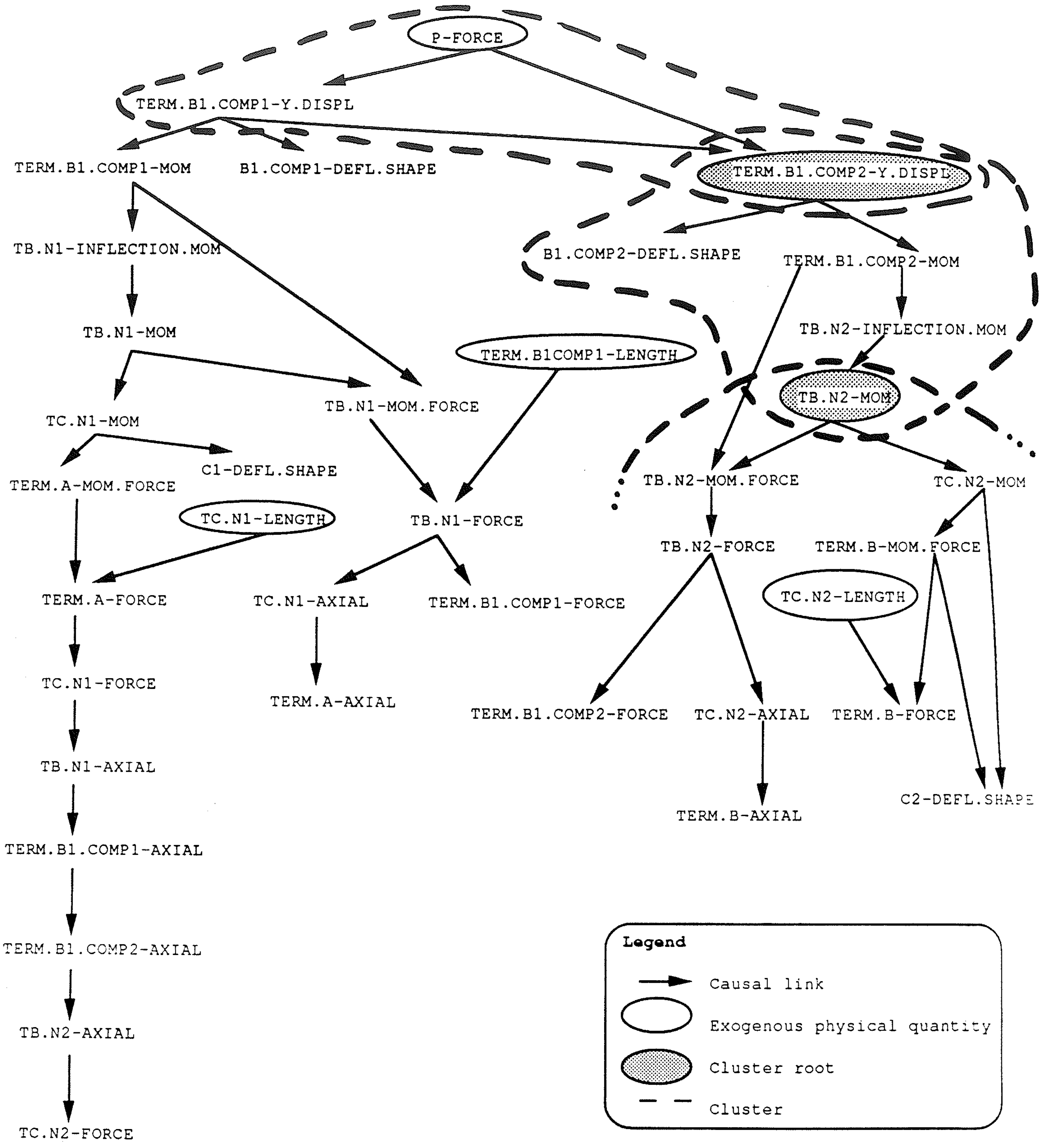
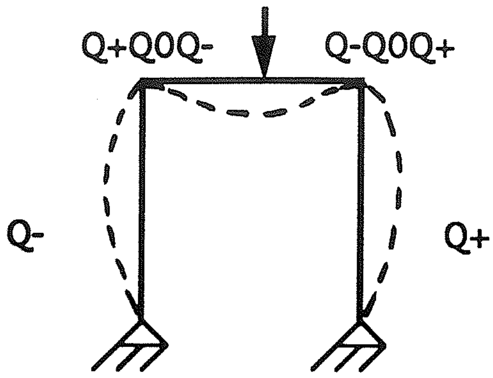


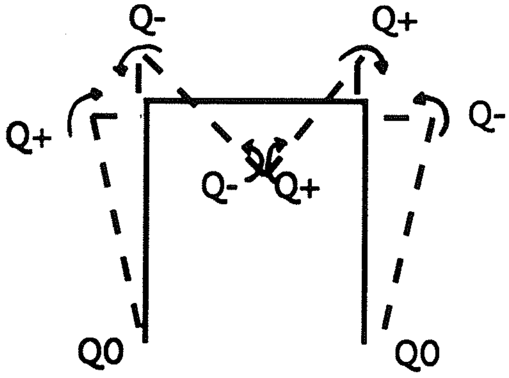
Figure 11: Causal Dependency Network of Frame Example

JC1-Load.Defl.Y.Rel
JC1-Defl.Equil.Y.Rel
B1.comp1-Displ.Defl.Shape.Rel
B1.comp1-Displ.Mom.Rel.I
B1.comp1-Inflection.Rel.J
B1.comp1-Inflection.Rel.I
N1-Moment.Equil.Eq
B1.comp2-Displ.Defl.Shape.Rel
B1.comp2-Displ.Mom.Rel.J
B1.comp2-Inflection.Rel.I
B1.comp2-Inflection.Rel.J
N2-Moment.Equil.Eq
B1.comp2-Moment.Equil.Eq
B1.comp2-Mom.Force.Rel
C2-Moment.Equil.Eq
C2-Mom.Defl.Shape.Rel
C2-Mom.Force.Rel
N2-Force.Equil.Eq
B1.comp2-Force.Equil.Eq
C2-Axial.Equil.Eq
C1-Moment.Equil.Eq
C1-Mom.Defl.Shape.Rel
C1-Mom.Force.Rel
C1-Force.Equil.Eq
B1.comp1-Moment.Equil.Eq
B1.comp1-Mom.Force.Rel
N1-Axial.Equil.Eq
B1.comp1-Axial.Equil.Eq
JC1-Axial.Equil.Eq
B1.comp2-Axial.Equil.Eq
N2-Axial.Equil.Eq
N1-Force.Equil.Eq
B1.comp1-Force.Equil.Eq
C1-Axial.Equil.Eq

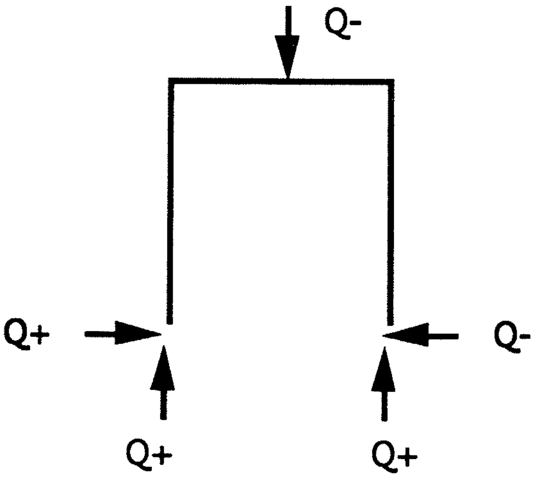
Figure 12: Agenda of Tasks for the Frame Example



Deflected shape



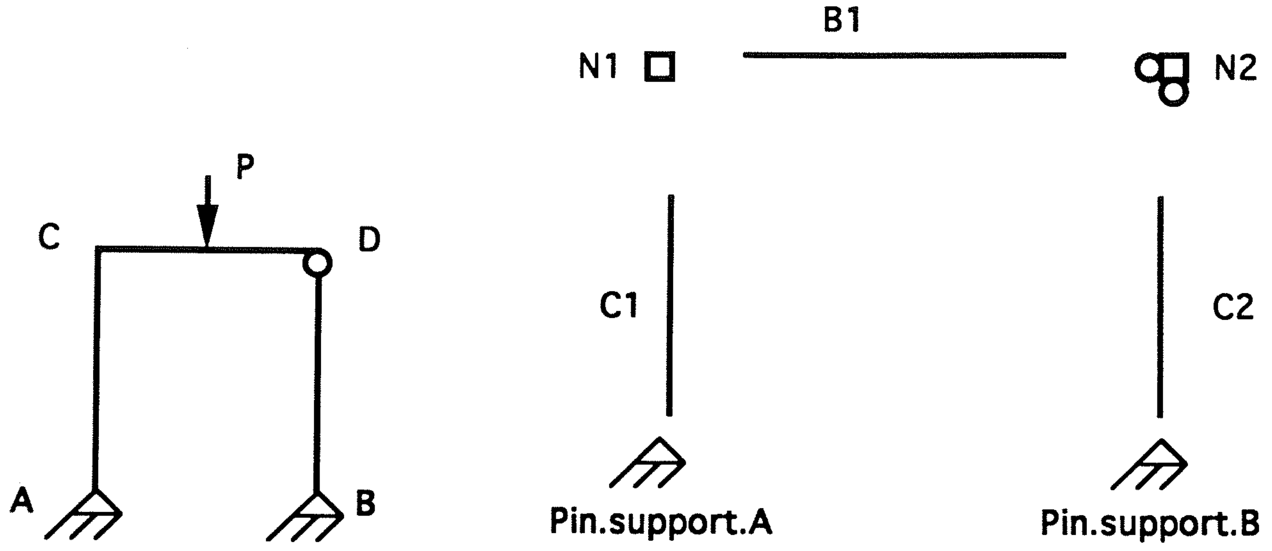
Moment Diagram



Reactions

Figure 13: Qualitative Behavior of Frame Example

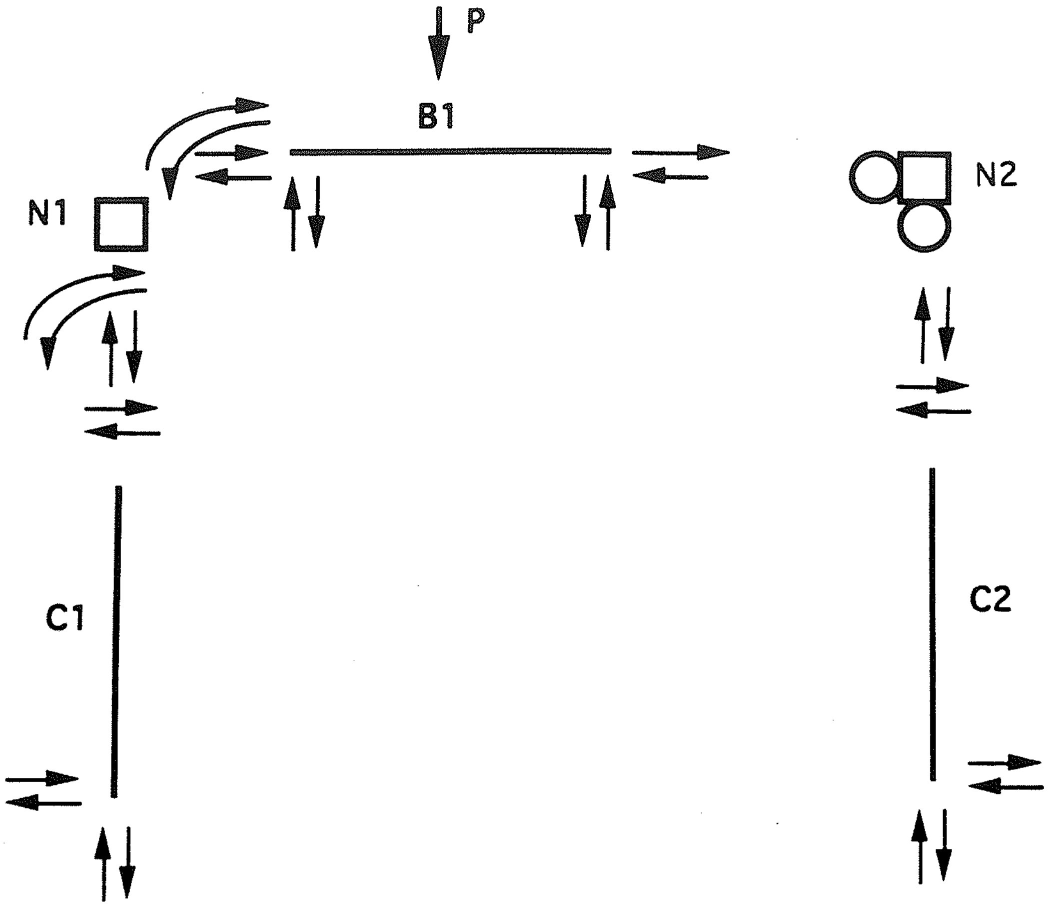
In the second example, we attempt to further demonstrate the use of a qualitative analysis tool in deriving the unusual behavior of a frame structure shown in Figure 14. As shown in Figure 14, this frame has pin supports at points A and B, a hinge at node D, and a concentrated load acts at the center of the horizontal beam CD. The representation of this frame is shown in this figure, where *B1* is the instance representing the beam CD, *C1* and *C2* represent the two columns, node *N1* and *N2* represent respectively the rigid joint C and the hinged joint D, and *Pin.support.A* and *Pin.Support.B* represent the two supports of the frame. Figure 15 summarizes the result of the model generation steps that transform the schematic input model into a behavior model (Figure 15a) and a specific model (Figure 15b). As illustrated in Figure 16, this frame structure poses an unusual behavior that has a zero moment at the rigid node C. This is because the horizontal reaction at the support B is zero since moments about node D must be zero. Thus the horizontal reaction at support A will be zero and the moment at node C, due to the horizontal reaction at A will be zero. The whole frame will sway to the right in order to release the moment at node C. This behavior is inferred by Qstruc. The reasoning steps of QStruc are illustrated by the causal dependency network (Figure 17) and the agenda of tasks (Figure 18).



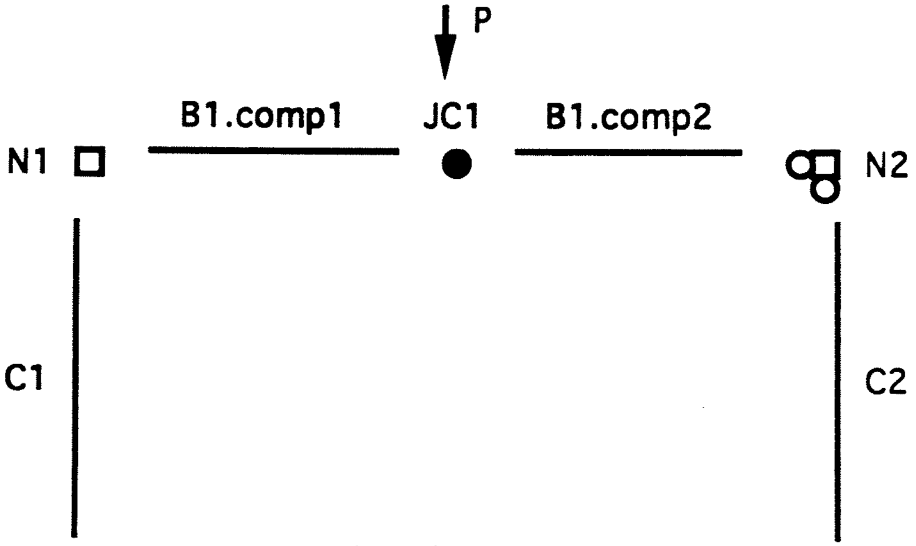
Schematic Model

QStruc Representation of Frame Example

Figure 14: Hinged Frame Example

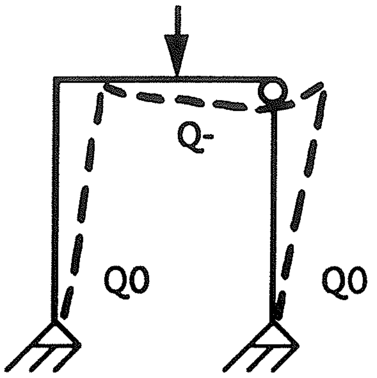


Interaction Map of Behavior Model

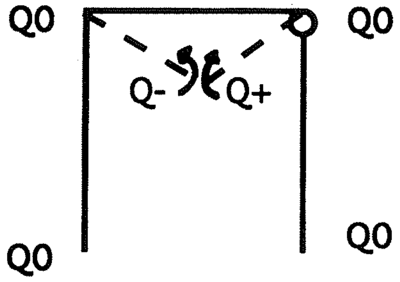


Specific Model

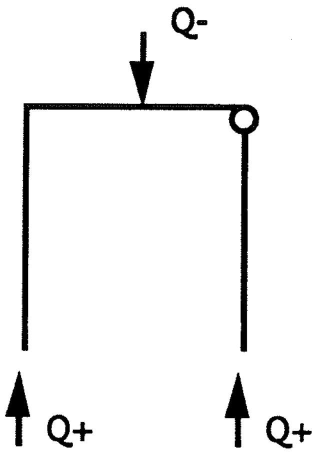
Figure 15: Model Generation Results



Deflected shape



Moment Diagram



Reactions

Figure 16: Qualitative Behavior of Hinged Frame Example

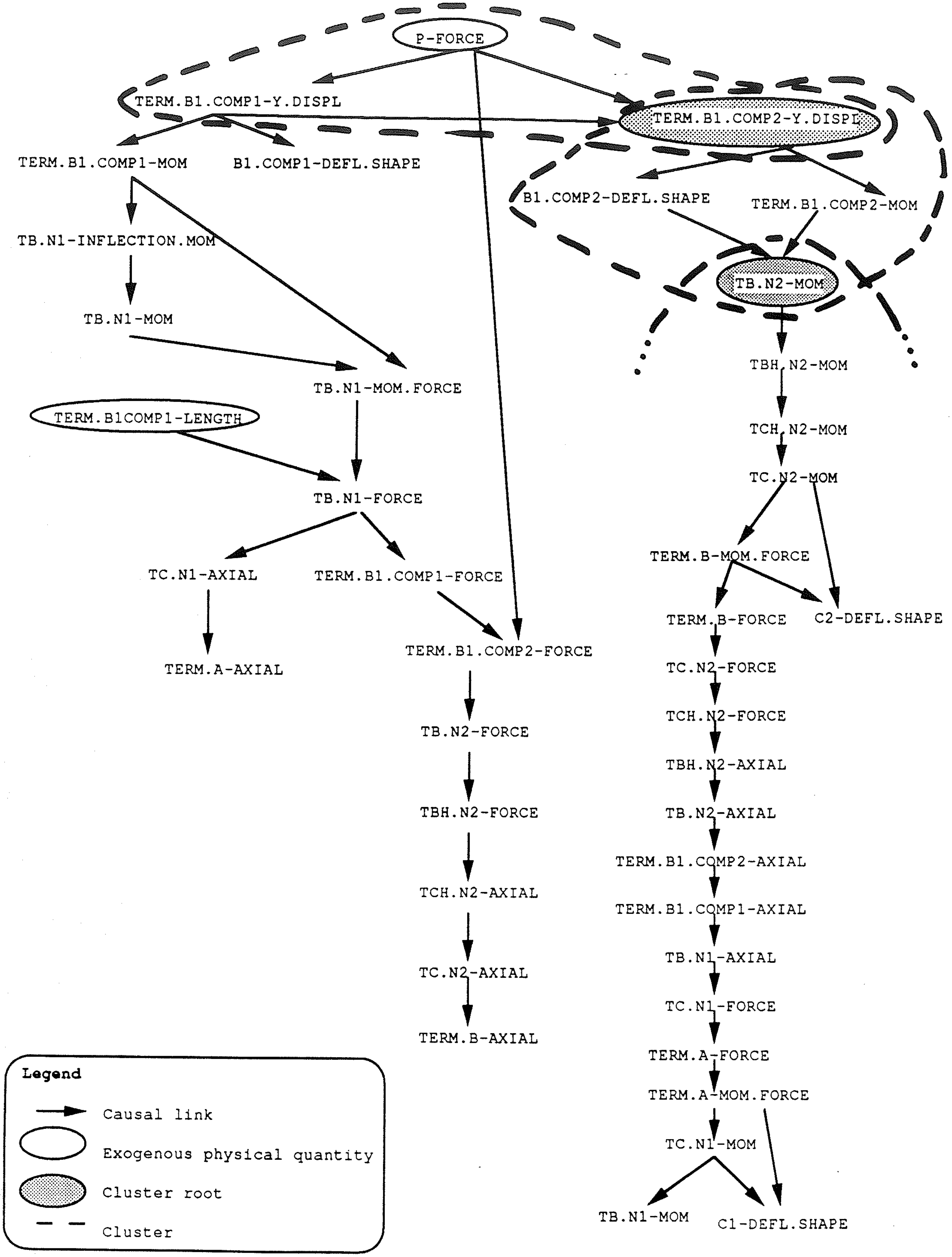


Figure 17: Causal Dependency Network of Hinged Frame Example

JC1-Load.Defl.Y.Rel
JC1-Defl.Equil.Y.Rel
B1.comp1-Displ.Defl.Shape.Rel
B1.comp1-Displ.Mom.Rel.I
B1.comp1-Inflection.Rel.J
B1.comp1-Inflection.Rel.I
B1.comp2-Displ.Defl.Shape.Rel
B1.comp2-Displ.Mom.Rel.J
B1.comp2-Mom.Defl.Shape.Rel
HB.N2-Moment.Equil.Eq
N2-Moment.Equil.Eq
HC.N2-Moment.Equil.Eq
C2-Moment.Equil.Eq
C2-Mom.Defl.Shape.Rel
C2-Mom.Force.Rel
C2-Force.Equil.Eq
HC.N2-Force.Equil.Eq
N2-Axial.Equil.Eq
HB.N2-Axial.Equil.Eq
B1.comp2-Axial.Equil.Eq
JC1-Axial.Equil.Eq
B1.comp1-Axial.Equil.Eq
N1-Axial.Equil.Eq
C1-Axial.Equil.Eq
C1-Mom.Force.Rel
C1-Moment.Equil.Eq
C1-Mom.Defl.Shape.Rel
N1-Moment.Equil.Eq
B1.comp1-Moment.Equil.Eq
B1.comp1-Mom.Force.Rel
N1-Force.Equil.Eq
B1.comp1-Force.Equil.Eq
C1-Axial.Equil.Eq
JC1-Force.Equil.Eq
B1.comp2-Force.Equil.Eq
HB.N2-Force.Equil.Eq
N2-Force.Equil.Eq
HC.N2-Axial.Equil.Eq
C2-Axial.Equil.Eq

Figure 18: Agenda of Tasks for the Frame Example

These examples demonstrate the potential use of qualitative analysis in deriving the behavior of structures in the conceptual design stage when information is incomplete and qualitative. QStruc enables the user to gain a better understanding of structural behavior in case of non-trivial problems. It also provides a capability to focus on critical behavior regions and to visualize the causal influence of a potential change of a parameter on its dependents. Last but not least, it records the analysis steps that have taken place in deriving the structural response.

7 CONCLUSIONS

In this report we have presented a framework QStruc for qualitative analysis of structures that could serve as a useful analysis tool for preliminary design of structures. This framework is based on first principles as well as experiential knowledge. We have described the representation scheme for structures and the domain knowledge. We have discussed a series of model transformations, each of which is useful for analyzing a specific behavior aspect of a structure. The system enables the user to explicitly focus on specific structural behavior, on regions relevant for the behavior of interest, and on the active processes for the given scenario.

We proposed a "greedy" algorithm for qualitative analysis which enables the system to:

- identify a solution path based on the causal ordering among parameters,
- determine the response behavior as much as possible from known parameters, and
- focus at different levels of abstraction (e.g., the dependency of specific parameter, processes and physical laws, components).

From our knowledge, such an explicit representation of the causal dependency among physical parameters has not been used extensively in structural engineering analysis. This representation scheme seems to work effectively and is useful to explain the analysis of moments, reactions, and deflected shape tendencies. Furthermore, when defining the solution strategy, we resolve some ambiguities by employing Quantity Lattice together with additional information about the physical parameters. In this preliminary work, we have demonstrated the potential use of a qualitative approach in structural modeling and analysis. We believe that this approach and the proposed framework, when fully validated, will become a useful tool for supporting preliminary analysis and design of structures.

The benefit of a qualitative reasoning approach is that the designer can gain some insight into the structural behavior in the early stage of design, before most of the parameters have been assigned exact values. This behavior based reasoning may enable the structural engineer to identify potential design problems in the early stages of design. It may also help the engineer to define and visualize constraints and dependencies among parameters.

The approach developed in QStruc is based on reasoning at a micro-component level. In order to enable the qualitative analysis of more complex structures there is a need to enhance the reasoning strategy with a capability of reasoning at a macro-substructure or structure level. Some other limitations of the presented framework are inherent results of the current limitations of the qualitative reasoning technology. These limitations regard the generation of imprecise models and of insufficient (some times ambiguous) output data, caused by the initial qualitative, incomplete description of the structural system and the impreciseness of the information used. Another limitation regards cases of incomplete prediction of structural behavior caused by currently applicable types of constraints used and by limited types of developed qualitative reasoning strategies.

Considering the complexity of the structural design process and the present limitations of qualitative reasoning technology, the future directions for improving the present framework include:

- extending the QStruc prototype so that it will be able to analyze more complex structures [Ref. 8];
- integrating QStruc in a broader CAE framework to enhance the conceptual design process - alternative configuration generation, interpretation of design alternatives, and preliminary sizing of components;
- developing a friendly and expressive user interface that will enable the engineer to dynamically use the modeling, analysis, and interpretation capabilities of QStruc in the conceptual design stage;
- extending the current solution strategy with new reasoning methodologies that would enable the transition from a qualitative model to a quantitative model.

REFERENCES

1. Cagan, J. and Agogino, A.M., "Innovative Design of Mechanical Structures from First Principles." *AI EDAM*, Vol.1, No.3, pp169-186, 1987.

2. DeKleer, J., "How Circuits Work," *Artificial Intelligence* 24, pp205-280, 1984.

3. DeKleer, J. and Brown, J., "A Qualitative Physics Based on Confluences," *Artificial Intelligence* 24, pp7-83, 1984.

4. DeKleer, J. and Brown, J., "Theories of Causal Ordering," *Artificial Intelligence* 29, 1986, pp33-61.

5. Forbus, K.D., "Qualitative Process Theory," *Artificial Intelligence* 24, pp85-168, 1984.

6. Franke, D.W. and Dvorak, D.L., "Component Connection Models," *Workshop on Model Based Reasoning, held in conjunction with the Eleventh Joint Conference on AI*, Detroit, Michigan, USA, pp97-101, 1989.

7. Fruchter, R. and Gluck, J., "Application of AI Techniques to the Analysis of Structures." *Computers & Structures,* Vol.30, No.3, pp747-753, 1988.

8. Fruchter, R. and Krawinkler, H., "QLRS: An Approach for Qualitative Lateral Resistant System Interpretation." to be presented at the *8th ASCE Conference on Computing in Civil Engineering,* Dallas 1992 (in preparation).

9. Ganguly, J., Kausel, E. and Sriram, D., "A Knowledge Based System for Deriving Qualitative Seismic Behavior." *Computer Utilization in Structural Engineering, Proc. of the ASCE Structures Congress,* ed. Nelson J.K., pp418-427, 1989.

10. Iwasaki, Y., "Qualitative Reasoning." *The Handbook of Artificial Intelligence,* Vol. IV, ed. Barr A., Cohen P.R. and Feigenbaum E.A., Addison-Wesley, pp323-414, 1989.

11. Iwasaki, Y. and Simon, H.A., "Causality in Device Behavior." *Artificial Intelligence,* Vol.29, pp3-32, 1986.

12. Kuipers,B., "Commonsense Reasoning about Causality: Deriving Behavior from Structure," *Artificial Intelligence* 24, 169-203, 1984.

13. Kuipers, B., "Qualitative Simulation," *Artificial Intelligence* 24, 289-339, 1984.

14. Roddis, K.W. and Martin, J.L., "Qualitative Reasoning about Steel Bridge Fatigue and Fracture," *Fifth International workshop on Qualitative reasoning about Physical Systems,* ed. B. Kuipers, pp302-317, 1991.

15. Slater, J.H. "Qualitative Physics and the Prediction of Structural Behavior." *Proc. of ASCE Symposium Expert Systems in Civil Engineering,* ed. Kostem C.N. and Maher M.L., pp239-248, 1986.

16. Simon, H. A., "On the definition of the causal relations," *Journal of Philosophy* 49, pp517-528, 1952.

17. Simon, H. A., " Causal Ordering and Identifiability," *Studies in Econometric Methods,* Wiley, New York, pp49-74, 1953.

18. Simmons, R., "Commonsense Arithmetic Reasoning." *Proc. of AAAI86,* Morgan Kaufmann, San Mateo, CA, pp118-124, 1986

19. *Readings in Qualitative Reasoning about Physical Systems,* ed. Weld, D.S. and deKleer, J., Morgan Kaufman, San Mateo, CA, 1990.

20. *Qualitative Reasoning about Physical Systems,* ed. Bobrow, D.G., MIT Press, MA, 1984.

Appendix A Examples of qualitative operators

The following tables define qualitative operators on qualitative values. For a complete discussion of qualitative calculus see [Ref. 10].

Table A-1 Qualitative Addition

[y]

	+	Q-	Q0	Q+
	Q-	Q-	Q-	?
[x]	Q0	Q-	Q0	Q+
	Q+	?	Q+	Q+

Table A-2 Qualitative Negation

[x]	[y]= -[x]
Q-	Q+
Q0	Q0
Q+	Q-

(? represents an unknown qualitative result;
[x] and [y] stand for two qualitative parameters)

Appendix B Use of Quantity Lattice in reducing qualitative calculus ambiguity

Consider a force equilibrium equation consisting of three forces, an unknown quantity $F1$ and two known quantities $F2$ and $F3$.

$$F_1 + F_2 + F_3 = 0$$

$$(C+) + (Q-) + (Q+) = 0$$

Relying only on qualitative calculus would require the consideration of all three cases $Q-$, $Q0$, $Q+$ for the unknown F_1, leading to ambiguous solutions. This can be resolved if there is a means to reason about the relative magnitudes of the two other forces based on first principles. For example, we can further employ the information that the relationship among the lever arms L_2 and L_3 (of the forces F_2 and F_3) is $L_2 < L_3$, and that the moment equilibrium equation with respect to the *application point* of F_1 is:

$$F_2 L_2 + F_3 L_3 = 0$$

Using Quantity Lattice, the relationship $F_2 > F_3$ between the two forces can be derived. This information can further be used to decide that the resulting qualitative value of F_1 is $Q+$.

Appendix C Examples from a QStruc - KEE Session

In the following we present the reasoning steps and the results of a sample run of QStruc on a TI Explorer for a continuous beam example.

C1. Schematic Model

We start the qualitative structural analysis by creating a schematic model. In this case, we define instance objects of the beam, terminals, supports and load. Figure C-1 illustrates these objects as instances of the corresponding *Physical. objects* subclasses and of the *Schematic.model* subclass.

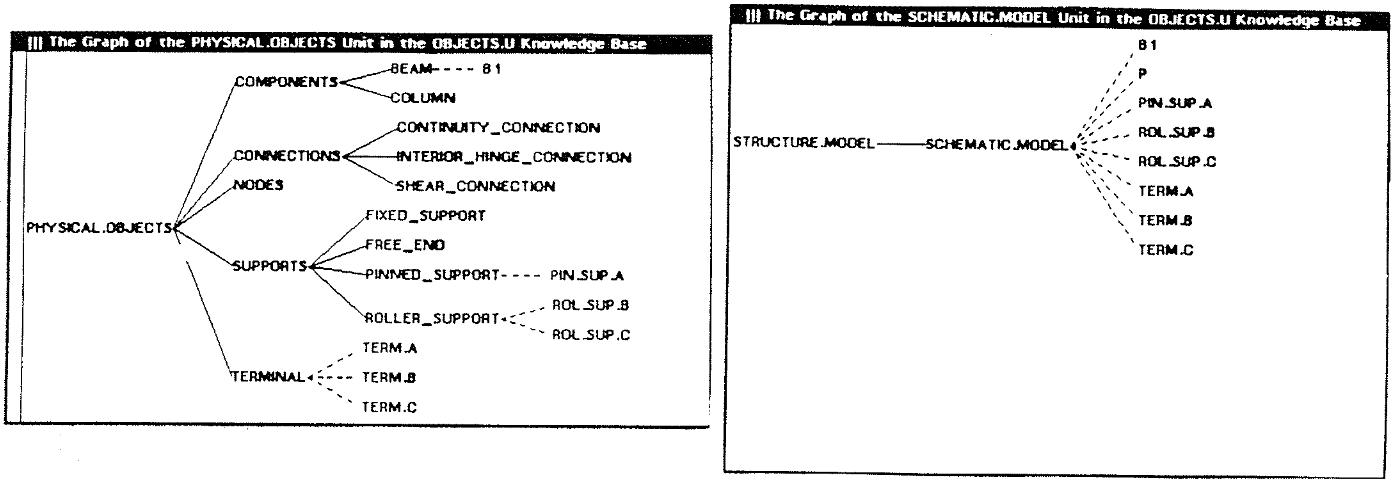


Figure C-1: Schematic Model Creation

C2. General Functions of QStruc

We next run the qualitative structural analysis main function "qstruc".

```
> (qstruc '#%b1)
```

This function sequentially performs the model generation and qualitative analysis steps as follows:

information.transfer	- generates the behavior model
gen.specific.model	- generates the specific model
gen.eqs.rels	- generates the process model

gen-causal-order	- transfers the process model to the causal ordering module and generates the causal ordering for the given structure
cluster-info	- generates the clusters together with the information about: cluster root, list of physical parameters whose qualitative value can be further inferred, and priority of clusters
gen-cl-hierarchy	- generates the hierarchy of clusters in ascending order of priority
identify-dependents	- generates the information necessary to build the causal dependency network among the physical parameters that describe the behavior of the structure
all.tasks	- generates the agenda of tasks from the identified hierarchy of clusters
exec.agenda	- executes the tasks sequentially until the agenda is empty
information.transfer.back	- transfers the inferred qualitative values of the physical parameters from the terminals to the corresponding support objects.

C3. Model Generation

C3.1 Behavior Model

Figure C-2 presents the initial information and the inferred interactions resulted from the transformation of the schematic model into the behavior model. This transformation process has been discussed in Section 4.1. The figure illustrates the values of the slots of the corresponding terminals before and after this transformation.

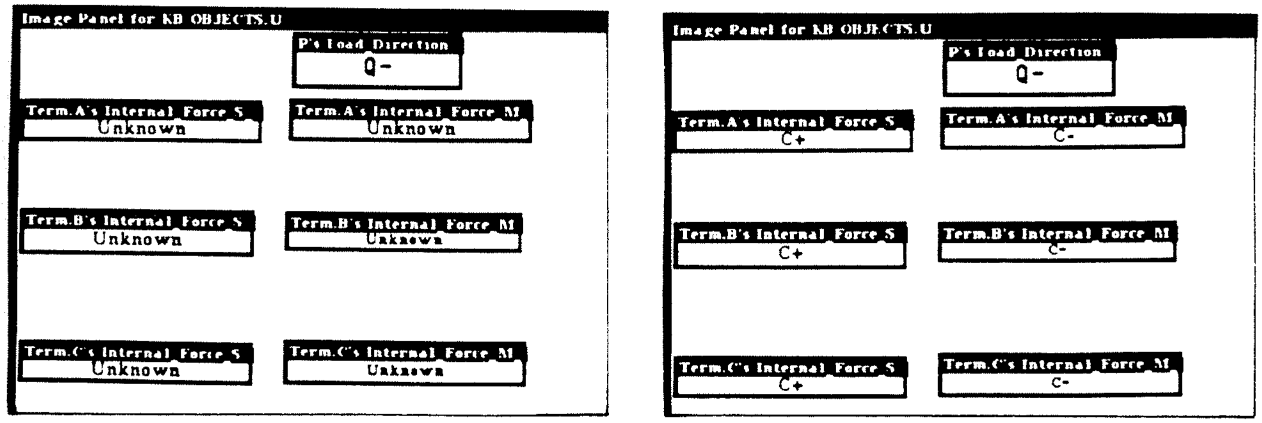


Image Panel for KB OBJECTS.U

P's Load Direction
Q-

Term.A's Internal Force S
Unknown

Term.A's Internal Force M
Unknown

Term.B's Internal Force S
Unknown

Term.B's Internal Force M
Unknown

Term.C's Internal Force S
Unknown

Term.C's Internal Force M
Unknown

Image Panel for KB OBJECTS.U

P's Load Direction
Q-

Term.A's Internal Force S
C+

Term.A's Internal Force M
C-

Term.B's Internal Force S
C+

Term.B's Internal Force M
C-

Term.C's Internal Force S
C+

Term.C's Internal Force M
C-

Figure C-2: Interaction Map Identification (before and after)

C3.2 Specific Model

Next, the system generates the specific model by applying focusing rules and decomposing the initial beam at relevant points: at support location along the component and load application point. The representation of the specific model by new *Physical.component* instances is graphically illustrated in Figure C-3, and the updated representation of the structure is shown in Figure C-4.

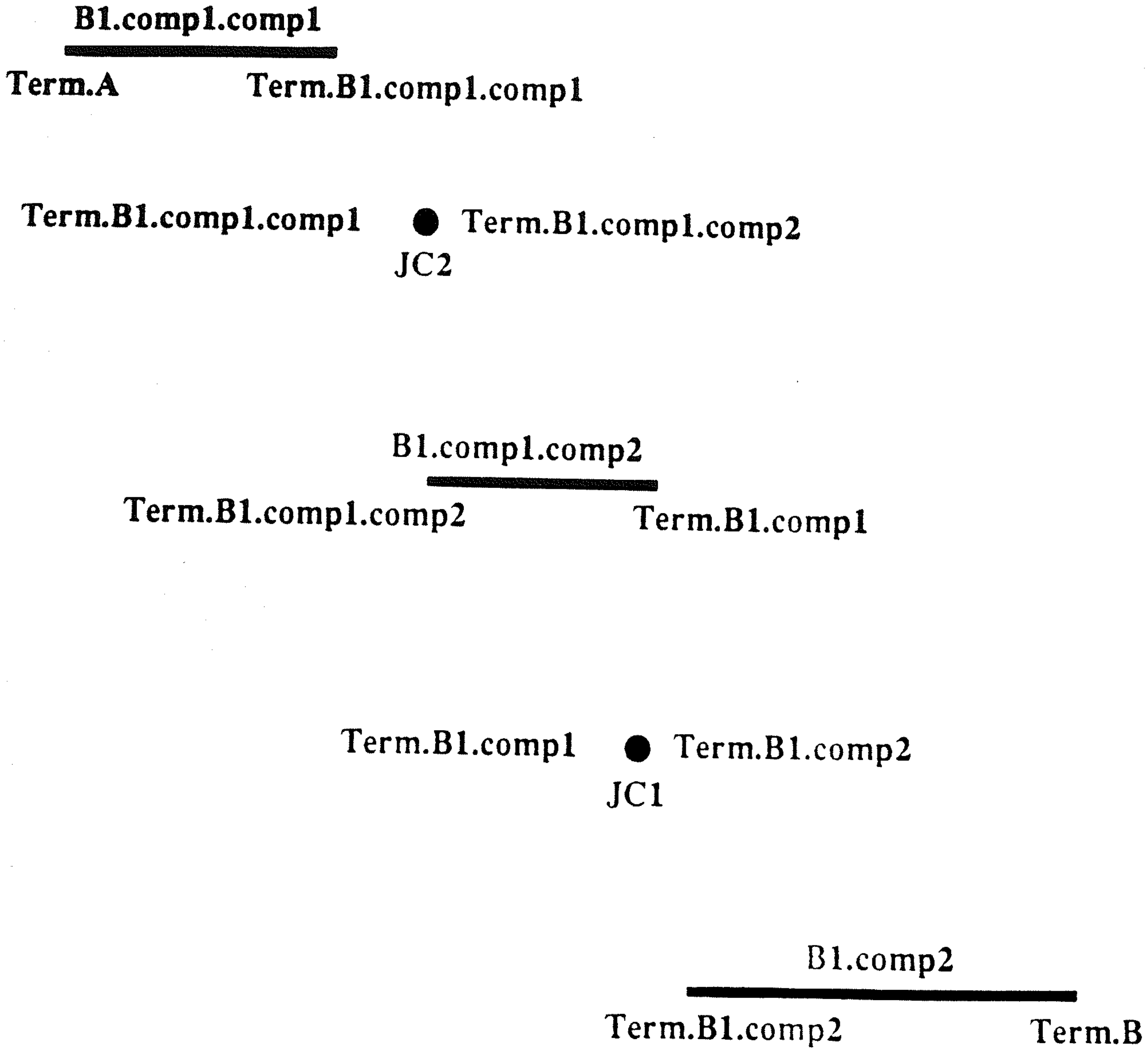


Figure C-3: Specific Model Decomposition

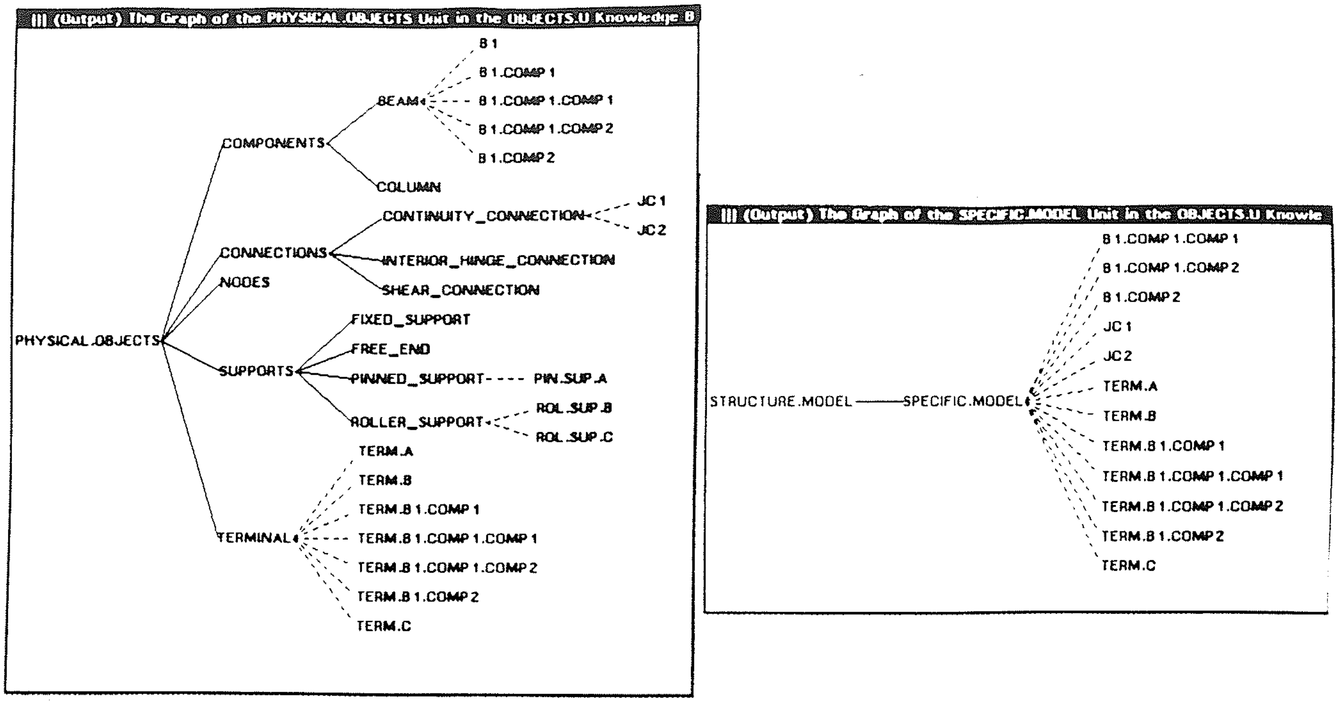


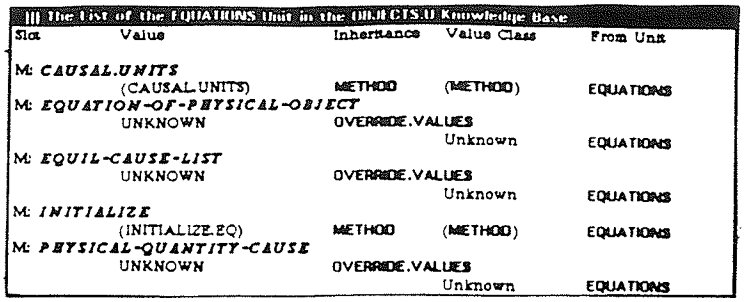
Figure C-4: Updated Specific Model

C3.3 Process Model

The process model is generated for the derived specific model. The instances of applicable *equations, relations* and *physical quantities* are created (see Figure 6 of Section 4.3). Below we further present the information contained in such units. Figure C-5 shows the frame of the class *equations* and gives an example of an instance - *B1.comp1.comp1-Moment.Equil.Eq*. The slots represent the following information that is necessary for further reasoning:

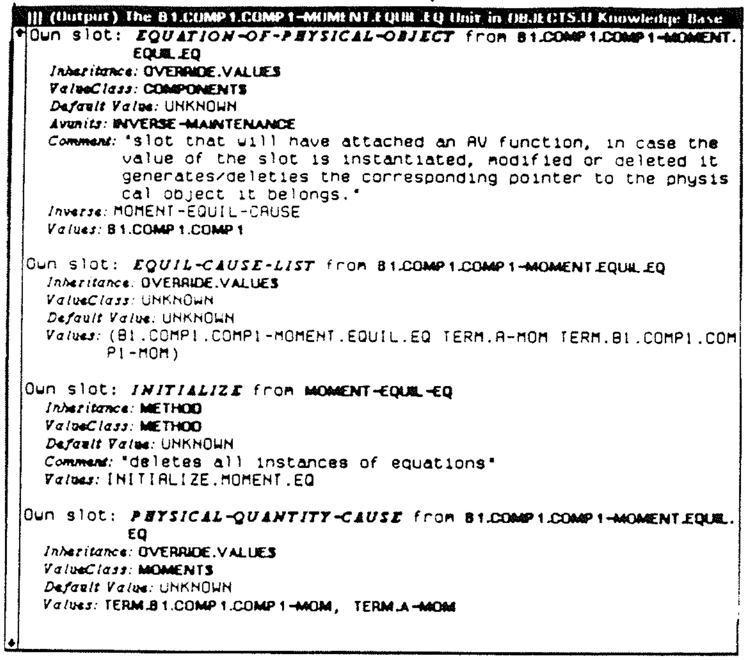
causal.units	- contains the general method for generating the values of the other slots of the units
equation-of-physical-object	- indicates the physical object the specific equation belongs to
equil-cause-list	- represents the list of the information concerning a specific equation, that is used in the causal ordering reasoning step. The arguments of the list consists of the uniquely created name of the equation and the name of the involved physical quantities. As mentioned before, we chose a self-explanatory naming convention that indicates the considered physical object and the applied physical law.
initialize	- deletes the instances of equations in the case the user decides to initialize the process model

Equation



```
||| The List of the EQUATIONS Unit in the OBJECTS.U Knowledge Base
Slot          Value                     Inheritance      Value Class      From Unit

M: CAUSAL.UNITS
              (CAUSAL.UNITS)            METHOD           (METHOD)         EQUATIONS
M: EQUATION-OF-PHYSICAL-OBJECT
              UNKNOWN                   OVERRIDE.VALUES
                                                         Unknown          EQUATIONS
M: EQUIL-CAUSE-LIST
              UNKNOWN                   OVERRIDE.VALUES
                                                         Unknown          EQUATIONS
M: INITIALIZE
              (INITIALIZE.EQ)           METHOD           (METHOD)         EQUATIONS
M: PHYSICAL-QUANTITY-CAUSE
              UNKNOWN                   OVERRIDE.VALUES
                                                         Unknown          EQUATIONS
```

Instance of Equation



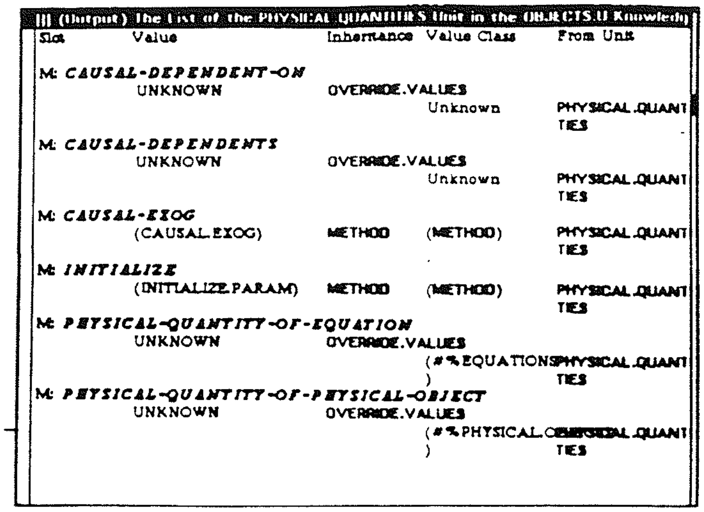
```
||| (Output) The B1.COMP1.COMP1-MOMENT.EQUIL.EQ Unit in OBJECTS.U Knowledge Base
Own slot: EQUATION-OF-PHYSICAL-OBJECT from B1.COMP1.COMP1-MOMENT.
          EQUIL.EQ
  Inheritance: OVERRIDE.VALUES
  ValueClass: COMPONENTS
  Default Value: UNKNOWN
  Avunits: INVERSE-MAINTENANCE
  Comment: "slot that will have attached an AV function, in case the
           value of the slot is instantiated, modified or deleted it
           generates/deleties the corresponding pointer to the physis
           cal object it belongs."
  Inverse: MOMENT-EQUIL-CAUSE
  Values: B1.COMP1.COMP1

Own slot: EQUIL-CAUSE-LIST from B1.COMP1.COMP1-MOMENT.EQUIL.EQ
  Inheritance: OVERRIDE.VALUES
  ValueClass: UNKNOWN
  Default Value: UNKNOWN
  Values: (B1.COMP1.COMP1-MOMENT.EQUIL.EQ TERM.A-MOM TERM.B1.COMP1.COM
          P1-MOM)

Own slot: INITIALIZE from MOMENT-EQUIL-EQ
  Inheritance: METHOD
  ValueClass: METHOD
  Default Value: UNKNOWN
  Comment: "deletes all instances of equations"
  Values: INITIALIZE.MOMENT.EQ

Own slot: PHYSICAL-QUANTITY-CAUSE from B1.COMP1.COMP1-MOMENT.EQUIL.
          EQ
  Inheritance: OVERRIDE.VALUES
  ValueClass: MOMENTS
  Default Value: UNKNOWN
  Values: TERM.B1.COMP1.COMP1-MOM,  TERM.A-MOM
```

Figure C-5: Equation Unit and Instance Example

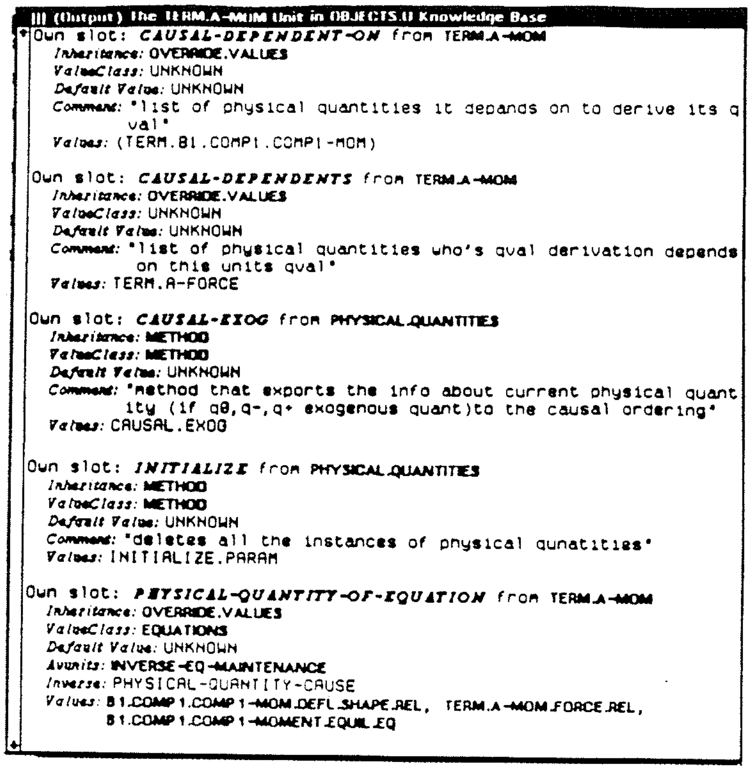
Figure C-6 illustrates the frame of the class *Physical.quantities* and gives an example of an instance - *Term.A-Mom*. The slots contain the following information that is necessary for further reasoning:

causal-dependent-on	- the list of physical quantities it depends on
causal-dependents	- the list of physical quantities that depend on the current physical quantity
causal-exog	- a method that determines whether the physical quantity is known and transfers this information to the causal ordering module
initialize	- a method to delete the instances of physical quantities in the case the user decides to initialize the process model

physical-quantity-of-equation	- the equations/relations the physical quantity is involved in
physical-quantity-of-physical-object	- the physical object the physical quantity belongs to.



```
||| (Output) The List of the PHYSICAL.QUANTITIES Unit in the OBJECTS.U Knowledg
Slot          Value                  Inheritance     Value Class     From Unit

M: CAUSAL-DEPENDENT-ON
              UNKNOWN                OVERRIDE.VALUES
                                                     Unknown         PHYSICAL.QUANTI
                                                                     TIES
M: CAUSAL-DEPENDENTS
              UNKNOWN                OVERRIDE.VALUES
                                                     Unknown         PHYSICAL.QUANTI
                                                                     TIES
M: CAUSAL-EXOG
              (CAUSAL.EXOG)          METHOD          (METHOD)        PHYSICAL.QUANTI
                                                                     TIES
M: INITIALIZE
              (INITIALIZE.PARAM)     METHOD          (METHOD)        PHYSICAL.QUANTI
                                                                     TIES
M: PHYSICAL-QUANTITY-OF-EQUATION
              UNKNOWN                OVERRIDE.VALUES
                                                     (#%EQUATIONSPHYSICAL.QUANTI
                                                     )               TIES
M: PHYSICAL-QUANTITY-OF-PHYSICAL-OBJECT
              UNKNOWN                OVERRIDE.VALUES
                                                     (#%PHYSICAL.OPHYSICAL.QUANTI
                                                     )               TIES
```

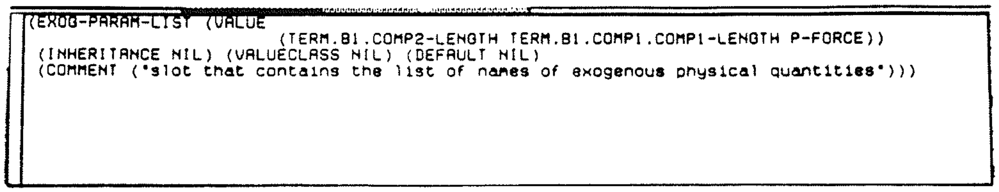


```
||| (Output) The TERM.A-MOM Unit in OBJECTS.U Knowledge Base
Own slot: CAUSAL-DEPENDENT-ON from TERM.A-MOM
   Inheritance: OVERRIDE.VALUES
   ValueClass: UNKNOWN
   Default Value: UNKNOWN
   Comment: "list of physical quantities it depands on to derive its q
          val"
   Values: (TERM.B1.COMP1.COMP1-MOM)

Own slot: CAUSAL-DEPENDENTS from TERM.A-MOM
   Inheritance: OVERRIDE.VALUES
   ValueClass: UNKNOWN
   Default Value: UNKNOWN
   Comment: "list of physical quantities who's qval derivation depends
          on this units qval"
   Values: TERM.A-FORCE

Own slot: CAUSAL-EXOG from PHYSICAL.QUANTITIES
   Inheritance: METHOD
   ValueClass: METHOD
   Default Value: UNKNOWN
   Comment: "method that exports the info about current physical quant
          ity (if q0,q-,q+ exogenous quant)to the causal ordering"
   Values: CAUSAL.EXOG

Own slot: INITIALIZE from PHYSICAL.QUANTITIES
   Inheritance: METHOD
   ValueClass: METHOD
   Default Value: UNKNOWN
   Comment: "deletes all the instances of physical qunatities"
   Values: INITIALIZE.PARAM

Own slot: PHYSICAL-QUANTITY-OF-EQUATION from TERM.A-MOM
   Inheritance: OVERRIDE.VALUES
   ValueClass: EQUATIONS
   Default Value: UNKNOWN
   Avunits: INVERSE-EQ-MAINTENANCE
   Inverse: PHYSICAL-QUANTITY-CAUSE
   Values: B1.COMP1.COMP1-MOM.DEFL.SHAPE.REL, TERM.A-MOM.FORCE.REL,
         B1.COMP1.COMP1-MOMENT.EQUIL.EQ
```

Figure C-6: Physical.Quantities Unit and Instance Example

C4. Qualitative Analysis

Next, the system applies a causal reasoning mechanism to derive the causal ordering among the physical quantities. For this purpose, the system takes the information about the applicable equations and relations representing the process model. This is represented in the form of a list of lists that is stored in the slot *causal-units* of the *Causal-Ordering* object. Each of these lists represents the information stored in the slot *equil-cause-list* of all the equation and relation instances. The *Causal-Ordering* contains the information about the exogenous parameters stored as a list in the slot *exog-param-list* (Figure C-7). These two lists of information - the *causal-units* and the *exog-param-list* - are used by the causal ordering module to infer the causal ordering among the equations, relations, and physical parameters. The method slot that triggers the causal ordering reasoning is *gen-causal-order*.



```
(EXOG-PARAM-LIST (VALUE
                    (TERM.B1.COMP2-LENGTH TERM.B1.COMP1.COMP1-LENGTH P-FORCE))
 (INHERITANCE NIL) (VALUECLASS NIL) (DEFAULT NIL)
 (COMMENT ("slot that contains the list of names of exogenous physical quantities")))
```

Figure C-7: List of Exogenous Physical Quantities

The result of causal ordering is further clustered as a function of physical objects and the priority of each cluster is identified. Figure C-8 illustrates the representation of the information related to the cluster corresponding to JC2. The first two lists show the causal sequence of the equations/relations, the second list represents the unknown physical parameters of the cluster whose qualitative value can eventually be derived, and the last argument represents the priority of the cluster.

```
( ( (JC2-LOAD.DISPL.REL  TERM.B1.COMP1.COMP2-Y.DISPL  P-FORCE)
    (JC2-DISPL.EQUIL.EQ  TERM.B1.COMP1.COMP1-Y.DISPL
                         TERM.B1.COMP1.COMP2-Y.DISPL) )
  (TERM.B1.COMP1.COMP2-Y.DISPL  TERM.B1.COMP1.COMP2-Y.DISPL)
  2)
```

Figure C-8: Cluster Information Example

Next, the system generates the cluster hierarchy, identifies the causal dependencies among the physical quantities and generates the agenda of tasks, as discussed and illustrated in Section 5. Note, that the user has the possibility to inspect specific physical quantities, their dependents and/or the physical quantities that it depends on. This may be a useful task in the decision process involving a change in the physical quantity under consideration. Figure C-9 shows the command and the result for obtaining the dependents of the physical quantity *Term.B1.comp2-Mom*.

```
>(graph-dependents '#%term.b1.comp2-mom)
```

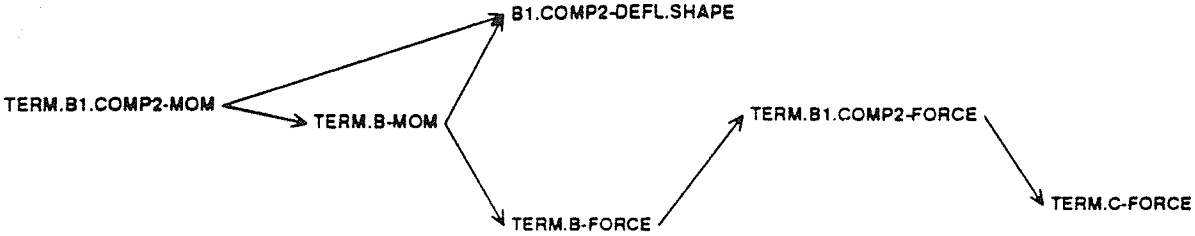
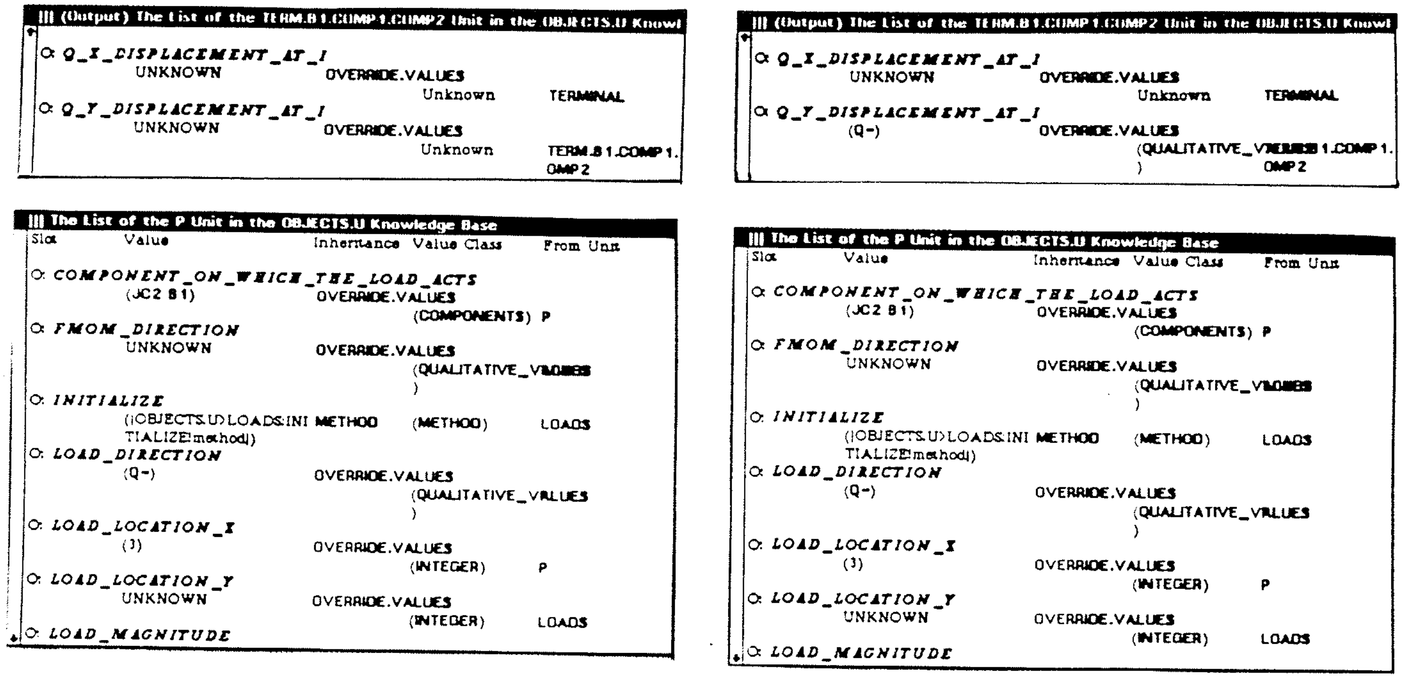


Figure C-9: Inspecting a Specific Physical Quantity

In the following we illustrate the results of the qualitative evaluation step obtained by executing the tasks in the agenda presented in Figure 8, Section 5.5.

In executing the task JC2-Load.Displ.Rel the system sends a message to the method slot *Load-displ-qeval* of the *JC2* unit, to infer the qualitative value *Q_Y_displacement_at_i* of *Term.B1.comp1.comp2* using the known value *Load_direction Q-* of *P* (Figure C-10).



```
||| (Output) The List of the TERM.B1.COMP1.COMP2 Unit in the OBJECTS.U Knowl
○ Q_X_DISPLACEMENT_AT_I
          UNKNOWN               OVERRIDE.VALUES
                                          Unknown        TERMINAL
○ Q_Y_DISPLACEMENT_AT_I
          UNKNOWN               OVERRIDE.VALUES
                                          Unknown        TERM.B1.COMP1.
                                                         OMP2
```

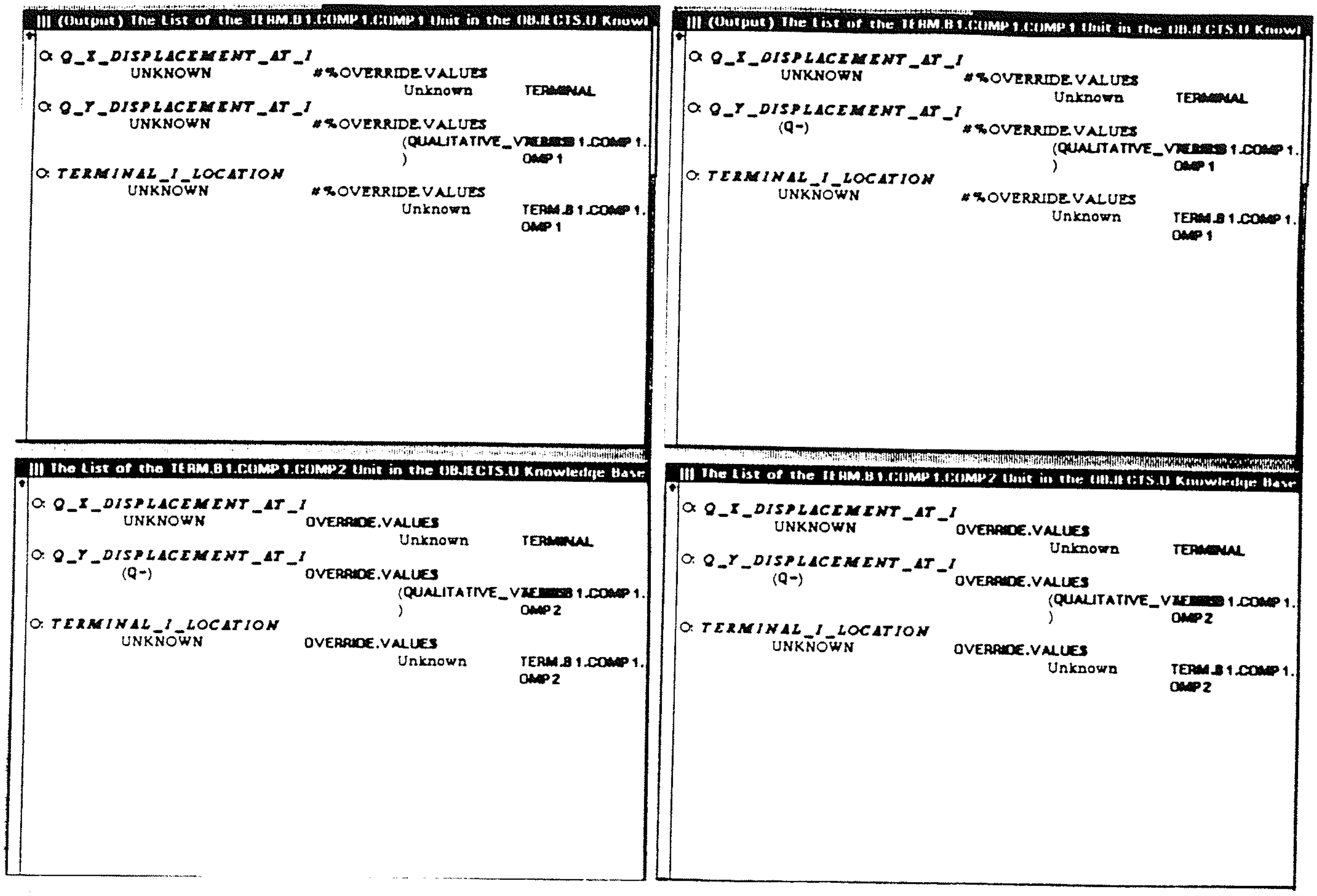
```
||| (Output) The List of the TERM.B1.COMP1.COMP2 Unit in the OBJECTS.U Knowl
○ Q_X_DISPLACEMENT_AT_I
          UNKNOWN               OVERRIDE.VALUES
                                          Unknown        TERMINAL
○ Q_Y_DISPLACEMENT_AT_I
          (Q-)                  OVERRIDE.VALUES
                                          (QUALITATIVE_V[illegible]1.COMP1.
                                          )              OMP2
```

```
||| The List of the P Unit in the OBJECTS.U Knowledge Base
Slot      Value                 Inheritance  Value Class     From Unit

○ COMPONENT_ON_WHICH_THE_LOAD_ACTS
          (JC2 B1)              OVERRIDE.VALUES
                                             (COMPONENTS)  P
○ FMOM_DIRECTION
          UNKNOWN               OVERRIDE.VALUES
                                             (QUALITATIVE_V[illegible]
                                             )
○ INITIALIZE
          (|OBJECTS.U>LOADS:INI METHOD       (METHOD)        LOADS
          TIALIZE!method|)
○ LOAD_DIRECTION
          (Q-)                  OVERRIDE.VALUES
                                             (QUALITATIVE_VALUES
                                             )
○ LOAD_LOCATION_X
          (3)                   OVERRIDE.VALUES
                                             (INTEGER)       P
○ LOAD_LOCATION_Y
          UNKNOWN               OVERRIDE.VALUES
                                             (INTEGER)       LOADS
○ LOAD_MAGNITUDE
```

```
||| The List of the P Unit in the OBJECTS.U Knowledge Base
Slot      Value                 Inheritance  Value Class     From Unit

○ COMPONENT_ON_WHICH_THE_LOAD_ACTS
          (JC2 B1)              OVERRIDE.VALUES
                                             (COMPONENTS)  P
○ FMOM_DIRECTION
          UNKNOWN               OVERRIDE.VALUES
                                             (QUALITATIVE_V[illegible]
                                             )
○ INITIALIZE
          (|OBJECTS.U>LOADS:INI METHOD       (METHOD)        LOADS
          TIALIZE!method|)
○ LOAD_DIRECTION
          (Q-)                  OVERRIDE.VALUES
                                             (QUALITATIVE_VALUES
                                             )
○ LOAD_LOCATION_X
          (3)                   OVERRIDE.VALUES
                                             (INTEGER)       P
○ LOAD_LOCATION_Y
          UNKNOWN               OVERRIDE.VALUES
                                             (INTEGER)       LOADS
○ LOAD_MAGNITUDE
```

Figure C-10: Result of Executing JC2-Load.Displ.Rel Task

In executing the task JC2-Displ.Equil.Eq the system sends a message to the method slot *Defl-equil-qeval* of the *JC2* unit, to infer the qualitative value *Q_Y_displacement_at_i* of *Term.B1.comp1.comp1* using the known qualitative value *Q_Y_displacement_at_i* of *Term.B1.comp1.comp2* (Figure C-11).



```
||| (Output) The List of the TERM.B1.COMP1.COMP1 Unit in the OBJECTS.U Knowl
O: Q_X_DISPLACEMENT_AT_I
        UNKNOWN               #%OVERRIDE.VALUES
                                        Unknown        TERMINAL
O: Q_Y_DISPLACEMENT_AT_I
        UNKNOWN               #%OVERRIDE.VALUES
                                        (QUALITATIVE_V[illegible] 1.COMP 1.
                                        )              OMP 1
O: TERMINAL_I_LOCATION
        UNKNOWN               #%OVERRIDE.VALUES
                                        Unknown        TERM.B 1.COMP 1.
                                                       OMP 1
```

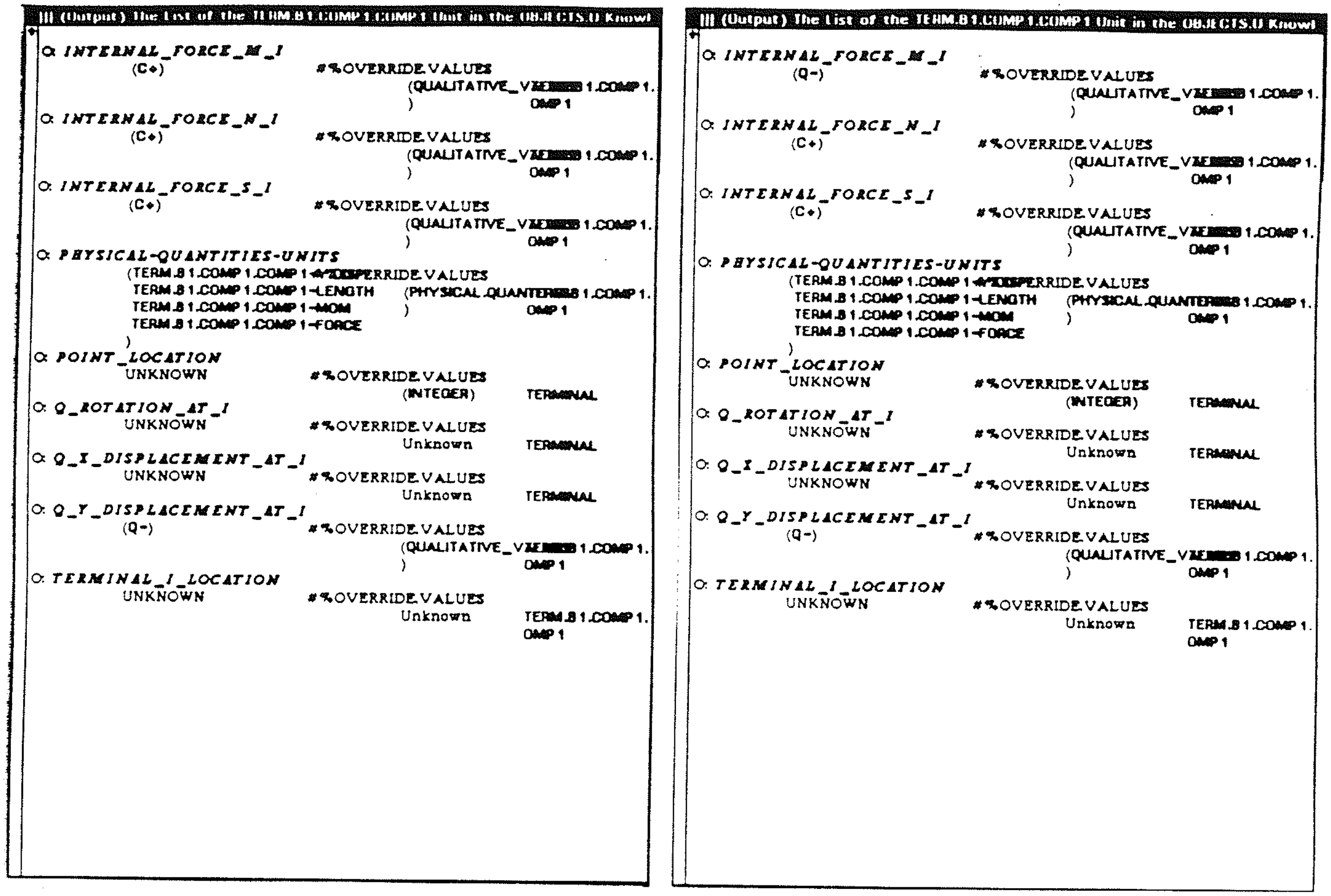
```
||| (Output) The List of the TERM.B1.COMP1.COMP1 Unit in the OBJECTS.U Knowl
O: Q_X_DISPLACEMENT_AT_I
        UNKNOWN               #%OVERRIDE.VALUES
                                        Unknown        TERMINAL
O: Q_Y_DISPLACEMENT_AT_I
        (Q-)                  #%OVERRIDE.VALUES
                                        (QUALITATIVE_V[illegible] 1.COMP 1.
                                        )              OMP 1
O: TERMINAL_I_LOCATION
        UNKNOWN               #%OVERRIDE.VALUES
                                        Unknown        TERM.B 1.COMP 1.
                                                       OMP 1
```

```
||| The List of the TERM.B1.COMP1.COMP2 Unit in the OBJECTS.U Knowledge Base
O: Q_X_DISPLACEMENT_AT_I
        UNKNOWN               OVERRIDE.VALUES
                                        Unknown        TERMINAL
O: Q_Y_DISPLACEMENT_AT_I
        (Q-)                  OVERRIDE.VALUES
                                        (QUALITATIVE_V[illegible] 1.COMP 1.
                                        )              OMP 2
O: TERMINAL_I_LOCATION
        UNKNOWN               OVERRIDE.VALUES
                                        Unknown        TERM.B 1.COMP 1.
                                                       OMP 2
```

```
||| The List of the TERM.B1.COMP1.COMP2 Unit in the OBJECTS.U Knowledge Base
O: Q_X_DISPLACEMENT_AT_I
        UNKNOWN               OVERRIDE.VALUES
                                        Unknown        TERMINAL
O: Q_Y_DISPLACEMENT_AT_I
        (Q-)                  OVERRIDE.VALUES
                                        (QUALITATIVE_V[illegible] 1.COMP 1.
                                        )              OMP 2
O: TERMINAL_I_LOCATION
        UNKNOWN               OVERRIDE.VALUES
                                        Unknown        TERM.B 1.COMP 1.
                                                       OMP 2
```

Figure C-11: Result of Executing JC2-Displ.Equil.Eq Task

In executing the task B1.comp1.comp1-Displ.Mom.Rel.I the system sends a message to the method slot *Mom-displ-i-qeval* to infer the qualitative value *Internal_force_M_i* of *Term.B1.comp1.comp1* using the known qualitative value *Q_Y_displacement_at_i* at the continuity connection (Figure C-12).

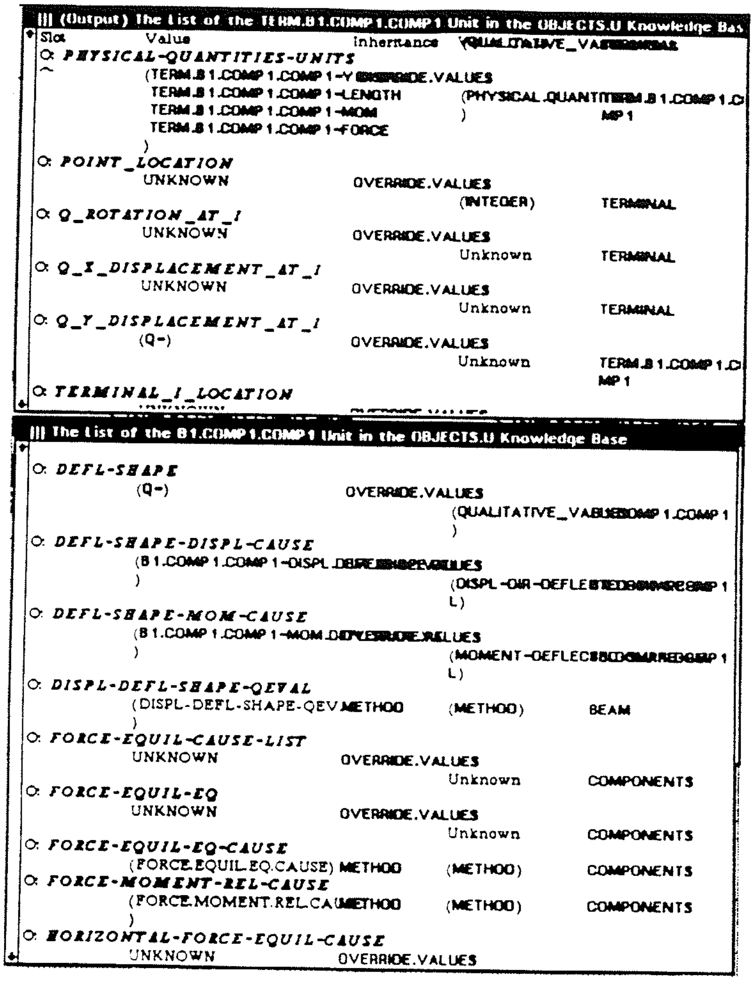


```
||| (Output) The List of the TERM.B1.COMP1.COMP1 Unit in the OBJECTS.U Knowl
O: INTERNAL_FORCE_M_I
        (C+)                    #%OVERRIDE.VALUES
                                        (QUALITATIVE_VTERM.B1.COMP1.
                                        )               OMP1
O: INTERNAL_FORCE_N_I
        (C+)                    #%OVERRIDE.VALUES
                                        (QUALITATIVE_VTERM.B1.COMP1.
                                        )               OMP1
O: INTERNAL_FORCE_S_I
        (C+)                    #%OVERRIDE.VALUES
                                        (QUALITATIVE_VTERM.B1.COMP1.
                                        )               OMP1
O: PHYSICAL-QUANTITIES-UNITS
        (TERM.B1.COMP1.COMP1-#%OVERRIDE.VALUES
         TERM.B1.COMP1.COMP1-LENGTH     (PHYSICAL.QUANTERM.B1.COMP1.
         TERM.B1.COMP1.COMP1-MOM        )               OMP1
         TERM.B1.COMP1.COMP1-FORCE
        )
O: POINT_LOCATION
        UNKNOWN                 #%OVERRIDE.VALUES
                                        (INTEGER)       TERMINAL
O: Q_ROTATION_AT_I
        UNKNOWN                 #%OVERRIDE.VALUES
                                        Unknown         TERMINAL
O: Q_X_DISPLACEMENT_AT_I
        UNKNOWN                 #%OVERRIDE.VALUES
                                        Unknown         TERMINAL
O: Q_Y_DISPLACEMENT_AT_I
        (Q-)                    #%OVERRIDE.VALUES
                                        (QUALITATIVE_VTERM.B1.COMP1.
                                        )               OMP1
O: TERMINAL_I_LOCATION
        UNKNOWN                 #%OVERRIDE.VALUES
                                        Unknown         TERM.B1.COMP1.
                                                        OMP1
```

```
||| (Output) The List of the TERM.B1.COMP1.COMP1 Unit in the OBJECTS.U Knowl
O: INTERNAL_FORCE_M_I
        (Q-)                    #%OVERRIDE.VALUES
                                        (QUALITATIVE_VTERM.B1.COMP1.
                                        )               OMP1
O: INTERNAL_FORCE_N_I
        (C+)                    #%OVERRIDE.VALUES
                                        (QUALITATIVE_VTERM.B1.COMP1.
                                        )               OMP1
O: INTERNAL_FORCE_S_I
        (C+)                    #%OVERRIDE.VALUES
                                        (QUALITATIVE_VTERM.B1.COMP1.
                                        )               OMP1
O: PHYSICAL-QUANTITIES-UNITS
        (TERM.B1.COMP1.COMP1-#%OVERRIDE.VALUES
         TERM.B1.COMP1.COMP1-LENGTH     (PHYSICAL.QUANTERM.B1.COMP1.
         TERM.B1.COMP1.COMP1-MOM        )               OMP1
         TERM.B1.COMP1.COMP1-FORCE
        )
O: POINT_LOCATION
        UNKNOWN                 #%OVERRIDE.VALUES
                                        (INTEGER)       TERMINAL
O: Q_ROTATION_AT_I
        UNKNOWN                 #%OVERRIDE.VALUES
                                        Unknown         TERMINAL
O: Q_X_DISPLACEMENT_AT_I
        UNKNOWN                 #%OVERRIDE.VALUES
                                        Unknown         TERMINAL
O: Q_Y_DISPLACEMENT_AT_I
        (Q-)                    #%OVERRIDE.VALUES
                                        (QUALITATIVE_VTERM.B1.COMP1.
                                        )               OMP1
O: TERMINAL_I_LOCATION
        UNKNOWN                 #%OVERRIDE.VALUES
                                        Unknown         TERM.B1.COMP1.
                                                        OMP1
```

Figure C-12: Result of Executing B1.comp1.comp1-Displ.Mom.Rel.I Task

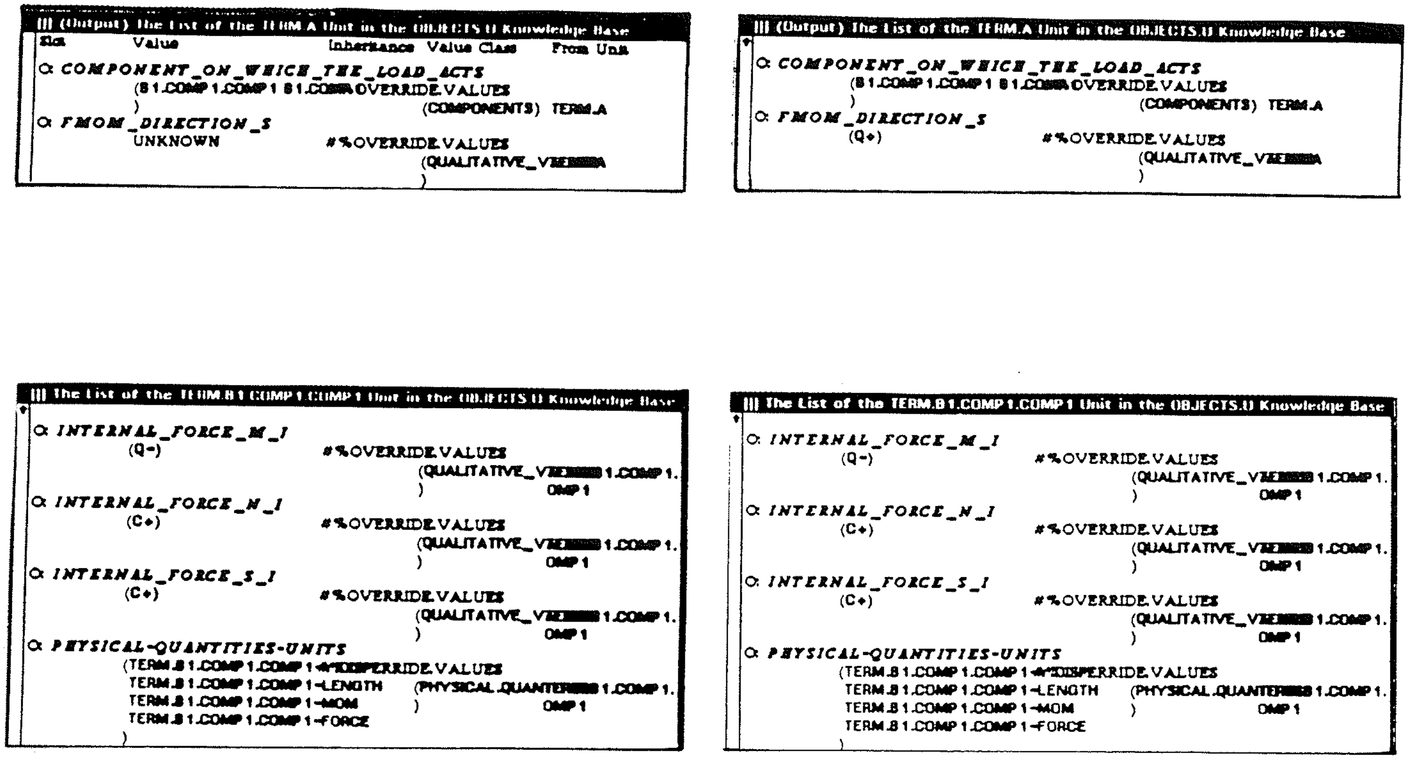
In executing the task B1.comp1.comp1-Displ.Defl.Shape.Rel the system sends a message to the method slot *Displ-defl-shape-qeval* to infer the qualitative value *Defl-Shape* of *B1.comp1.comp1* using the known qualitative value *Q_Y_displacement_at_i* of *Term.B1.comp1.comp1* and the support conditions (Figure C-13).



```
||| (Output) The List of the TERM.B1.COMP1.COMP1 Unit in the OBJECTS.U Knowledge Bas
Slot         Value                     Inheritance   QUALITATIVE_VA[illegible]
○ PHYSICAL-QUANTITIES-UNITS
         (TERM.B1.COMP1.COMP1-Y OVERRIDE.VALUES
          TERM.B1.COMP1.COMP1-LENGTH        (PHYSICAL.QUANTITERM.B1.COMP1.C
          TERM.B1.COMP1.COMP1-MOM           )                 MP1
          TERM.B1.COMP1.COMP1-FORCE
         )
○ POINT_LOCATION
         UNKNOWN                 OVERRIDE.VALUES
                                           (INTEGER)          TERMINAL
○ Q_ROTATION_AT_I
         UNKNOWN                 OVERRIDE.VALUES
                                           Unknown            TERMINAL
○ Q_X_DISPLACEMENT_AT_I
         UNKNOWN                 OVERRIDE.VALUES
                                           Unknown            TERMINAL
○ Q_Y_DISPLACEMENT_AT_I
         (Q-)                    OVERRIDE.VALUES
                                           Unknown            TERM.B1.COMP1.C
                                                              MP1
○ TERMINAL_I_LOCATION

||| The List of the B1.COMP1.COMP1 Unit in the OBJECTS.U Knowledge Base
○ DEFL-SHAPE
         (Q-)                    OVERRIDE.VALUES
                                           (QUALITATIVE_VA[illegible]COMP1.COMP1
                                           )
○ DEFL-SHAPE-DISPL-CAUSE
         (B1.COMP1.COMP1-DISPL.[illegible]VALUES
         )                                 (DISPL-DIR-DEFLE[illegible]1
                                           L)
○ DEFL-SHAPE-MOM-CAUSE
         (B1.COMP1.COMP1-MOM.[illegible]VALUES
         )                                 (MOMENT-DEFLEC[illegible]1
                                           L)
○ DISPL-DEFL-SHAPE-QEVAL
         (DISPL-DEFL-SHAPE-QEV METHOD      (METHOD)           BEAM
         )
○ FORCE-EQUIL-CAUSE-LIST
         UNKNOWN                 OVERRIDE.VALUES
                                           Unknown            COMPONENTS
○ FORCE-EQUIL-EQ
         UNKNOWN                 OVERRIDE.VALUES
                                           Unknown            COMPONENTS
○ FORCE-EQUIL-EQ-CAUSE
         (FORCE.EQUIL.EQ.CAUSE) METHOD     (METHOD)           COMPONENTS
○ FORCE-MOMENT-REL-CAUSE
         (FORCE.MOMENT.REL.CAUMETHOD       (METHOD)           COMPONENTS
         )
○ HORIZONTAL-FORCE-EQUIL-CAUSE
         UNKNOWN                 OVERRIDE.VALUES
```

Figure C-13: Result of Executing B1.comp1.comp1-Displ.Defl.Shape.Rel Task

In executing the task B1.comp1.comp1-Mom.Equil.Eq the system sends a message to the method slot *Balancing-Mom-qeval* to infer the qualitative value *FMom_direction_S* (that is the moment of the unknown vertical force at *Term.A* of *B1.comp1.comp1)* using the known qualitative value *Internal_force_M_i* of *Term.B1.comp1.comp1* (Figure C-14).



```
||| (Output) The List of the TERM.A Unit in the OBJECTS.U Knowledge Base
Slot      Value                    Inheritance  Value Class     From Unit
○ COMPONENT_ON_WHICH_THE_LOAD_ACTS
          (B1.COMP1.COMP1 B1.COMP OVERRIDE.VALUES
          )                                     (COMPONENTS)    TERM.A
○ FMOM_DIRECTION_S
          UNKNOWN                  #%OVERRIDE.VALUES
                                                (QUALITATIVE_V
                                                )
```

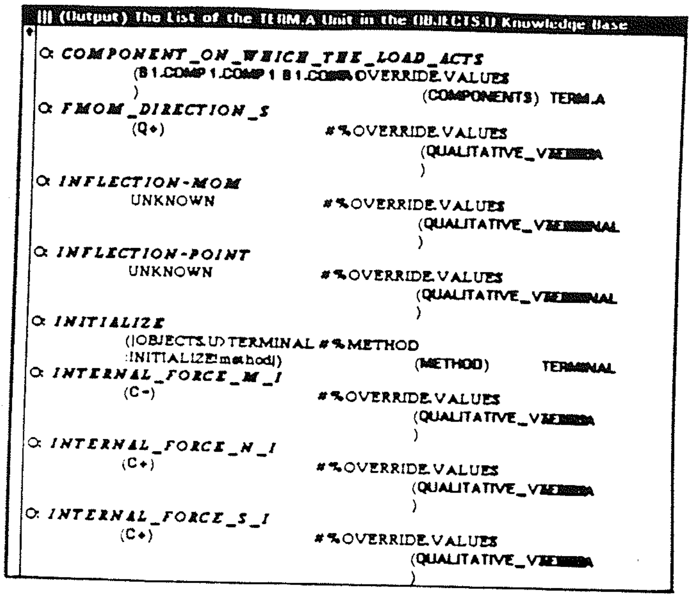
```
||| (Output) The List of the TERM.A Unit in the OBJECTS.U Knowledge Base
○ COMPONENT_ON_WHICH_THE_LOAD_ACTS
          (B1.COMP1.COMP1 B1.COMP OVERRIDE.VALUES
          )                                     (COMPONENTS)    TERM.A
○ FMOM_DIRECTION_S
          (Q+)                     #%OVERRIDE.VALUES
                                                (QUALITATIVE_V
                                                )
```

```
||| The List of the TERM.B1.COMP1.COMP1 Unit in the OBJECTS.U Knowledge Base
○ INTERNAL_FORCE_M_I
          (Q-)                     #%OVERRIDE.VALUES
                                        (QUALITATIVE_V      1.COMP1.
                                        )                   OMP1
○ INTERNAL_FORCE_N_I
          (C+)                     #%OVERRIDE.VALUES
                                        (QUALITATIVE_V      1.COMP1.
                                        )                   OMP1
○ INTERNAL_FORCE_S_I
          (C+)                     #%OVERRIDE.VALUES
                                        (QUALITATIVE_V      1.COMP1.
                                        )                   OMP1
○ PHYSICAL-QUANTITIES-UNITS
          (TERM.B1.COMP1.COMP1-#%OVERRIDE.VALUES
           TERM.B1.COMP1.COMP1-LENGTH   (PHYSICAL.QUANTE    1.COMP1.
           TERM.B1.COMP1.COMP1-MOM      )                   OMP1
           TERM.B1.COMP1.COMP1-FORCE
          )
```

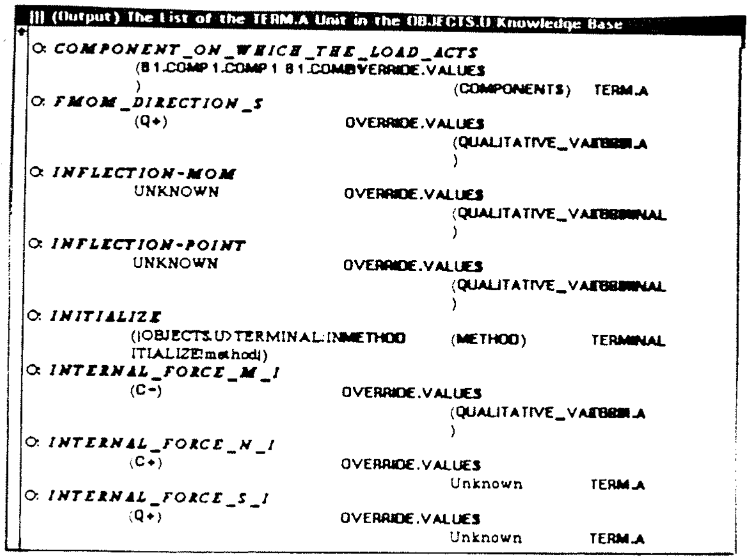
```
||| The List of the TERM.B1.COMP1.COMP1 Unit in the OBJECTS.U Knowledge Base
○ INTERNAL_FORCE_M_I
          (Q-)                     #%OVERRIDE.VALUES
                                        (QUALITATIVE_V      1.COMP1.
                                        )                   OMP1
○ INTERNAL_FORCE_N_I
          (C+)                     #%OVERRIDE.VALUES
                                        (QUALITATIVE_V      1.COMP1.
                                        )                   OMP1
○ INTERNAL_FORCE_S_I
          (C+)                     #%OVERRIDE.VALUES
                                        (QUALITATIVE_V      1.COMP1.
                                        )                   OMP1
○ PHYSICAL-QUANTITIES-UNITS
          (TERM.B1.COMP1.COMP1-#%OVERRIDE.VALUES
           TERM.B1.COMP1.COMP1-LENGTH   (PHYSICAL.QUANTE    1.COMP1.
           TERM.B1.COMP1.COMP1-MOM      )                   OMP1
           TERM.B1.COMP1.COMP1-FORCE
          )
```

Figure C-14: Result of Executing B1.comp1.comp1-Mom.Equil.Eq Task

In executing the task B1.comp1.comp1-Mom.Force.Rel the system sends a message to the method slot *Calc.S.Force* to infer the qualitative value *Internal_force_S_i* of *Term.A* using the known qualitative value of *FMom_direction_S* and the *Point_location* with respect to which the unknown force generates this moment (Figure C-15).



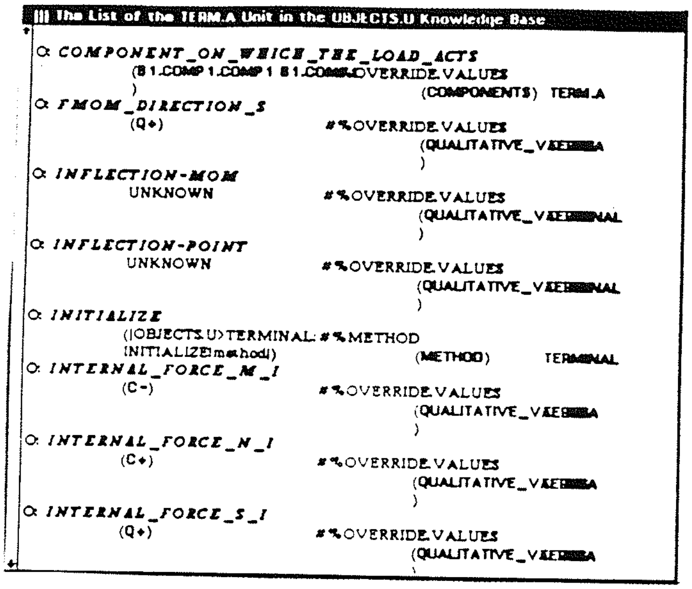
```
||| (Output) The List of the TERM.A Unit in the OBJECTS.U Knowledge Base
○ COMPONENT_ON_WHICH_THE_LOAD_ACTS
        (B1.COMP1.COMP1 B1.COM OVERRIDE.VALUES
        )                                  (COMPONENTS)  TERM.A
○ FMOM_DIRECTION_S
        (Q+)                  #%OVERRIDE.VALUES
                                   (QUALITATIVE_V  A
                                   )
○ INFLECTION-MOM
        UNKNOWN               #%OVERRIDE.VALUES
                                   (QUALITATIVE_V  NAL
                                   )
○ INFLECTION-POINT
        UNKNOWN               #%OVERRIDE.VALUES
                                   (QUALITATIVE_V  NAL
                                   )
○ INITIALIZE
        (|OBJECTS.U>TERMINAL #%METHOD
        :INITIALIZE!method|)               (METHOD)      TERMINAL
○ INTERNAL_FORCE_M_I
        (C-)                  #%OVERRIDE.VALUES
                                   (QUALITATIVE_V  A
                                   )
○ INTERNAL_FORCE_N_I
        (C+)                  #%OVERRIDE.VALUES
                                   (QUALITATIVE_V  A
                                   )
○ INTERNAL_FORCE_S_I
        (C+)                  #%OVERRIDE.VALUES
                                   (QUALITATIVE_V  A
                                   )
```



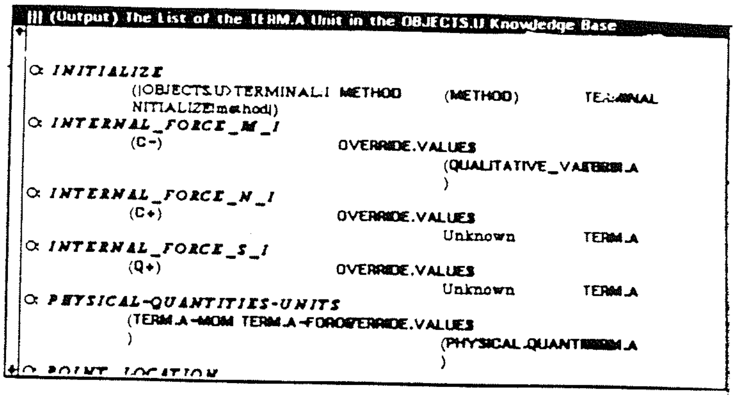
```
||| (Output) The List of the TERM.A Unit in the OBJECTS.U Knowledge Base
○ COMPONENT_ON_WHICH_THE_LOAD_ACTS
        (B1.COMP1.COMP1 B1.COMBVERRIDE.VALUES
        )                                  (COMPONENTS)   TERM.A
○ FMOM_DIRECTION_S
        (Q+)                  OVERRIDE.VALUES
                                   (QUALITATIVE_VA  .A
                                   )
○ INFLECTION-MOM
        UNKNOWN               OVERRIDE.VALUES
                                   (QUALITATIVE_VA  NAL
                                   )
○ INFLECTION-POINT
        UNKNOWN               OVERRIDE.VALUES
                                   (QUALITATIVE_VA  NAL
                                   )
○ INITIALIZE
        (|OBJECTS.U>TERMINAL:INMETHOD      (METHOD)       TERMINAL
        ITIALIZE!method|)
○ INTERNAL_FORCE_M_I
        (C-)                  OVERRIDE.VALUES
                                   (QUALITATIVE_VA  .A
                                   )
○ INTERNAL_FORCE_N_I
        (C+)                  OVERRIDE.VALUES
                                   Unknown        TERM.A
○ INTERNAL_FORCE_S_I
        (Q+)                  OVERRIDE.VALUES
                                   Unknown        TERM.A
```

Figure C-15: Result of Executing B1.comp1.comp1-Mom.Force.Rel Task

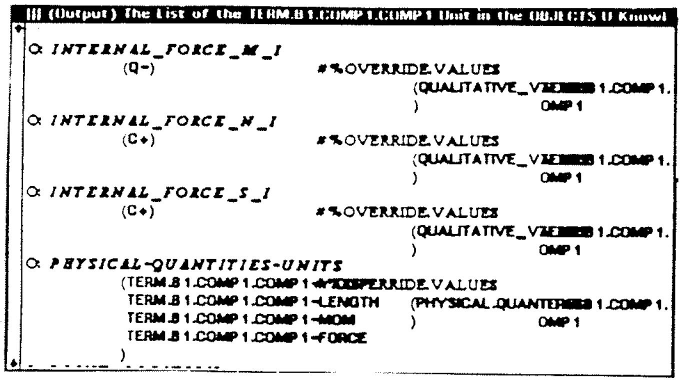
In executing the task B1.comp1.comp1-Force.Equil.Eq the system sends a message to the method slot *Balancing-Force-qeval* to infer the qualitative value *Internal_force_S_i* of *Term.B1.comp1.comp1* using the known qualitative value *Internal_force_S_i* of *Term.A* (Figure C-16).



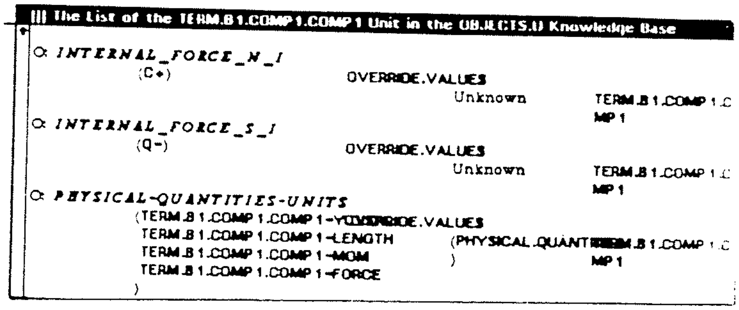
```
||| The List of the TERM.A Unit in the OBJECTS.U Knowledge Base
○ COMPONENT_ON_WHICH_THE_LOAD_ACTS
        (B1.COMP1.COMP1 B1.COM[illegible]OVERRIDE.VALUES
        )                                   (COMPONENTS)  TERM.A
○ FMOM_DIRECTION_S
        (Q+)                    #%OVERRIDE.VALUES
                                    (QUALITATIVE_V[illegible]A
                                    )
○ INFLECTION-MOM
        UNKNOWN                 #%OVERRIDE.VALUES
                                    (QUALITATIVE_V[illegible]NAL
                                    )
○ INFLECTION-POINT
        UNKNOWN                 #%OVERRIDE.VALUES
                                    (QUALITATIVE_V[illegible]NAL
                                    )
○ INITIALIZE
        (|OBJECTS.U>TERMINAL:#%METHOD
        INITIALIZE!method|)                 (METHOD)      TERMINAL
○ INTERNAL_FORCE_M_I
        (C-)                    #%OVERRIDE.VALUES
                                    (QUALITATIVE_V[illegible]A
                                    )
○ INTERNAL_FORCE_N_I
        (C+)                    #%OVERRIDE.VALUES
                                    (QUALITATIVE_V[illegible]A
                                    )
○ INTERNAL_FORCE_S_I
        (Q+)                    #%OVERRIDE.VALUES
                                    (QUALITATIVE_V[illegible]A
```



```
||| (Output) The List of the TERM.A Unit in the OBJECTS.U Knowledge Base
○ INITIALIZE
        (|OBJECTS.U>TERMINAL:I METHOD     (METHOD)        TERMINAL
        NITIALIZE!method|)
○ INTERNAL_FORCE_M_I
        (C-)                  OVERRIDE.VALUES
                                  (QUALITATIVE_VA[illegible].A
                                  )
○ INTERNAL_FORCE_N_I
        (C+)                  OVERRIDE.VALUES
                                  Unknown         TERM.A
○ INTERNAL_FORCE_S_I
        (Q+)                  OVERRIDE.VALUES
                                  Unknown         TERM.A
○ PHYSICAL-QUANTITIES-UNITS
        (TERM.A-MOM TERM.A-FOROVERRIDE.VALUES
        )                         (PHYSICAL.QUANT[illegible].A
                                  )
○ POINT_LOCATION
```



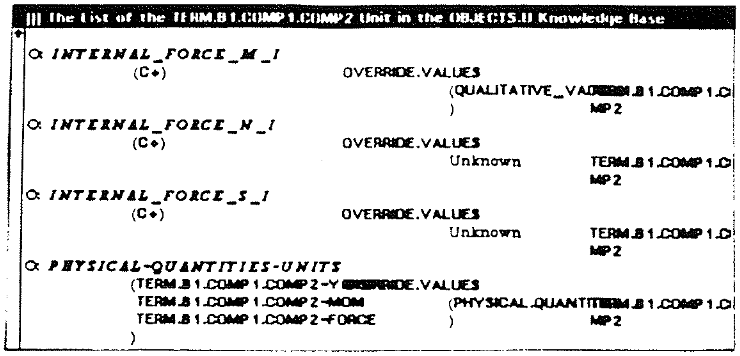
```
||| (Output) The List of the TERM.B1.COMP1.COMP1 Unit in the OBJECTS.U Knowl
○ INTERNAL_FORCE_M_I
        (Q-)                #%OVERRIDE.VALUES
                                (QUALITATIVE_V[illegible]1.COMP1.
                                )               OMP1
○ INTERNAL_FORCE_N_I
        (C+)                #%OVERRIDE.VALUES
                                (QUALITATIVE_V[illegible]1.COMP1.
                                )               OMP1
○ INTERNAL_FORCE_S_I
        (C+)                #%OVERRIDE.VALUES
                                (QUALITATIVE_V[illegible]1.COMP1.
                                )               OMP1
○ PHYSICAL-QUANTITIES-UNITS
        (TERM.B1.COMP1.COMP1-[illegible]ERRIDE.VALUES
         TERM.B1.COMP1.COMP1-LENGTH     (PHYSICAL.QUANTER[illegible]1.COMP1.
         TERM.B1.COMP1.COMP1-MOM        )               OMP1
         TERM.B1.COMP1.COMP1-FORCE
        )
```



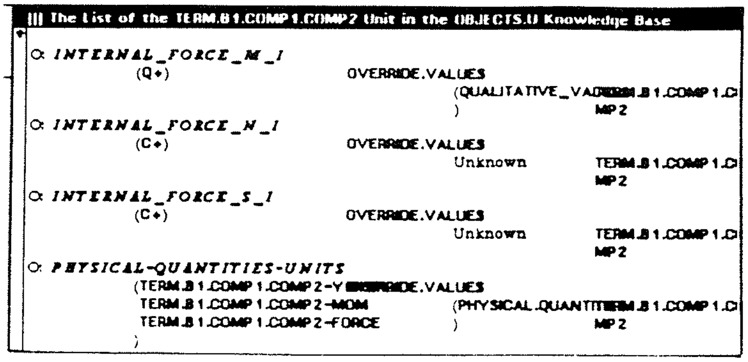
```
||| The List of the TERM.B1.COMP1.COMP1 Unit in the OBJECTS.U Knowledge Base
○ INTERNAL_FORCE_N_I
        (C+)                  OVERRIDE.VALUES
                                  Unknown         TERM.B1.COMP1.C
                                                  MP1
○ INTERNAL_FORCE_S_I
        (Q-)                  OVERRIDE.VALUES
                                  Unknown         TERM.B1.COMP1.C
                                                  MP1
○ PHYSICAL-QUANTITIES-UNITS
        (TERM.B1.COMP1.COMP1-Y[illegible]RIDE.VALUES
         TERM.B1.COMP1.COMP1-LENGTH     (PHYSICAL.QUANT[illegible]M.B1.COMP1.C
         TERM.B1.COMP1.COMP1-MOM        )               MP1
         TERM.B1.COMP1.COMP1-FORCE
        )
```

Figure C-16: Result of Executing B1.comp1.comp1-Force.Equil.Eq Task

In executing the task B1.comp1.comp2-Displ.Mom.Rel.J the system sends a message to the method slot *Mom-displ-j-qeval* to infer the qualitative value *Internal_force_M_i* of *Term.B1.comp1.comp2* using the known qualitative value *Q_Y_displacement_at_i* (Figure C-17).



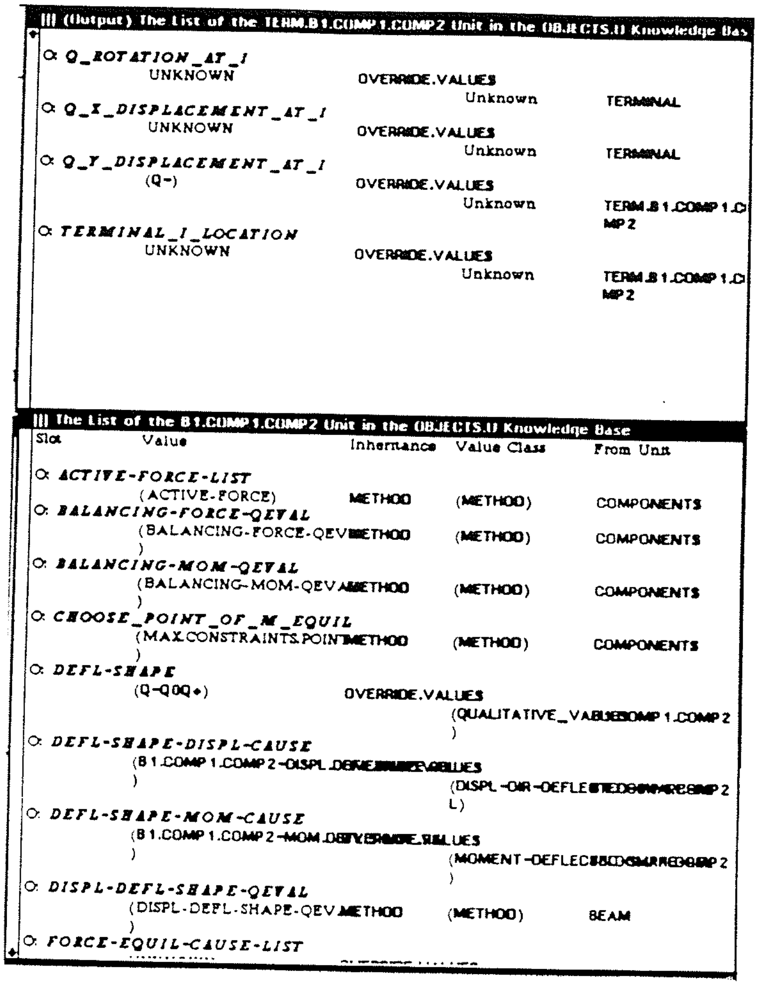
```
||| The List of the TERM.B1.COMP1.COMP2 Unit in the OBJECTS.U Knowledge Base
O: INTERNAL_FORCE_M_I
         (C+)                      OVERRIDE.VALUES
                                            (QUALITATIVE_VATERM.B1.COMP1.C
                                            )             MP2
O: INTERNAL_FORCE_N_I
         (C+)                      OVERRIDE.VALUES
                                            Unknown       TERM.B1.COMP1.C
                                                          MP2
O: INTERNAL_FORCE_S_I
         (C+)                      OVERRIDE.VALUES
                                            Unknown       TERM.B1.COMP1.C
                                                          MP2
O: PHYSICAL-QUANTITIES-UNITS
         (TERM.B1.COMP1.COMP2-Y OVERRIDE.VALUES
          TERM.B1.COMP1.COMP2-MOM           (PHYSICAL.QUANTITERM.B1.COMP1.C
          TERM.B1.COMP1.COMP2-FORCE         )             MP2
         )
```



```
||| The List of the TERM.B1.COMP1.COMP2 Unit in the OBJECTS.U Knowledge Base
O: INTERNAL_FORCE_M_I
         (Q+)                      OVERRIDE.VALUES
                                            (QUALITATIVE_VATERM.B1.COMP1.C
                                            )             MP2
O: INTERNAL_FORCE_N_I
         (C+)                      OVERRIDE.VALUES
                                            Unknown       TERM.B1.COMP1.C
                                                          MP2
O: INTERNAL_FORCE_S_I
         (C+)                      OVERRIDE.VALUES
                                            Unknown       TERM.B1.COMP1.C
                                                          MP2
O: PHYSICAL-QUANTITIES-UNITS
         (TERM.B1.COMP1.COMP2-Y OVERRIDE.VALUES
          TERM.B1.COMP1.COMP2-MOM           (PHYSICAL.QUANTITERM.B1.COMP1.C
          TERM.B1.COMP1.COMP2-FORCE         )             MP2
         )
```

Figure C-17: Result of Executing B1.comp1.comp2-Displ.Mom.Rel.J Task

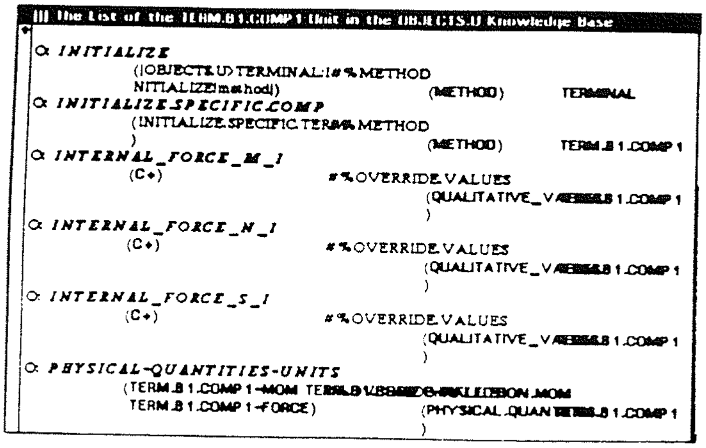
In executing the task B1.comp1.comp2-Displ.Defl.Shape.Rel the system sends a message to the method slot *Displ-defl-shape-qeval* to infer the qualitative value *Defl-Shape* of *B1.comp1.comp2* using the known qualitative value *Q_Y_displacement_at_i* of *Term.B1.comp1.comp2* and the support conditions (Figure C-18).



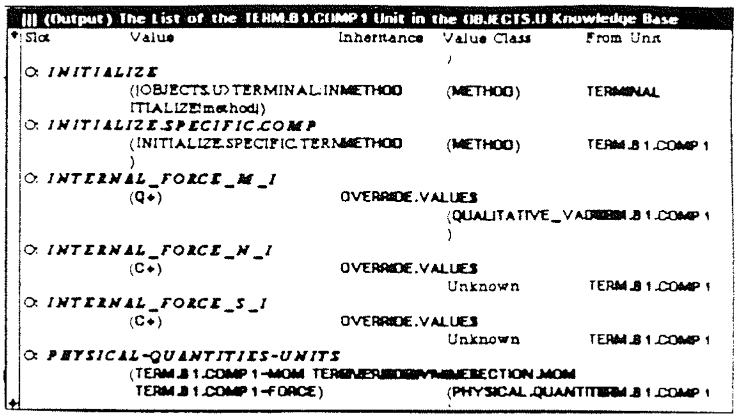
```
||| (Output) The List of the TERM.B1.COMP1.COMP2 Unit in the OBJECTS.U Knowledge Bas
○ Q_ROTATION_AT_I
        UNKNOWN                 OVERRIDE.VALUES
                                        Unknown         TERMINAL
○ Q_X_DISPLACEMENT_AT_I
        UNKNOWN                 OVERRIDE.VALUES
                                        Unknown         TERMINAL
○ Q_Y_DISPLACEMENT_AT_I
        (Q-)                    OVERRIDE.VALUES
                                        Unknown         TERM.B1.COMP1.C
                                                        MP2
○ TERMINAL_I_LOCATION
        UNKNOWN                 OVERRIDE.VALUES
                                        Unknown         TERM.B1.COMP1.C
                                                        MP2

||| The List of the B1.COMP1.COMP2 Unit in the OBJECTS.U Knowledge Base
Slot            Value           Inheritance     Value Class     From Unit

○ ACTIVE-FORCE-LIST
        (ACTIVE-FORCE)          METHOD          (METHOD)        COMPONENTS
○ BALANCING-FORCE-QEVAL
        (BALANCING-FORCE-QEVMETHOD              (METHOD)        COMPONENTS
        )
○ BALANCING-MOM-QEVAL
        (BALANCING-MOM-QEVAMETHOD               (METHOD)        COMPONENTS
        )
○ CHOOSE_POINT_OF_M_EQUIL
        (MAX.CONSTRAINTS.POINTMETHOD            (METHOD)        COMPONENTS
        )
○ DEFL-SHAPE
        (Q-Q0Q+)                OVERRIDE.VALUES
                                        (QUALITATIVE_VA[illegible]OMP1.COMP2
                                        )
○ DEFL-SHAPE-DISPL-CAUSE
        (B1.COMP1.COMP2-DISPL.DE[illegible]VALUES
        )
                                        (DISPL-DIR-DEFLE[illegible]P2
                                        L)
○ DEFL-SHAPE-MOM-CAUSE
        (B1.COMP1.COMP2-MOM.DE[illegible]LUES
        )
                                        (MOMENT-DEFLEC[illegible]P2
                                        )
○ DISPL-DEFL-SHAPE-QEVAL
        (DISPL-DEFL-SHAPE-QEV METHOD            (METHOD)        BEAM
        )
○ FORCE-EQUIL-CAUSE-LIST
```

Figure C-18: Result of Executing B1.comp1.comp2-Displ.Defl.Shape.Rel Task

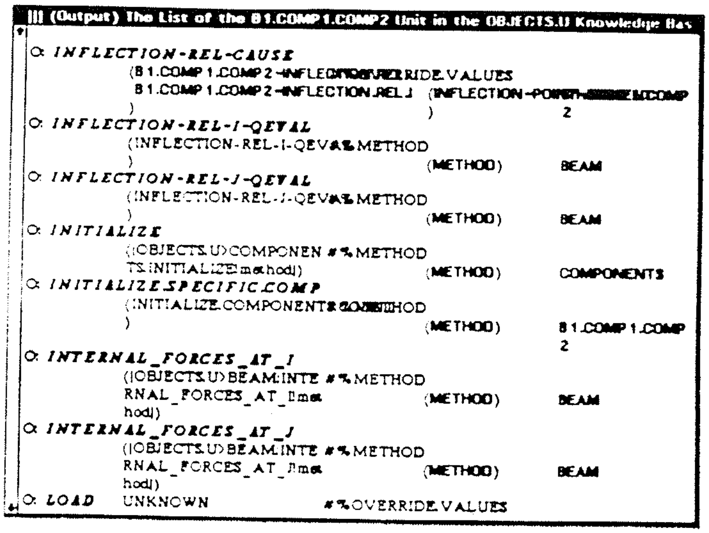
In executing the task B1.comp1.comp2-Inflection.Rel.I the system sends a message to the method slot *Inflection-rel-i-qeval* to infer the qualitative value *Inflection_mom* knowing that there is an inflection point on *B1.comp1.comp2*. This is followed by executing the task B1.comp1.comp2-Inflection.Rel.J. The system sends a message to method slot *Inflection-rel-j-qeval* of *B1.comp1.comp2* to infer the qualitative value *Inernal_Force_M_i* of *Term.B1.comp1* knowing that there is an inflection point on *B1.comp1.comp2* (Figure C-19).



```
||| The List of the TERM.B1.COMP1 Unit in the OBJECTS.U Knowledge Base
○ INITIALIZE
        (|OBJECTS.U>TERMINAL:I#%METHOD
        NITIALIZE|method|)                        (METHOD)      TERMINAL
○ INITIALIZE.SPECIFIC.COMP
        (INITIALIZE.SPECIFIC.TERM#%METHOD
        )                                         (METHOD)      TERM.B1.COMP1
○ INTERNAL_FORCE_M_I
        (C+)                   #%OVERRIDE.VALUES
                                          (QUALITATIVE_V[illegible]B1.COMP1
                                          )
○ INTERNAL_FORCE_N_I
        (C+)                   #%OVERRIDE.VALUES
                                          (QUALITATIVE_V[illegible]B1.COMP1
                                          )
○ INTERNAL_FORCE_S_I
        (C+)                   #%OVERRIDE.VALUES
                                          (QUALITATIVE_V[illegible]B1.COMP1
                                          )
○ PHYSICAL-QUANTITIES-UNITS
        (TERM.B1.COMP1-MOM TE[illegible]ON.MOM
         TERM.B1.COMP1-FORCE)                     (PHYSICAL.QUAN[illegible].B1.COMP1
                                          )
```



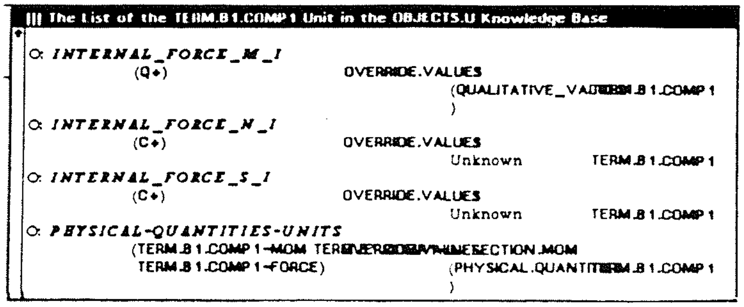
```
||| (Output) The List of the TERM.B1.COMP1 Unit in the OBJECTS.U Knowledge Base
Slot        Value                         Inheritance    Value Class        From Unit
                                                         )
○ INITIALIZE
            (|OBJECTS.U>TERMINAL:INMETHOD               (METHOD)           TERMINAL
            ITIALIZE|method|)
○ INITIALIZE.SPECIFIC.COMP
            (INITIALIZE.SPECIFIC.TERMMETHOD             (METHOD)           TERM.B1.COMP1
            )
○ INTERNAL_FORCE_M_I
            (Q+)                          OVERRIDE.VALUES
                                                         (QUALITATIVE_VA[illegible].B1.COMP1
                                                         )
○ INTERNAL_FORCE_N_I
            (C+)                          OVERRIDE.VALUES
                                                         Unknown            TERM.B1.COMP1
○ INTERNAL_FORCE_S_I
            (C+)                          OVERRIDE.VALUES
                                                         Unknown            TERM.B1.COMP1
○ PHYSICAL-QUANTITIES-UNITS
            (TERM.B1.COMP1-MOM TER[illegible]ECTION.MOM
             TERM.B1.COMP1-FORCE)                        (PHYSICAL.QUANTI[illegible].B1.COMP1
```



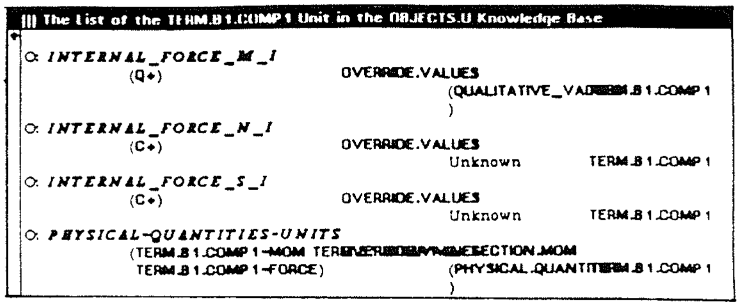
```
||| (Output) The List of the B1.COMP1.COMP2 Unit in the OBJECTS.U Knowledge Bas
○ INFLECTION-REL-CAUSE
        (B1.COMP1.COMP2-INFLEC[illegible]RIDE.VALUES
         B1.COMP1.COMP2-INFLECTION.REL.J  (INFLECTION-PO[illegible]COMP
        )                                  )                2
○ INFLECTION-REL-I-QEVAL
        (INFLECTION-REL-I-QEVA#%METHOD
        )                                  (METHOD)         BEAM
○ INFLECTION-REL-J-QEVAL
        (INFLECTION-REL-J-QEVA#%METHOD
        )                                  (METHOD)         BEAM
○ INITIALIZE
        (|OBJECTS.U>COMPONEN #%METHOD
        TS:INITIALIZE|method|)             (METHOD)         COMPONENTS
○ INITIALIZE.SPECIFIC.COMP
        (INITIALIZE.COMPONENTS[illegible]HOD
        )                                  (METHOD)         B1.COMP1.COMP
                                                            2
○ INTERNAL_FORCES_AT_I
        (|OBJECTS.U>BEAM:INTE #%METHOD
        RNAL_FORCES_AT_I|met               (METHOD)         BEAM
        hod|)
○ INTERNAL_FORCES_AT_J
        (|OBJECTS.U>BEAM:INTE #%METHOD
        RNAL_FORCES_AT_J|met               (METHOD)         BEAM
        hod|)
○ LOAD  UNKNOWN               #%OVERRIDE.VALUES
```

Figure C-19: Result of Executing B1.comp1.comp2-Inflection.Rel.J Task

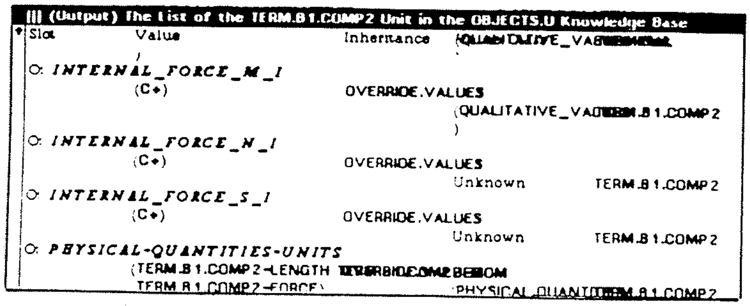
In executing the task JC1-Mom.Equil.Eq the system sends a message to the method slot *Balancing-Mom-qeval* to infer the qualitative value *Internal_force_M_i* of *Term.B1.comp2* using the known qualitative value *Internal_force_M_i* of *Term.B1.comp1* (Figure C-20).



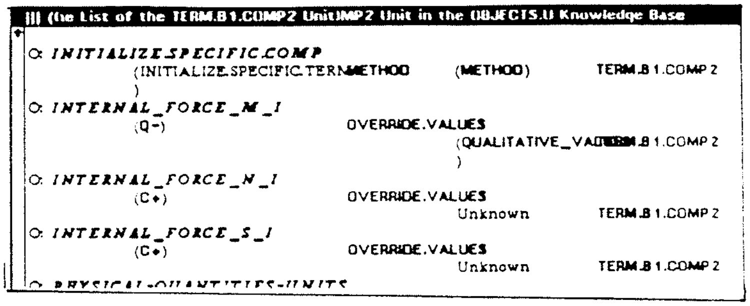
```
||| The List of the TERM.B1.COMP1 Unit in the OBJECTS.U Knowledge Base
O: INTERNAL_FORCE_M_I
          (Q+)                    OVERRIDE.VALUES
                                          (QUALITATIVE_VATERM.B 1.COMP 1
                                          )
O: INTERNAL_FORCE_N_I
          (C+)                    OVERRIDE.VALUES
                                          Unknown        TERM.B 1.COMP 1
O: INTERNAL_FORCE_S_I
          (C+)                    OVERRIDE.VALUES
                                          Unknown        TERM.B 1.COMP 1
O: PHYSICAL-QUANTITIES-UNITS
          (TERM.B 1.COMP 1-MOM TERVERRIDEVALUESECTION.MOM
           TERM.B 1.COMP 1-FORCE)                 (PHYSICAL.QUANTITERM.B 1.COMP 1
                                          )
```



```
||| The List of the TERM.B1.COMP1 Unit in the OBJECTS.U Knowledge Base
O: INTERNAL_FORCE_M_I
          (Q+)                    OVERRIDE.VALUES
                                          (QUALITATIVE_VATERM.B 1.COMP 1
                                          )
O: INTERNAL_FORCE_N_I
          (C+)                    OVERRIDE.VALUES
                                          Unknown        TERM.B 1.COMP 1
O: INTERNAL_FORCE_S_I
          (C+)                    OVERRIDE.VALUES
                                          Unknown        TERM.B 1.COMP 1
O: PHYSICAL-QUANTITIES-UNITS
          (TERM.B 1.COMP 1-MOM TERVERRIDEVALUESECTION.MOM
           TERM.B 1.COMP 1-FORCE)                 (PHYSICAL.QUANTITERM.B 1.COMP 1
                                          )
```



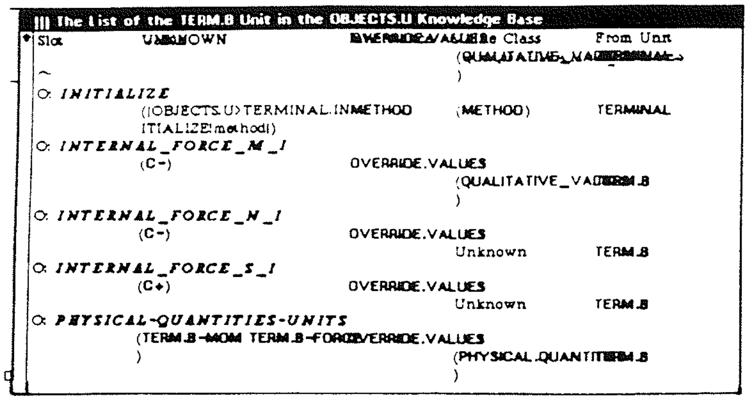
```
||| (Output) The List of the TERM.B1.COMP2 Unit in the OBJECTS.U Knowledge Base
Slot      Value                   Inheritance     (QUALITATIVE_VA[illegible]
          )
O: INTERNAL_FORCE_M_I
          (C+)                    OVERRIDE.VALUES
                                          (QUALITATIVE_VATERM.B 1.COMP 2
                                          )
O: INTERNAL_FORCE_N_I
          (C+)                    OVERRIDE.VALUES
                                          Unknown        TERM.B 1.COMP 2
O: INTERNAL_FORCE_S_I
          (C+)                    OVERRIDE.VALUES
                                          Unknown        TERM.B 1.COMP 2
O: PHYSICAL-QUANTITIES-UNITS
          (TERM.B 1.COMP 2-LENGTH TERRIDECOMP2BEMOM
           TERM.B 1.COMP 2-FORCE)                 (PHYSICAL.QUANTITERM.B 1.COMP 2
```



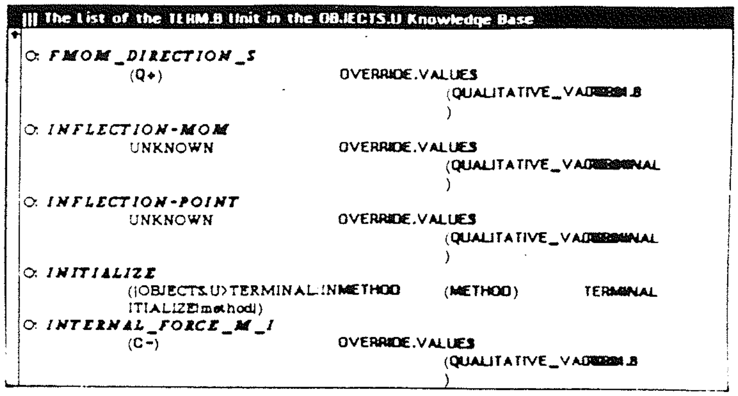
```
||| The List of the TERM.B1.COMP2 UnitMP2 Unit in the OBJECTS.U Knowledge Base
O: INITIALIZE.SPECIFIC.COMP
          (INITIALIZE.SPECIFIC.TERMMETHOD      (METHOD)       TERM.B 1.COMP 2
          )
O: INTERNAL_FORCE_M_I
          (Q-)                    OVERRIDE.VALUES
                                          (QUALITATIVE_VATERM.B 1.COMP 2
                                          )
O: INTERNAL_FORCE_N_I
          (C+)                    OVERRIDE.VALUES
                                          Unknown        TERM.B 1.COMP 2
O: INTERNAL_FORCE_S_I
          (C+)                    OVERRIDE.VALUES
                                          Unknown        TERM.B 1.COMP 2
O: PHYSICAL-QUANTITIES-UNITS
```

Figure C-20: Result of Executing JC1-Mom.Equil.Eq Task

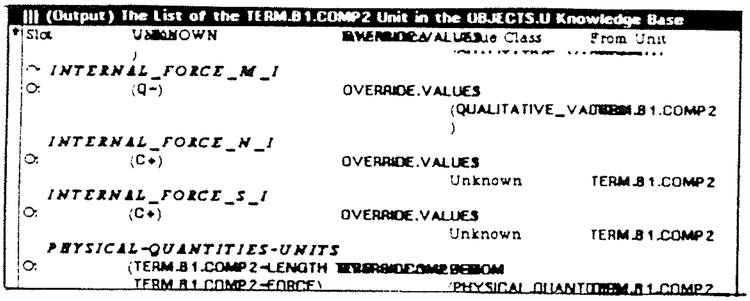
In executing the task B1.comp2-Mom.Equil.Eq the system sends a message to the method slot *Balancing-Mom-qeval* to infer the qualitative value *FMom_direction_S* of *Term.B* using the known qualitative value *Internal_force_M_i* of *Term.B1.comp2* (Figure C-21).



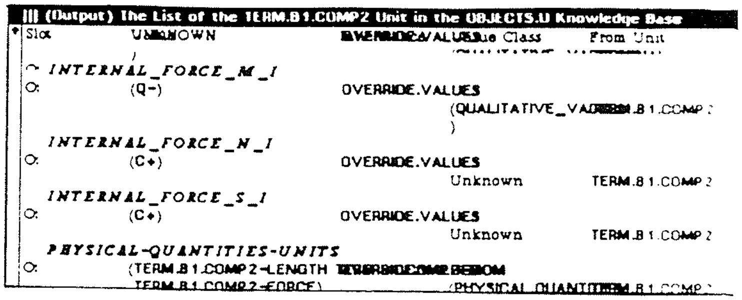
```
||| The List of the TERM.B Unit in the OBJECTS.U Knowledge Base
Slot          UNKNOWN                 OVERRIDE.VALUESue Class      From Unit
                                              (QUALITATIVE_VATERMINAL
~                                             )
○ INITIALIZE
              (|OBJECTS.U>TERMINAL.INMETHOD   (METHOD)             TERMINAL
              ITIALIZE!method|)
○ INTERNAL_FORCE_M_I
              (C-)                    OVERRIDE.VALUES
                                              (QUALITATIVE_VATERM.B
                                              )
○ INTERNAL_FORCE_N_I
              (C-)                    OVERRIDE.VALUES
                                              Unknown              TERM.B
○ INTERNAL_FORCE_S_I
              (C+)                    OVERRIDE.VALUES
                                              Unknown              TERM.B
○ PHYSICAL-QUANTITIES-UNITS
              (TERM.B-MOM TERM.B-FORCEOVERRIDE.VALUES
              )                               (PHYSICAL.QUANTITERM.B
                                              )
```



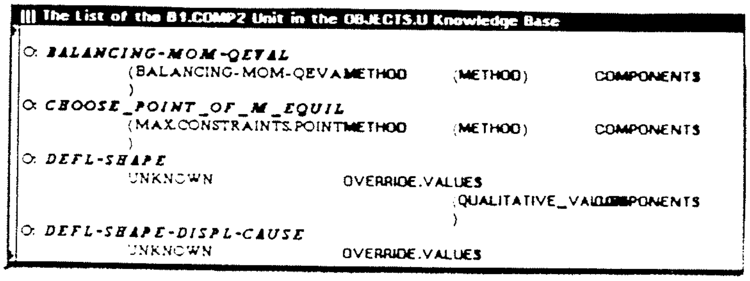
```
||| The List of the TERM.B Unit in the OBJECTS.U Knowledge Base

○ FMOM_DIRECTION_S
              (Q+)                    OVERRIDE.VALUES
                                              (QUALITATIVE_VATERM.B
                                              )
○ INFLECTION-MOM
              UNKNOWN                 OVERRIDE.VALUES
                                              (QUALITATIVE_VATERMINAL
                                              )
○ INFLECTION-POINT
              UNKNOWN                 OVERRIDE.VALUES
                                              (QUALITATIVE_VATERMINAL
                                              )
○ INITIALIZE
              (|OBJECTS.U>TERMINAL.INMETHOD   (METHOD)             TERMINAL
              ITIALIZE!method|)
○ INTERNAL_FORCE_M_I
              (C-)                    OVERRIDE.VALUES
                                              (QUALITATIVE_VATERM.B
                                              )
```



```
||| (Output) The List of the TERM.B1.COMP2 Unit in the OBJECTS.U Knowledge Base
Slot          UNKNOWN                 OVERRIDE.VALUESue Class      From Unit
              )
~ INTERNAL_FORCE_M_I
○             (Q-)                    OVERRIDE.VALUES
                                              (QUALITATIVE_VATERM.B1.COMP2
                                              )
  INTERNAL_FORCE_N_I
○             (C+)                    OVERRIDE.VALUES
                                              Unknown              TERM.B1.COMP2
  INTERNAL_FORCE_S_I
○             (C+)                    OVERRIDE.VALUES
                                              Unknown              TERM.B1.COMP2
  PHYSICAL-QUANTITIES-UNITS
○             (TERM.B1.COMP2-LENGTH OVERRIDE.VALUES
              TERM.B1.COMP2-FORCE)            (PHYSICAL.QUANTITERM.B1.COMP2
```



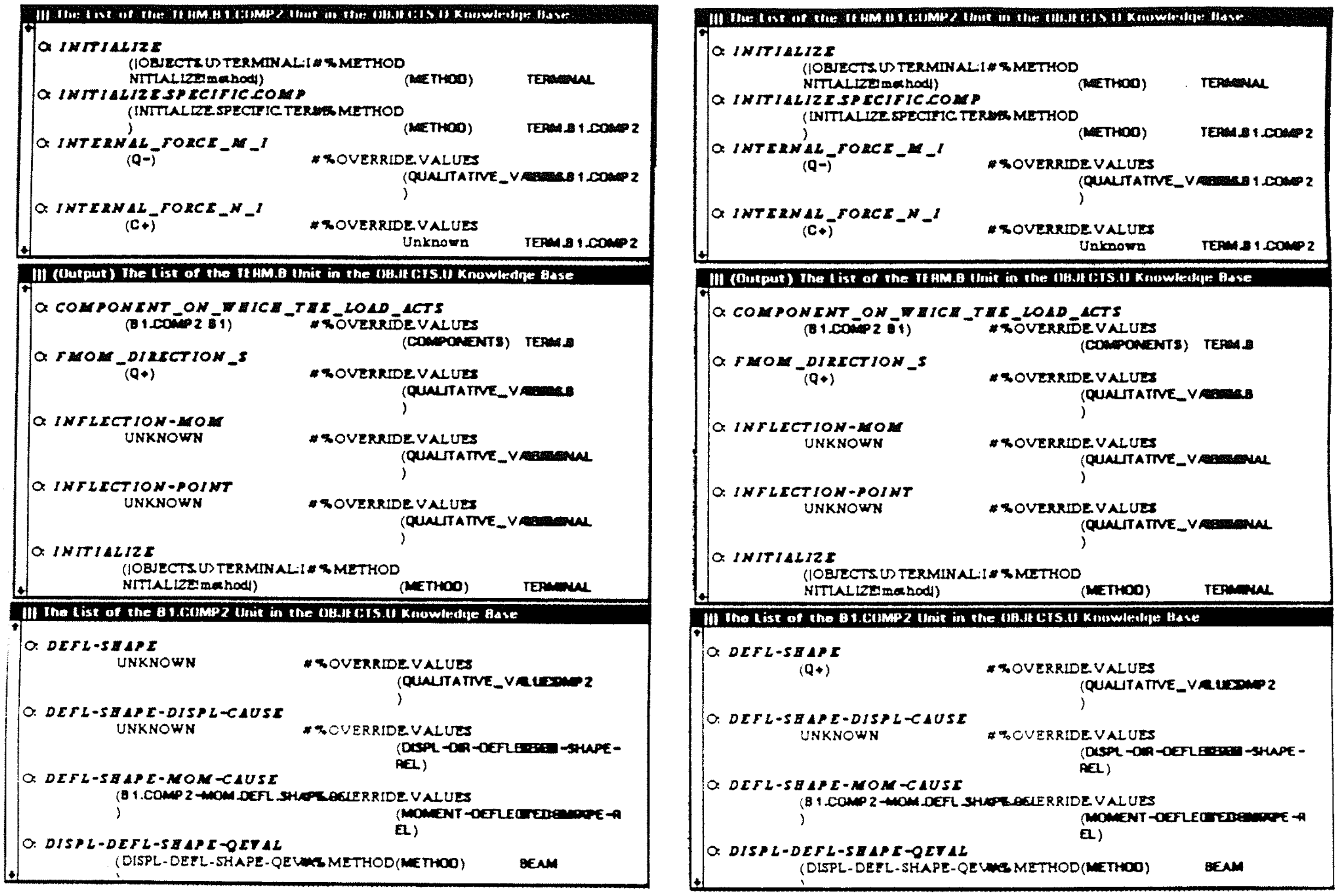
```
||| (Output) The List of the TERM.B1.COMP2 Unit in the OBJECTS.U Knowledge Base
Slot          UNKNOWN                 OVERRIDE.VALUESue Class      From Unit
              )
~ INTERNAL_FORCE_M_I
○             (Q-)                    OVERRIDE.VALUES
                                              (QUALITATIVE_VATERM.B1.COMP2
                                              )
  INTERNAL_FORCE_N_I
○             (C+)                    OVERRIDE.VALUES
                                              Unknown              TERM.B1.COMP2
  INTERNAL_FORCE_S_I
○             (C+)                    OVERRIDE.VALUES
                                              Unknown              TERM.B1.COMP2
  PHYSICAL-QUANTITIES-UNITS
○             (TERM.B1.COMP2-LENGTH OVERRIDE.VALUES
              TERM.B1.COMP2-FORCE)            (PHYSICAL.QUANTITERM.B1.COMP2
```



```
||| The List of the B1.COMP2 Unit in the OBJECTS.U Knowledge Base

○ BALANCING-MOM-QEVAL
              (BALANCING-MOM-QEVAMETHOD       (METHOD)             COMPONENTS
              )
○ CHOOSE_POINT_OF_M_EQUIL
              (MAX.CONSTRAINTS.POINTMETHOD    (METHOD)             COMPONENTS
              )
○ DEFL-SHAPE
              UNKNOWN                 OVERRIDE.VALUES
                                              (QUALITATIVE_VALCOMPONENTS
                                              )
○ DEFL-SHAPE-DISPL-CAUSE
              UNKNOWN                 OVERRIDE.VALUES
```

Figure C-21: Result of Executing B1.comp2-Mom.Equil.Eq Task

In executing the task B1.comp2-Mom.Defl.Shape.Rel the system sends a message to the method slot *Mom-defl-shape-qeval* to infer the qualitative value *Defl-shape* of *B1.comp2* using the known qualitative values *Internal_force_M_i* of *Term.B1.comp2* and *FMom_direction_S* of *Term.B* (Figure C-22).

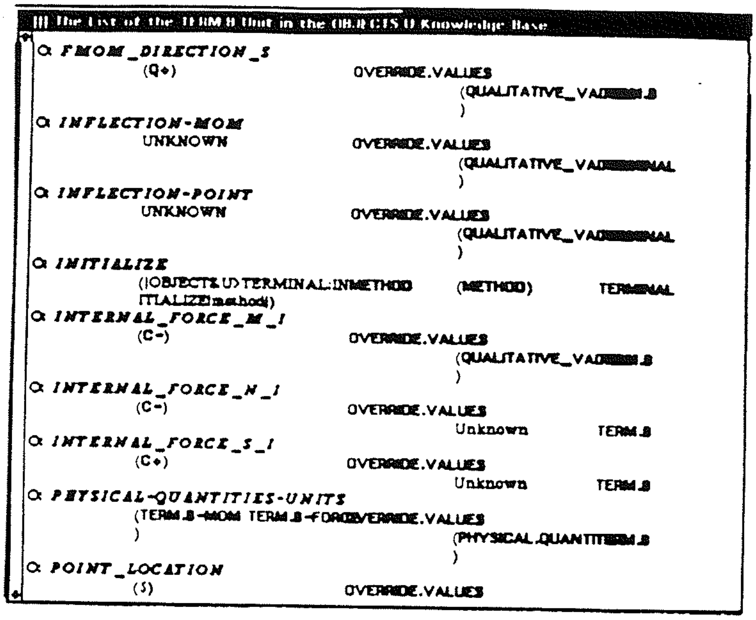


```
||| The List of the TERM.B1.COMP2 Unit in the OBJECTS.U Knowledge Base
○ INITIALIZE
          (|OBJECTS.U>TERMINAL:I#%METHOD
          NITIALIZE!method|)                      (METHOD)        TERMINAL
○ INITIALIZE.SPECIFIC.COMP
          (INITIALIZE.SPECIFIC.TER[illegible]METHOD
          )                                       (METHOD)        TERM.B1.COMP2
○ INTERNAL_FORCE_M_I
          (Q-)               #%OVERRIDE.VALUES
                                                  (QUALITATIVE_V[illegible]B1.COMP2
                                                  )
○ INTERNAL_FORCE_N_I
          (C+)               #%OVERRIDE.VALUES
                                                  Unknown         TERM.B1.COMP2

||| (Output) The List of the TERM.B Unit in the OBJECTS.U Knowledge Base
○ COMPONENT_ON_WHICH_THE_LOAD_ACTS
          (B1.COMP2 B1)      #%OVERRIDE.VALUES
                                                  (COMPONENTS)  TERM.B
○ FMOM_DIRECTION_S
          (Q+)               #%OVERRIDE.VALUES
                                                  (QUALITATIVE_V[illegible]B
                                                  )
○ INFLECTION-MOM
          UNKNOWN            #%OVERRIDE.VALUES
                                                  (QUALITATIVE_V[illegible]NAL
                                                  )
○ INFLECTION-POINT
          UNKNOWN            #%OVERRIDE.VALUES
                                                  (QUALITATIVE_V[illegible]NAL
                                                  )
○ INITIALIZE
          (|OBJECTS.U>TERMINAL:I#%METHOD
          NITIALIZE!method|)                      (METHOD)        TERMINAL

||| The List of the B1.COMP2 Unit in the OBJECTS.U Knowledge Base
○ DEFL-SHAPE
          UNKNOWN            #%OVERRIDE.VALUES
                                                  (QUALITATIVE_VALUE[illegible]MP2
                                                  )
○ DEFL-SHAPE-DISPL-CAUSE
          UNKNOWN            #%OVERRIDE.VALUES
                                                  (DISPL-DIR-DEFL[illegible]-SHAPE-
                                                  REL)
○ DEFL-SHAPE-MOM-CAUSE
          (B1.COMP2-MOM.DEFL.SHAP[illegible]ERRIDE.VALUES
          )                                       (MOMENT-DEFLE[illegible]PE-R
                                                  EL)
○ DISPL-DEFL-SHAPE-QEVAL
          (DISPL-DEFL-SHAPE-QE[illegible]METHOD(METHOD)          BEAM
```

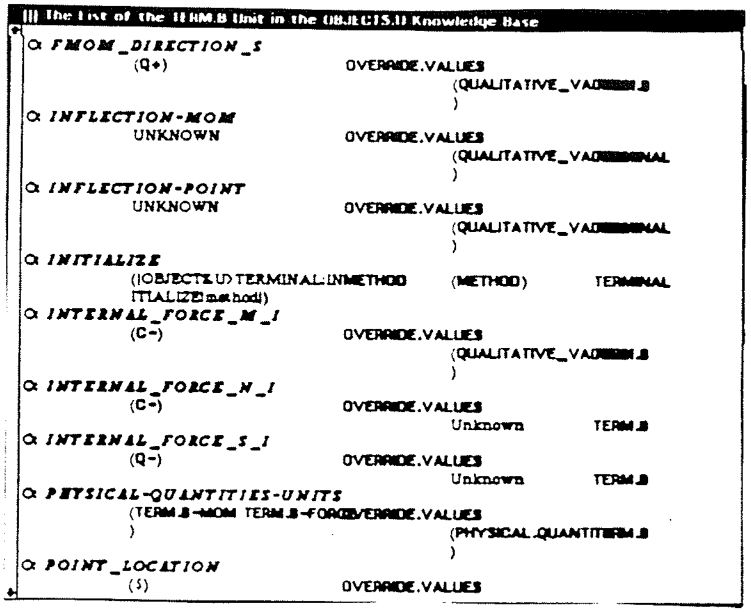
```
||| The List of the TERM.B1.COMP2 Unit in the OBJECTS.U Knowledge Base
○ INITIALIZE
          (|OBJECTS.U>TERMINAL:I#%METHOD
          NITIALIZE!method|)                      (METHOD)        TERMINAL
○ INITIALIZE.SPECIFIC.COMP
          (INITIALIZE.SPECIFIC.TER[illegible]METHOD
          )                                       (METHOD)        TERM.B1.COMP2
○ INTERNAL_FORCE_M_I
          (Q-)               #%OVERRIDE.VALUES
                                                  (QUALITATIVE_V[illegible]B1.COMP2
                                                  )
○ INTERNAL_FORCE_N_I
          (C+)               #%OVERRIDE.VALUES
                                                  Unknown         TERM.B1.COMP2

||| (Output) The List of the TERM.B Unit in the OBJECTS.U Knowledge Base
○ COMPONENT_ON_WHICH_THE_LOAD_ACTS
          (B1.COMP2 B1)      #%OVERRIDE.VALUES
                                                  (COMPONENTS)  TERM.B
○ FMOM_DIRECTION_S
          (Q+)               #%OVERRIDE.VALUES
                                                  (QUALITATIVE_V[illegible]B
                                                  )
○ INFLECTION-MOM
          UNKNOWN            #%OVERRIDE.VALUES
                                                  (QUALITATIVE_V[illegible]NAL
                                                  )
○ INFLECTION-POINT
          UNKNOWN            #%OVERRIDE.VALUES
                                                  (QUALITATIVE_V[illegible]NAL
                                                  )
○ INITIALIZE
          (|OBJECTS.U>TERMINAL:I#%METHOD
          NITIALIZE!method|)                      (METHOD)        TERMINAL

||| The List of the B1.COMP2 Unit in the OBJECTS.U Knowledge Base
○ DEFL-SHAPE
          (Q+)               #%OVERRIDE.VALUES
                                                  (QUALITATIVE_VALUE[illegible]MP2
                                                  )
○ DEFL-SHAPE-DISPL-CAUSE
          UNKNOWN            #%OVERRIDE.VALUES
                                                  (DISPL-DIR-DEFL[illegible]-SHAPE-
                                                  REL)
○ DEFL-SHAPE-MOM-CAUSE
          (B1.COMP2-MOM.DEFL.SHAP[illegible]ERRIDE.VALUES
          )                                       (MOMENT-DEFLE[illegible]PE-R
                                                  EL)
○ DISPL-DEFL-SHAPE-QEVAL
          (DISPL-DEFL-SHAPE-QE[illegible]METHOD(METHOD)          BEAM
```

Figure C-22: Result of Executing B1.comp2-Mom.Defl.Shape.Rel Task

In executing the task B1.comp2-Mom.Force.Rel the system sends a message to the method slot *Calc.S.Force* to infer the qualitative value *Internal_force_S_i* of *Term.B* using the known qualitative value of *FMom_direction_S* and the *Point_location* with respect to which the unknown force generates this moment (Figure C-23).



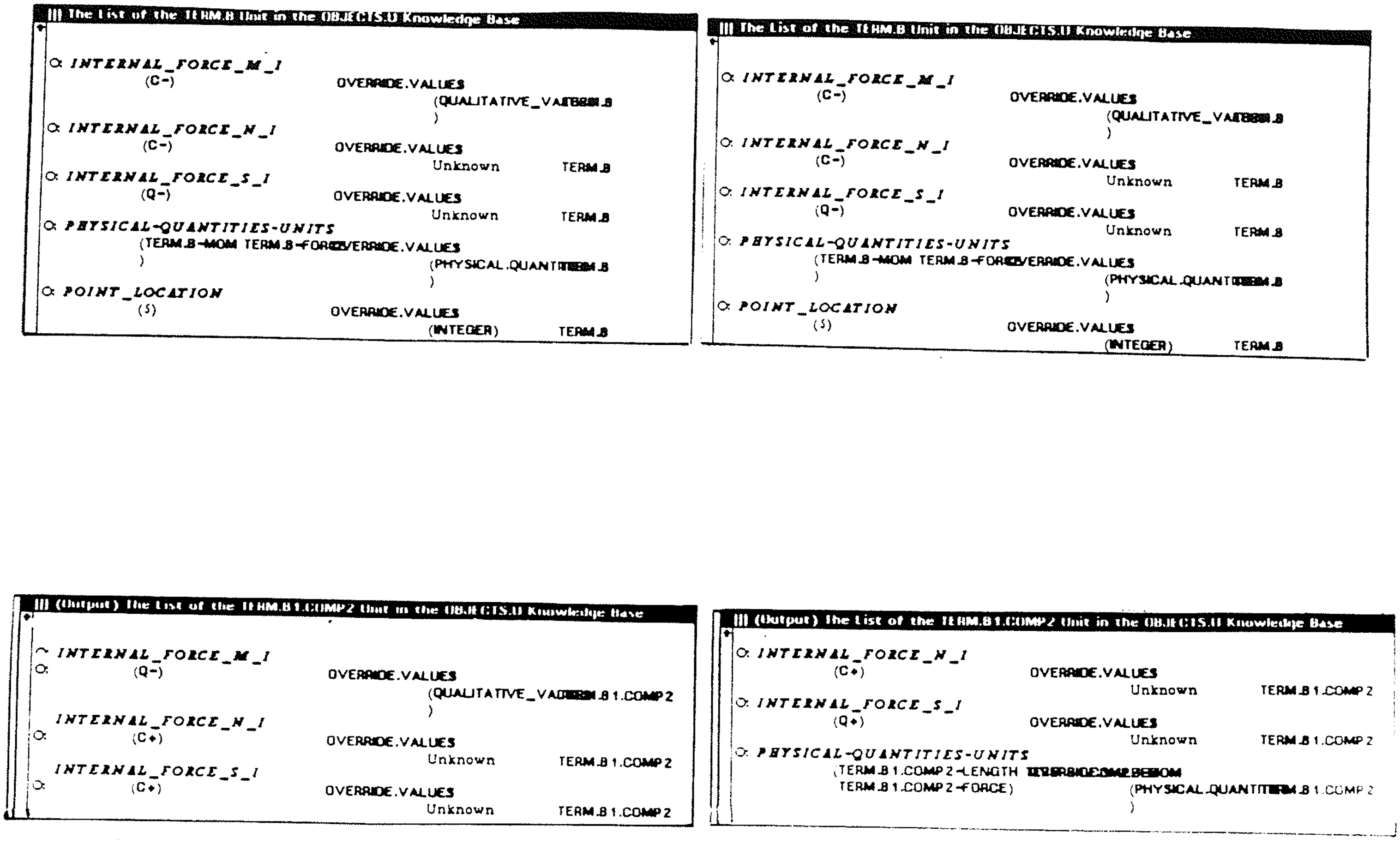
```
The List of the TERM.B Unit in the OBJECTS.U Knowledge Base
FMOM_DIRECTION_S
        (Q+)                      OVERRIDE.VALUES
                                          (QUALITATIVE_VA[illegible].B
                                          )
INFLECTION-MOM
        UNKNOWN                   OVERRIDE.VALUES
                                          (QUALITATIVE_VA[illegible]NAL
                                          )
INFLECTION-POINT
        UNKNOWN                   OVERRIDE.VALUES
                                          (QUALITATIVE_VA[illegible]NAL
                                          )
INITIALIZE
        (|OBJECTS.U) TERMINAL:INMETHOD    (METHOD)        TERMINAL
        ITIALIZE!method|)
INTERNAL_FORCE_M_I
        (C-)                      OVERRIDE.VALUES
                                          (QUALITATIVE_VA[illegible].B
                                          )
INTERNAL_FORCE_N_I
        (C-)                      OVERRIDE.VALUES
                                          Unknown         TERM.B
INTERNAL_FORCE_S_I
        (C+)                      OVERRIDE.VALUES
                                          Unknown         TERM.B
PHYSICAL-QUANTITIES-UNITS
        (TERM.B-MOM TERM.B-FORCOVERRIDE.VALUES
        )                                 (PHYSICAL.QUANTITERM.B
                                          )
POINT_LOCATION
        (S)                       OVERRIDE.VALUES
```



```
The List of the TERM.B Unit in the OBJECTS.U Knowledge Base
FMOM_DIRECTION_S
        (Q+)                      OVERRIDE.VALUES
                                          (QUALITATIVE_VA[illegible].B
                                          )
INFLECTION-MOM
        UNKNOWN                   OVERRIDE.VALUES
                                          (QUALITATIVE_VA[illegible]NAL
                                          )
INFLECTION-POINT
        UNKNOWN                   OVERRIDE.VALUES
                                          (QUALITATIVE_VA[illegible]NAL
                                          )
INITIALIZE
        (|OBJECTS.U) TERMINAL:INMETHOD    (METHOD)        TERMINAL
        ITIALIZE!method|)
INTERNAL_FORCE_M_I
        (C-)                      OVERRIDE.VALUES
                                          (QUALITATIVE_VA[illegible].B
                                          )
INTERNAL_FORCE_N_I
        (C-)                      OVERRIDE.VALUES
                                          Unknown         TERM.B
INTERNAL_FORCE_S_I
        (Q-)                      OVERRIDE.VALUES
                                          Unknown         TERM.B
PHYSICAL-QUANTITIES-UNITS
        (TERM.B-MOM TERM.B-FORCOVERRIDE.VALUES
        )                                 (PHYSICAL.QUANTITERM.B
                                          )
POINT_LOCATION
        (S)                       OVERRIDE.VALUES
```

Figure C-23: Result of Executing B1.comp2-Mom.Force.Rel Task

In executing the task B1.comp2-Force.Equil.Eq the system sends a message to the method slot *Balancing-Force-qeval* to infer the qualitative value *Internal_force_S_i of Term.B1.comp2* using the known qualitative value *Internal_force_S_i* of *Term.B* (Figure C-24).



```
||| The List of the TERM.B Unit in the OBJECTS.U Knowledge Base

○ INTERNAL_FORCE_M_I
        (C-)                    OVERRIDE.VALUES
                                        (QUALITATIVE_VATERM.B
                                        )
○ INTERNAL_FORCE_N_I
        (C-)                    OVERRIDE.VALUES
                                        Unknown         TERM.B
○ INTERNAL_FORCE_S_I
        (Q-)                    OVERRIDE.VALUES
                                        Unknown         TERM.B
○ PHYSICAL-QUANTITIES-UNITS
        (TERM.B-MOM TERM.B-FOROVERRIDE.VALUES
        )                               (PHYSICAL.QUANTITERM.B
                                        )
○ POINT_LOCATION
        (S)                     OVERRIDE.VALUES
                                        (INTEGER)       TERM.B
```

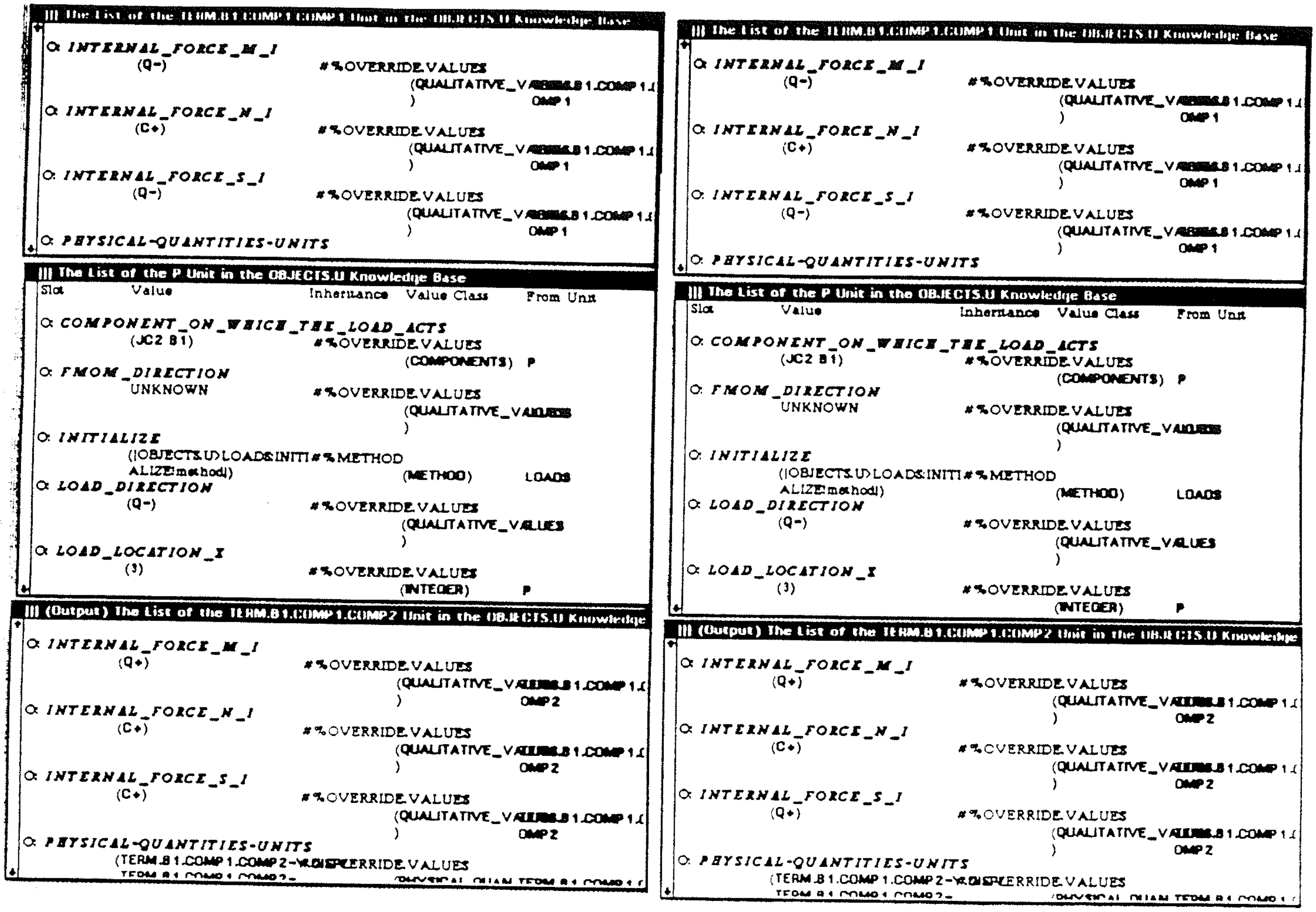
```
||| The List of the TERM.B Unit in the OBJECTS.U Knowledge Base

○ INTERNAL_FORCE_M_I
        (C-)                    OVERRIDE.VALUES
                                        (QUALITATIVE_VATERM.B
                                        )
○ INTERNAL_FORCE_N_I
        (C-)                    OVERRIDE.VALUES
                                        Unknown         TERM.B
○ INTERNAL_FORCE_S_I
        (Q-)                    OVERRIDE.VALUES
                                        Unknown         TERM.B
○ PHYSICAL-QUANTITIES-UNITS
        (TERM.B-MOM TERM.B-FOROVERRIDE.VALUES
        )                               (PHYSICAL.QUANTITERM.B
                                        )
○ POINT_LOCATION
        (S)                     OVERRIDE.VALUES
                                        (INTEGER)       TERM.B
```

```
||| (Output) The List of the TERM.B1.COMP2 Unit in the OBJECTS.U Knowledge Base

○ INTERNAL_FORCE_M_I
        (Q-)                    OVERRIDE.VALUES
                                        (QUALITATIVE_VATERM.B1.COMP2
                                        )
○ INTERNAL_FORCE_N_I
        (C+)                    OVERRIDE.VALUES
                                        Unknown         TERM.B1.COMP2
○ INTERNAL_FORCE_S_I
        (C+)                    OVERRIDE.VALUES
                                        Unknown         TERM.B1.COMP2
```

```
||| (Output) The List of the TERM.B1.COMP2 Unit in the OBJECTS.U Knowledge Base

○ INTERNAL_FORCE_N_I
        (C+)                    OVERRIDE.VALUES
                                        Unknown         TERM.B1.COMP2
○ INTERNAL_FORCE_S_I
        (Q+)                    OVERRIDE.VALUES
                                        Unknown         TERM.B1.COMP2
○ PHYSICAL-QUANTITIES-UNITS
        (TERM.B1.COMP2-LENGTH TERM.B1.COMP2-MOM
        TERM.B1.COMP2-FORCE)            (PHYSICAL.QUANTITERM.B1.COMP2
                                        )
```

Figure C-24: Result of Executing B1.comp2-Force.Equil.Eq Task

Executing the task JC2-Force.Equil.Eq the system sends a message to the method slot *Balancing-Force-qeval* to infer the qualitative value *Internal_force_S_i of Term.B1.comp1.comp2* using the known the qualitative values *Internal_force_S_i* of *Term.B1.comp1.comp1* and *Load_direction* of *P* (Figure C-25).



```
||| The List of the TERM.B1.COMP1.COMP1 Unit in the OBJECTS.U Knowledge Base
○ INTERNAL_FORCE_M_I
        (Q-)                  #%OVERRIDE.VALUES
                                  (QUALITATIVE_V[illegible]B1.COMP1.(
                                  )              OMP1
○ INTERNAL_FORCE_N_I
        (C+)                  #%OVERRIDE.VALUES
                                  (QUALITATIVE_V[illegible]B1.COMP1.(
                                  )              OMP1
○ INTERNAL_FORCE_S_I
        (Q-)                  #%OVERRIDE.VALUES
                                  (QUALITATIVE_V[illegible]B1.COMP1.(
                                  )              OMP1
○ PHYSICAL-QUANTITIES-UNITS
```

```
||| The List of the P Unit in the OBJECTS.U Knowledge Base
Slot     Value                Inheritance   Value Class     From Unit
○ COMPONENT_ON_WHICH_THE_LOAD_ACTS
        (JC2 B1)              #%OVERRIDE.VALUES
                                  (COMPONENTS)  P
○ FMOM_DIRECTION
        UNKNOWN               #%OVERRIDE.VALUES
                                  (QUALITATIVE_VALUES
                                  )
○ INITIALIZE
        (|OBJECTS.U>LOADS:INITI#%METHOD
        ALIZE:method|)            (METHOD)      LOADS
○ LOAD_DIRECTION
        (Q-)                  #%OVERRIDE.VALUES
                                  (QUALITATIVE_VALUES
                                  )
○ LOAD_LOCATION_X
        (3)                   #%OVERRIDE.VALUES
                                  (INTEGER)     P
```

```
||| (Output) The List of the TERM.B1.COMP1.COMP2 Unit in the OBJECTS.U Knowledge
○ INTERNAL_FORCE_M_I
        (Q+)                  #%OVERRIDE.VALUES
                                  (QUALITATIVE_V[illegible]B1.COMP1.(
                                  )              OMP2
○ INTERNAL_FORCE_N_I
        (C+)                  #%OVERRIDE.VALUES
                                  (QUALITATIVE_V[illegible]B1.COMP1.(
                                  )              OMP2
○ INTERNAL_FORCE_S_I
        (C+)                  #%OVERRIDE.VALUES
                                  (QUALITATIVE_V[illegible]B1.COMP1.(
                                  )              OMP2
○ PHYSICAL-QUANTITIES-UNITS
        (TERM.B1.COMP1.COMP2-[illegible]ERRIDE.VALUES
        TERM.B1.COMP1.COMP2-      (PHYSICAL_QUAN TERM.B1.COMP1.(
```

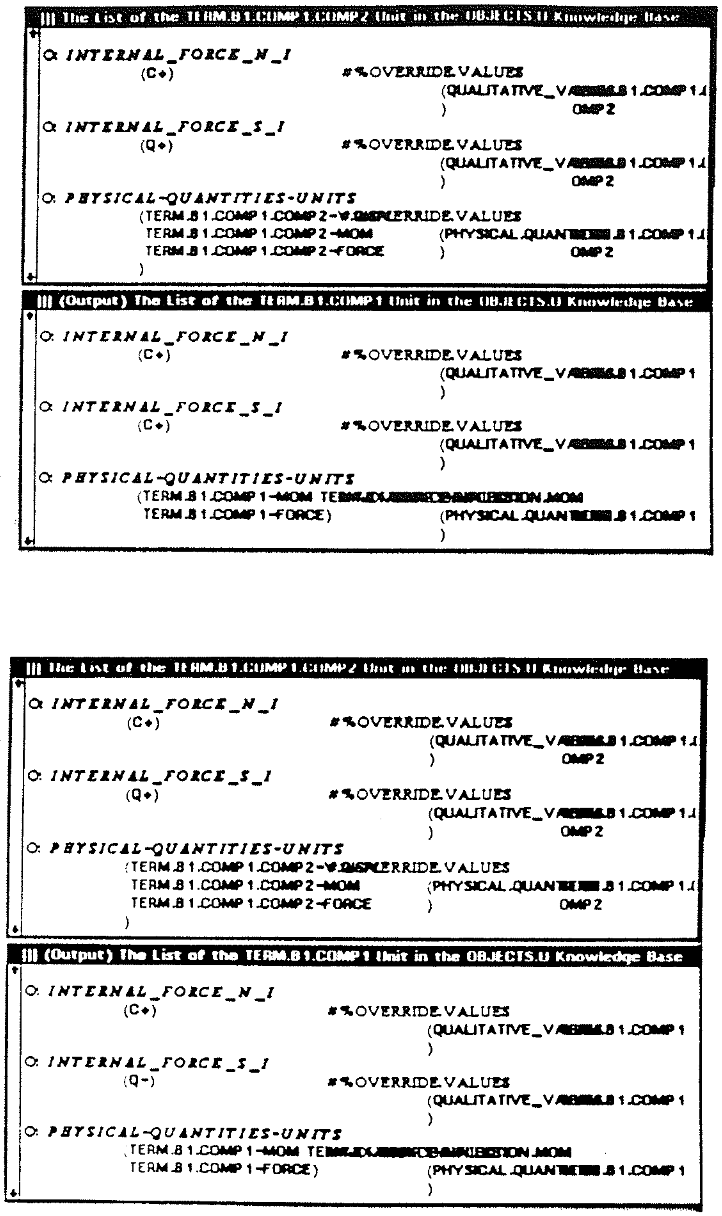
```
||| The List of the TERM.B1.COMP1.COMP1 Unit in the OBJECTS.U Knowledge Base
○ INTERNAL_FORCE_M_I
        (Q-)                  #%OVERRIDE.VALUES
                                  (QUALITATIVE_V[illegible]B1.COMP1.(
                                  )              OMP1
○ INTERNAL_FORCE_N_I
        (C+)                  #%OVERRIDE.VALUES
                                  (QUALITATIVE_V[illegible]B1.COMP1.(
                                  )              OMP1
○ INTERNAL_FORCE_S_I
        (Q-)                  #%OVERRIDE.VALUES
                                  (QUALITATIVE_V[illegible]B1.COMP1.(
                                  )              OMP1
○ PHYSICAL-QUANTITIES-UNITS
```

```
||| The List of the P Unit in the OBJECTS.U Knowledge Base
Slot     Value                Inheritance   Value Class     From Unit
○ COMPONENT_ON_WHICH_THE_LOAD_ACTS
        (JC2 B1)              #%OVERRIDE.VALUES
                                  (COMPONENTS)  P
○ FMOM_DIRECTION
        UNKNOWN               #%OVERRIDE.VALUES
                                  (QUALITATIVE_VALUES
                                  )
○ INITIALIZE
        (|OBJECTS.U>LOADS:INITI#%METHOD
        ALIZE:method|)            (METHOD)      LOADS
○ LOAD_DIRECTION
        (Q-)                  #%OVERRIDE.VALUES
                                  (QUALITATIVE_VALUES
                                  )
○ LOAD_LOCATION_X
        (3)                   #%OVERRIDE.VALUES
                                  (INTEGER)     P
```

```
||| (Output) The List of the TERM.B1.COMP1.COMP2 Unit in the OBJECTS.U Knowledge
○ INTERNAL_FORCE_M_I
        (Q+)                  #%OVERRIDE.VALUES
                                  (QUALITATIVE_V[illegible]B1.COMP1.(
                                  )              OMP2
○ INTERNAL_FORCE_N_I
        (C+)                  #%OVERRIDE.VALUES
                                  (QUALITATIVE_V[illegible]B1.COMP1.(
                                  )              OMP2
○ INTERNAL_FORCE_S_I
        (Q+)                  #%OVERRIDE.VALUES
                                  (QUALITATIVE_V[illegible]B1.COMP1.(
                                  )              OMP2
○ PHYSICAL-QUANTITIES-UNITS
        (TERM.B1.COMP1.COMP2-[illegible]ERRIDE.VALUES
        TERM.B1.COMP1.COMP2-      (PHYSICAL_QUAN TERM.B1.COMP1.(
```

Figure C-25: Result of Executing JC2-Force.Equil.Eq Task

In executing the task B1.comp1.comp2-Force.Equil.Eq the system sends a message to the method slot *Balancing-Force-qeval* to infer the qualitative value *Internal_force_S_i of Term.B1.comp1* using the known qualitative value *Internal_force_S_i* of *Term.B1.comp1.comp2* (Figure C-26).

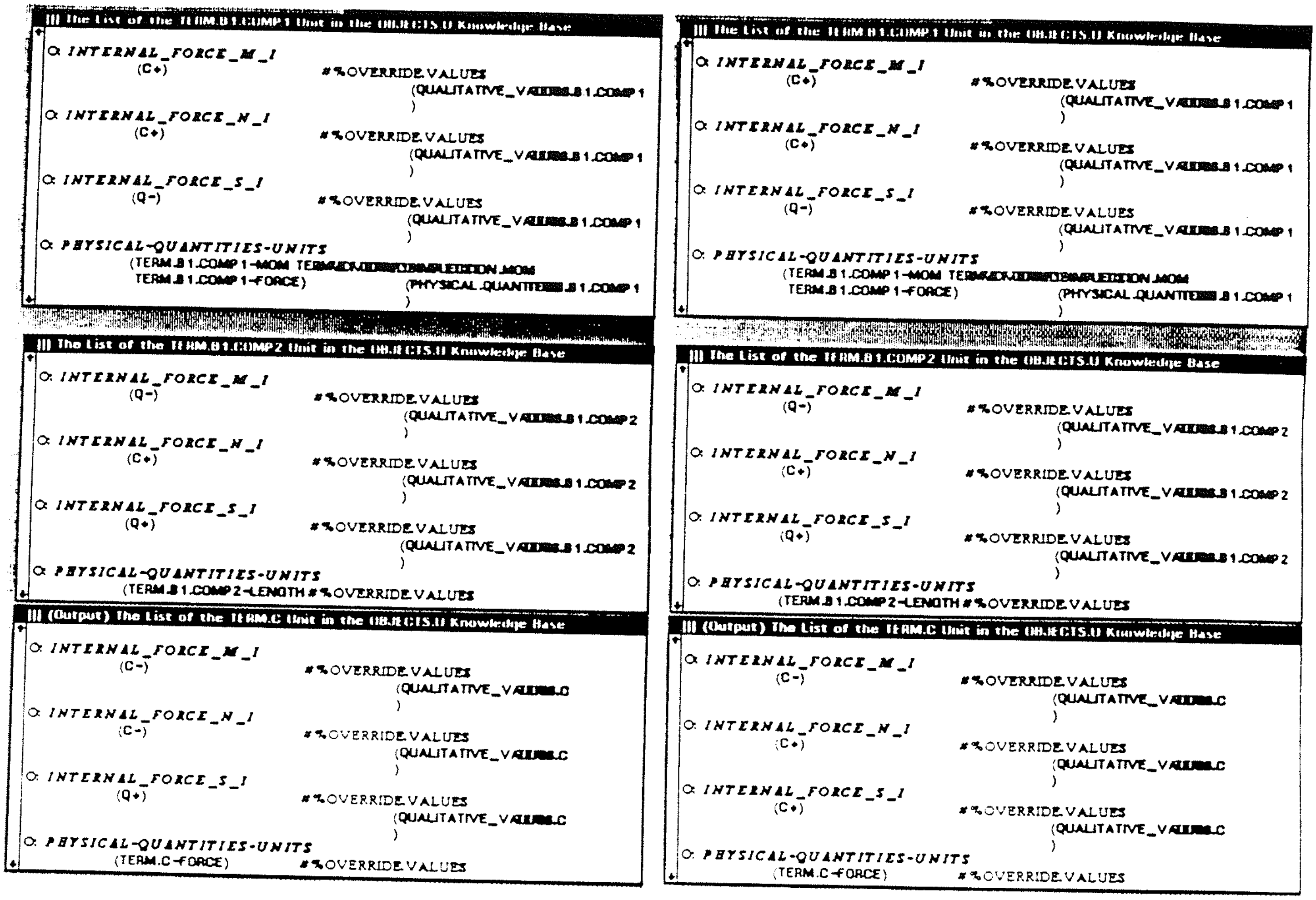


```
||| The List of the TERM.B1.COMP1.COMP2 Unit in the OBJECTS.U Knowledge Base
O: INTERNAL_FORCE_N_I
        (C+)                  #%OVERRIDE.VALUES
                                      (QUALITATIVE_V[illegible]B1.COMP1.C
                                      )               OMP2
O: INTERNAL_FORCE_S_I
        (Q+)                  #%OVERRIDE.VALUES
                                      (QUALITATIVE_V[illegible]B1.COMP1.C
                                      )               OMP2
O: PHYSICAL-QUANTITIES-UNITS
        (TERM.B1.COMP1.COMP2-[illegible]ERRIDE.VALUES
         TERM.B1.COMP1.COMP2-MOM      (PHYSICAL.QUAN[illegible].B1.COMP1.C
         TERM.B1.COMP1.COMP2-FORCE    )               OMP2
        )
||| (Output) The List of the TERM.B1.COMP1 Unit in the OBJECTS.U Knowledge Base
O: INTERNAL_FORCE_N_I
        (C+)                  #%OVERRIDE.VALUES
                                      (QUALITATIVE_V[illegible]B1.COMP1
                                      )
O: INTERNAL_FORCE_S_I
        (C+)                  #%OVERRIDE.VALUES
                                      (QUALITATIVE_V[illegible]B1.COMP1
                                      )
O: PHYSICAL-QUANTITIES-UNITS
        (TERM.B1.COMP1-MOM TE[illegible]ON.MOM
         TERM.B1.COMP1-FORCE)         (PHYSICAL.QUAN[illegible].B1.COMP1
                                      )
```

```
||| The List of the TERM.B1.COMP1.COMP2 Unit in the OBJECTS.U Knowledge Base
O: INTERNAL_FORCE_N_I
        (C+)                  #%OVERRIDE.VALUES
                                      (QUALITATIVE_V[illegible]B1.COMP1.C
                                      )               OMP2
O: INTERNAL_FORCE_S_I
        (Q+)                  #%OVERRIDE.VALUES
                                      (QUALITATIVE_V[illegible]B1.COMP1.C
                                      )               OMP2
O: PHYSICAL-QUANTITIES-UNITS
        (TERM.B1.COMP1.COMP2-[illegible]ERRIDE.VALUES
         TERM.B1.COMP1.COMP2-MOM      (PHYSICAL.QUAN[illegible].B1.COMP1.C
         TERM.B1.COMP1.COMP2-FORCE    )               OMP2
        )
||| (Output) The List of the TERM.B1.COMP1 Unit in the OBJECTS.U Knowledge Base
O: INTERNAL_FORCE_N_I
        (C+)                  #%OVERRIDE.VALUES
                                      (QUALITATIVE_V[illegible]B1.COMP1
                                      )
O: INTERNAL_FORCE_S_I
        (Q-)                  #%OVERRIDE.VALUES
                                      (QUALITATIVE_V[illegible]B1.COMP1
                                      )
O: PHYSICAL-QUANTITIES-UNITS
        (TERM.B1.COMP1-MOM TE[illegible]ON.MOM
         TERM.B1.COMP1-FORCE)         (PHYSICAL.QUAN[illegible].B1.COMP1
                                      )
```

Figure C-26: Result of Executing B1.comp1.comp2-Force.Equil.Eq Task

In executing the task JC1-Force.Equil.Eq the system first tries to apply the force equilibrium law. Since the two known qualitative values - *Internal_force_S_i* of *Term.B1.compl* *(Q-)* and *Term.B1.comp2* *(Q+)* - are of opposite sign the system invokes Quantity Lattice to identify which of the two forces is greater in order to derive the qualitative value *Internal_force_S_i* of *Term.C* (Figure C-27).



```
||| The List of the TERM.B1.COMP1 Unit in the OBJECTS.U Knowledge Base
O INTERNAL_FORCE_M_I
        (C+)                  #%OVERRIDE.VALUES
                                   (QUALITATIVE_V[illegible].B1.COMP1
                                   )
O INTERNAL_FORCE_N_I
        (C+)                  #%OVERRIDE.VALUES
                                   (QUALITATIVE_V[illegible].B1.COMP1
                                   )
O INTERNAL_FORCE_S_I
        (Q-)                  #%OVERRIDE.VALUES
                                   (QUALITATIVE_V[illegible].B1.COMP1
                                   )
O PHYSICAL-QUANTITIES-UNITS
        (TERM.B1.COMP1-MOM  TE[illegible]ON.MOM
         TERM.B1.COMP1-FORCE)          (PHYSICAL.QUANTIT[illegible].B1.COMP1
                                   )
```

```
||| The List of the TERM.B1.COMP2 Unit in the OBJECTS.U Knowledge Base
O INTERNAL_FORCE_M_I
        (Q-)                  #%OVERRIDE.VALUES
                                   (QUALITATIVE_V[illegible].B1.COMP2
                                   )
O INTERNAL_FORCE_N_I
        (C+)                  #%OVERRIDE.VALUES
                                   (QUALITATIVE_V[illegible].B1.COMP2
                                   )
O INTERNAL_FORCE_S_I
        (Q+)                  #%OVERRIDE.VALUES
                                   (QUALITATIVE_V[illegible].B1.COMP2
                                   )
O PHYSICAL-QUANTITIES-UNITS
        (TERM.B1.COMP2-LENGTH #%OVERRIDE.VALUES
```

```
||| (Output) The List of the TERM.C Unit in the OBJECTS.U Knowledge Base
O INTERNAL_FORCE_M_I
        (C-)                  #%OVERRIDE.VALUES
                                   (QUALITATIVE_V[illegible].C
                                   )
O INTERNAL_FORCE_N_I
        (C-)                  #%OVERRIDE.VALUES
                                   (QUALITATIVE_V[illegible].C
                                   )
O INTERNAL_FORCE_S_I
        (Q+)                  #%OVERRIDE.VALUES
                                   (QUALITATIVE_V[illegible].C
                                   )
O PHYSICAL-QUANTITIES-UNITS
        (TERM.C-FORCE)        #%OVERRIDE.VALUES
```

```
||| The List of the TERM.B1.COMP1 Unit in the OBJECTS.U Knowledge Base
O INTERNAL_FORCE_M_I
        (C+)                  #%OVERRIDE.VALUES
                                   (QUALITATIVE_V[illegible].B1.COMP1
                                   )
O INTERNAL_FORCE_N_I
        (C+)                  #%OVERRIDE.VALUES
                                   (QUALITATIVE_V[illegible].B1.COMP1
                                   )
O INTERNAL_FORCE_S_I
        (Q-)                  #%OVERRIDE.VALUES
                                   (QUALITATIVE_V[illegible].B1.COMP1
                                   )
O PHYSICAL-QUANTITIES-UNITS
        (TERM.B1.COMP1-MOM  TE[illegible]ON.MOM
         TERM.B1.COMP1-FORCE)          (PHYSICAL.QUANTIT[illegible].B1.COMP1
                                   )
```

```
||| The List of the TERM.B1.COMP2 Unit in the OBJECTS.U Knowledge Base
O INTERNAL_FORCE_M_I
        (Q-)                  #%OVERRIDE.VALUES
                                   (QUALITATIVE_V[illegible].B1.COMP2
                                   )
O INTERNAL_FORCE_N_I
        (C+)                  #%OVERRIDE.VALUES
                                   (QUALITATIVE_V[illegible].B1.COMP2
                                   )
O INTERNAL_FORCE_S_I
        (Q+)                  #%OVERRIDE.VALUES
                                   (QUALITATIVE_V[illegible].B1.COMP2
                                   )
O PHYSICAL-QUANTITIES-UNITS
        (TERM.B1.COMP2-LENGTH #%OVERRIDE.VALUES
```

```
||| (Output) The List of the TERM.C Unit in the OBJECTS.U Knowledge Base
O INTERNAL_FORCE_M_I
        (C-)                  #%OVERRIDE.VALUES
                                   (QUALITATIVE_V[illegible].C
                                   )
O INTERNAL_FORCE_N_I
        (C+)                  #%OVERRIDE.VALUES
                                   (QUALITATIVE_V[illegible].C
                                   )
O INTERNAL_FORCE_S_I
        (C+)                  #%OVERRIDE.VALUES
                                   (QUALITATIVE_V[illegible].C
                                   )
O PHYSICAL-QUANTITIES-UNITS
        (TERM.C-FORCE)        #%OVERRIDE.VALUES
```

Figure C-27: Result of Executing JC1-Force.Equil.Eq Task

We summarize part of the relevant qualitative analysis information (e.g., reactions at supports, internal shear forces and moments, and deflected shapes) in image panels shown in Figure C-28 and Figure C-29. Figure C-28 illustrates the known qualitative value of the concentrated load *P (Q-)* and the identified interactions at supports and relevant sections. Recall that C+ indicates that a constraint exists, and C- indicates that the constraint does not exist.

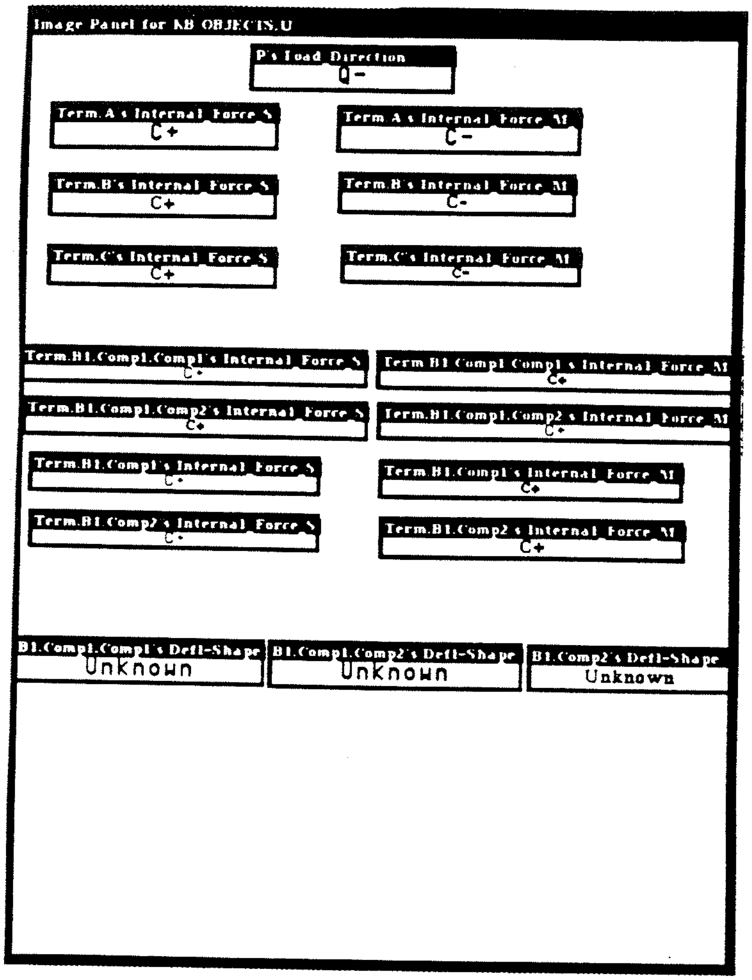


Image Panel for KB OBJECTS.U

P's Load Direction: Q-

Term.A's Internal Force S: C+
Term.A's Internal Force M: C-

Term.B's Internal Force S: C+
Term.B's Internal Force M: C-

Term.C's Internal Force S: C+
Term.C's Internal Force M: C-

Term.B1.Comp1.Comp1's Internal Force S: C+
Term.B1.Comp1.Comp1's Internal Force M: C+

Term.B1.Comp1.Comp2's Internal Force S: C+
Term.B1.Comp1.Comp2's Internal Force M: C+

Term.B1.Comp1's Internal Force S: C+
Term.B1.Comp1's Internal Force M: C+

Term.B1.Comp2's Internal Force S: C+
Term.B1.Comp2's Internal Force M: C+

B1.Comp1.Comp1's Defl-Shape: Unknown
B1.Comp1.Comp2's Defl-Shape: Unknown
B1.Comp2's Defl-Shape: Unknown

Figure C-28: Qualitative Analysis Image Panel - Before Qualitative Evaluation

Figure C-29 illustrates the inferred qualitative analysis results for:

- the vertical reactions at the supports and shear forces at the relevant sections, where *Q+* represents an upward force and *Q-* a downward force;
- the moments at the relevant sections of the continuous beam, where *Q+* represents a clockwise rotation for the moment and *Q-* a counter-clockwise rotation;
- the deflected shapes of the three sub-components of the continuous beam, where Q- represents a downward deflected shape, *Q+* represents an upward deflected shape, and *Q-Q0Q+* indicates also the change of curvature in the deflected shape.

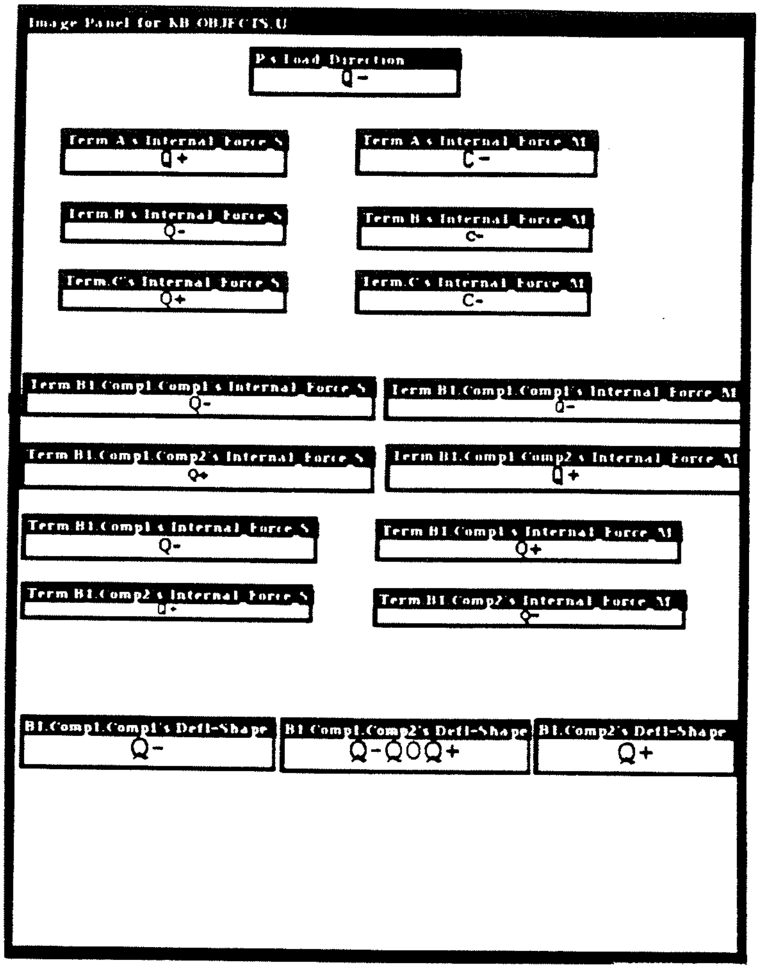


Image Panel for KB OBJECTS.U

P's Load Direction
Q-

Term.A's Internal Force S
Q+

Term.A's Internal Force M
C-

Term.B's Internal Force S
Q-

Term.B's Internal Force M
C-

Term.C's Internal Force S
Q+

Term.C's Internal Force M
C-

Term.B1.Comp1.Comp1's Internal Force S
Q-

Term.B1.Comp1.Comp1's Internal Force M
Q-

Term.B1.Comp1.Comp2's Internal Force S
Q+

Term.B1.Comp1.Comp2's Internal Force M
Q+

Term.B1.Comp1's Internal Force S
Q-

Term.B1.Comp1's Internal Force M
Q+

Term.B1.Comp2's Internal Force S
Q+

Term.B1.Comp2's Internal Force M
Q-

B1.Comp1.Comp1's Defl-Shape
Q-

B1.Comp1.Comp2's Defl-Shape
Q-Q0Q+

B1.Comp2's Defl-Shape
Q+

Figure C-29: Qualitative Analysis Image Panel - After Qualitative Evaluation

Appendix D Related Work in Qualitative Physics

Our work builds upon the body of work in the area of qualitative physics in artificial intelligence. Qualitative physics studies knowledge representation and techniques for reasoning about the behavior of physical systems without precise numerical information. This is exactly what is needed for our purpose because precise numerical information is unavailable at the conceptual design stage. The works in qualitative modeling and analysis which we have studied most closely in designing our system include those by deKleer and Brown [Refs. 2, 3, 4], Kuipers [Ref. 12, 13], Forbus [Ref. 5], and Iwasaki and Simon [Ref. 11]. In this section, we describe some of the main features of these approaches. We discuss their benefits and limitations and contrast them with those of QStruc from the perspective of their possible application to the domain of structural analysis.

D.1 Basic concepts

To facilitate our discussion, we first explain some of the basic concepts of qualitative analysis.

D.1.1 Qualitative calculus

In qualitative physics, just as in more conventional modeling techniques, physical systems are modeled as the set of variables. The behavior of a system is described in terms of how these variable values change with respect to time. In qualitative physics, the values of variables are not known precisely but only up to an interval that contains the real value. Such intervals are called *qualitative values* and variables *qualitative variables,* which we will denote with [] as in [x]. The set of all possible qualitative values (i.e. the intervals) that a variable can assume is predefined for each variable by dividing the entire range of the variable into non-overlapping intervals. The end points of qualitative values are called *landmark values*. The set of all landmark values for a variable is called the *quantity space* of the variable. The quantity space can be the same for all variables or different for each variable. The set of qualitative values commonly used is the set of positive, zero, and negative, denoted as *{-, 0, +}*. As we have described, we have used this set for all variables in QStruc. The quantity space of {-infinity, 0, +infinity} defines the set of qualitative values*{-, 0, +}*. A *qualitative state* of a system is defined by a set of qualitative variables and their values, as in *S1: {[x]=+, [y]= -, [z]=-}*, where S1 represents a state and *[x], [y], and [z]* are qualitative variables. For the rest of this section, we assume that the quantity space of every variable is {-infinity, 0, +infinity}, but the argument extends straightforwardly to different quantity spaces.

One can define *qualitative operations,* such as qualitative addition, subtraction, etc. on qualitative values, similar to those arithmetic operations on real numbers. (See Appendix A.) Equations consisting of qualitative variables and operators provide means to represent imprecise knowledge about functional relations among quantities. For example, if one knows that the larger the displacement of a weight on a spring becomes, the larger the force of the spring on the weight will be without knowing the precise form of the function, one can write

$$[df] = -[dx],$$

where df and dx denote the derivatives of the force on the weight and the displacement.

If one knows the precise form of the functional relation among variables but chooses to model a system qualitatively, quantitative information will be lost in the process. For example, a real-valued equation:

$$2x + 8y = 0$$

will be transformed into the following qualitative equation

$$[x] + [y] = 0.$$

Viewing qualitative equations as constraints on the values of the qualitative variables, one can solve a system for qualitative equations by propagating known variable values through qualitative equations according to the definitions of the qualitative operators. For instance, consider the following two qualitative equations:

$$[x] = [y]$$

$$[x] + [y] = [z].$$

If it is known that

$$[y] = -,$$

one can propagate this value to variable x through the first equation to obtain,

$$[x] = -.$$

Then, these values and the second equation can be used to conclude

$$[z] = -.$$

Note, however, that since qualitative arithmetic is ambiguous, one may not be able to obtain a unique solution even given the same number of equations as variables. For instance, in the above example, if the second equation is $[x] - [y] = [z]$ instead of $[x] + [y] = [z]$, the value of $[z]$ can be any one of $-$, 0 or $+$.

D.1.2 Qualitative prediction

Most work in qualitative physics is concerned with behavior of physical systems over time. Given a set of qualitative equations describing a system, one can try to predict its behavior over time. *Qualitative behavior* over time is a sequence (not necessarily linear) of qualitative states. One way to obtain a qualitative behavior from a set of qualitative equations is as follows: The first step determines the set of all possible qualitative states that are consistent with the equations. This is carried out by "solving" the equations to produce all solutions, each solution consisting of values of all the variables including derivatives. Each solution represents a possible qualitative state for the system. The second step examines each state and determines possible transitions i.e. the set of possible states the system can move into from the state. Upon the assumption that all the variables are continuous, a set of transition rules can be defined to determine possible transitions. For example:

- Transition from zero:
  If S1: [x]=0, [dx]=+
  then [x] must be + in the next state.

- Transition to zero:
  If S1: [x] = +, [dx] = -
  then [x] may be + or 0 in the next state.

- Continuity:
  If S1: [x] = + (or -)
  then [x] cannot be - (or +) in the next state.

The basic idea is that since the sign of the derivative of a variable indicates the direction in which the variable value is changing, given the variable value and its derivative, one can predict its possible next value. Also, since the variables are assumed continuous, the values cannot change from positive to negative or vice versa without going through a state where the value is zero. If more than one variable are changing towards a landmark value in one state, there can be multiple possible transitions depending on which variable reaches its landmark value first, making the prediction ambiguous. We will further address the issue of ambiguity in later discussion.

Figure D-1 illustrates an example of deriving the sequence of state transitions. In this figure states are represented by the qualitative values of a variable *[x]* and its qualitative derivative *[dx]*, and by possible transition arrows. Transitions represented by crossed-out arrows represent state transitions that are ruled out.

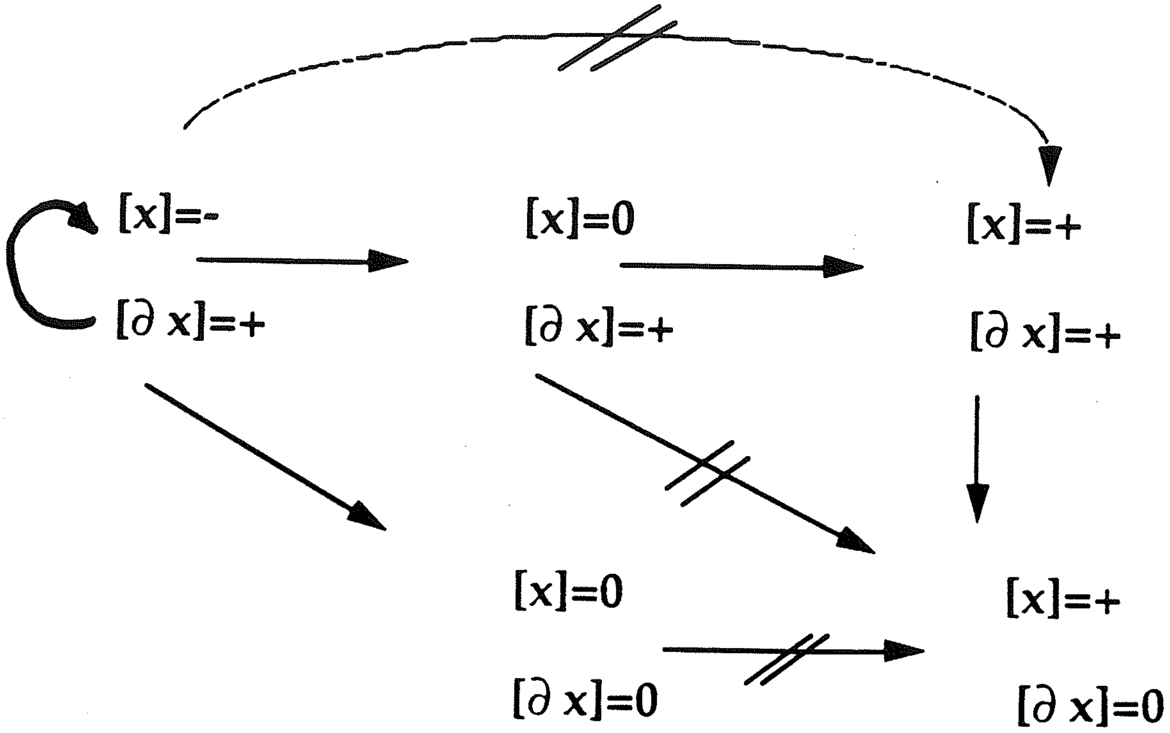


Figure D-1: Example of Transition Analysis

D.1.3 Causal reasoning

One important part of people's knowledge about the behavior of physical systems is knowledge about causal relations. The concept of causality undoubtedly plays an important role in people's understanding of how things work. The capability to represent and make use of causal knowledge has many benefits, of which the most significant from the point of view of using the knowledge for practical purposes, is that causal knowledge implies possible means to control the behavior to achieve a desired effect. For example, the knowledge that the current through a resistor depends on the resistance given a certain voltage across the resistor suggests us to change the resistor to achieve a desired value for the current, but we would not try to change the resistance by controlling the current. Despite the importance of the causal concept in human knowledge about the physical world, formal physical theories are usually expressed in terms of functional relations, which do not contain information about the direction of causality. For example, the equation representing the Newton's second law,

$$F = ma,$$

says neither that the force F causes the acceleration a, nor that the acceleration causes the force. However, in an informal explanation of how things achieve their behavior, people almost always use causal terminology.

In the following section, we will restrict ourselves to discussing causal dependency relations among variables as opposed to causal relations among states. Qualitative equations, just as ordinary equations, represent acausal, functional relations among variables. However, unlike in formal physics, the concept of causality is an important and explicit part of research in qualitative physics

D.2 Discussion of Approaches

In this section, we discuss some of the main features of representative works in qualitative physics, namely those of deKleer and Brown [Refs. 2, 3, 4], Kuipers [Refs. 12, 13], Forbus [Ref. 5], and Iwasaki and Simon [Ref. 11]), examine their benefits and limitations in structural engineering applications, and discuss their relation with the approaches taken in QStruc. The main goal of all but one (Iwasaki and Simon) of these works is to derive and explain the behavior of a system for a given model. Since we are not concerned about behavior over time in QStruc, we will mainly focus our discussion on other aspects of the works that are relevant to us. In particular, we are interested in (1) the primitives used to model physical systems, (2) modeling principles, and (3) the roles the notion of causality is handled by these systems.

D.2.1 ENVISION

deKleer and Brown implemented a system called ENVISION [Ref. 3], whose purpose is to infer the behavior of physical systems from their structure. deKleer and Brown take a component-oriented approach to modeling physical systems, in which a device is represented in terms of distinct components and their physical connections. Their approach is strongly influenced by the electrical circuit domain, which is their primary domain, but their examples also include non-electrical systems, such as hydraulic systems. Besides component topology, ENVISION is given the description of the behavior properties of each type of component in terms of qualitative equations as well as initial qualitative values to some of the variables. Connections also give rise to qualitative equations, though such equations are limited to the kind that only propagates information without altering it (e.g. an equation $[x] = [y]$ where x and y are variables such as voltages belonging to two components directly connected). In other words, connections are strictly passive in the sense that they cannot process information. Only components may process information. Given these inputs, ENVISION solves the equations by propagating known values through them and predicts the behavior.

Central to deKleer and Brown's approach are the *no-function-in-structure* and *locality principles.* The no-function-in-structure principle requires that the description of the objects should not implicitly assume the context in which the component is placed or the function the device is supposed to achieve. The locality principle requires that components interact only across physical connections. A simple example illustrating how this approach works can be found in [Ref. 10].

In ENVISION, causal relations among variables are determined based on the way values are propagated among them through equations in the process of predicting behavior. Thus, if the known value of [x] is propagated through an equation to determine the value of [y] appearing in the equation, the value of x is considered to be the cause of the value of y[2]. Since the description of each component cannot mention a variable belonging to another component, making connection equations the only place where variables belonging to multiple components can appear within one equation, components causally interact directly through local connections only.

A significant difference between ENVISION and QStruc is in whether or not the system infers the propagation paths of possible causal interactions and what and how information is propagated along them. In ENVISION, components can interact only through known physical connections, and the types of information a physical connection can propagate is known a priori. Connections simply propagate values without processing. Given this information, ENVISION determines the behavior and the causal paths along which the given values are actually propagated. In contrast, QStruc must first determine whether a path of propagation exists and whether a process or a behavior that allows information to be propagated takes place and why, before it can analyze behavior. Furthermore, the restricted role that connections play in ENVISION is insufficient to model the kinds of physical connections that QStruc must handle. Therefore, we enhanced the function of connections so that (1) they can process information, and (2) they may change their type as a consequence of qualitative interpretation and affect the overall structural behavior.

D.2.2 QSIM

QSIM [Ref. 12, 13] simulates behavior from a set of qualitative constraints among state variables. To use QSIM to model a physical device, a user must describe its

---

[2] ENVISION also talks about another type of causality, namely causal relations among states, which is based on the way derivative values in one state determines how the system transitions from the state to another. We will not discuss this type here because we are not concerned about behavior over time in Qstruc.

behavioral properties in the form of qualitative variables and constraints. The user must also specify the quantity space for each variable. The value of a qualitative variable in a state is a pair - <*qval, qdir*>, where *qval* is the qualitative value (an interval) and *qdir* is the direction of change (increase, decrease or steady). Given a set of qualitative variables, constraints, and some initial values, QSIM predicts all the possible courses of behavior over time, producing a tree of qualitative states with the initial state as the root. QSIM's qualitative constraints, unlike confluence equations in ENVISION, are directed in the sense that they specify the direction of causality among variables. A simple example illustrating how this approach works can be found in [Ref. 10].

The work on QSIM itself does not address the problem of modeling since the model is given as an input to QSIM in the form of qualitative constraints. The issue of modeling was addressed in a later extension to QSIM, called CC [Ref. 6]. CC takes the process-oriented approach of Qualitative Process Theory (described in the next subsection) to generate a model in the form of a set of qualitative constraints from a component-connection representation of a physical system.

D.2.3 Qualitative Process Theory (QPT)

Engineering design must be concerned with not only what components are present but also with the physical processes that can take place in these components. Forbus' Qualitative Process Theory (QPT) addresses this aspect by explicitly representing physical processes and reasoning about their occurrences [Ref. 5]. QPT takes the view that physical interactions happen through processes and that processes are the medium through which changes take place. Processes specify not only functional relations between variables but also direct influences processes have on variables.

For a physical process to take place, a certain condition must hold in the world, and when it does take place, the process has certain effects on the state of the world. Any definition of a process must specify such preconditions and consequences. In QPT, a process is defined in the following five parts [9]:

1. *individuals*:: The set of objects or other processes participating in the process.

2. *preconditions:* Non-quantitative conditions on the individuals that must be present for the process to take place.

3. *quantity conditions:* The conditions on the quantitative attributes of individuals that must hold for the process to take place.

4. *relations:* The functional relations that hold among quantities while the process is taking place.

5. *influences:* The direct influences (increase or decrease) of the process taking place on quantity values.

Objects (or sets of objects) are also represented in the same manner as processes, except for the fact that objects do not have influences. The representations of objects are called *individual views* in QPT. Given an input description of the state of the world in terms of a collection of individual views, the task of Qualitative Process Engine (QPE -- the implementation of QPT) is to determine all the possible processes that can take place in the situation, and predict the behavior of the system in terms of various processes occurring or dying out over time along with the changes in qualitative variable values caused by the processes. The direction of causal relations among variables are pre-specified explicitly in the relations and influences of process definitions.

A significant characteristic of QPT is that the concept of physical processes is as central to modeling of a physical system as the concept of objects. QPT provides a means to represent and reason about processes. In QPT, objects interact through processes, which are the sole medium of changes. Behavior prediction in QPT involves first identifying processes that allow objects to interact. In ENVISION, in contrast, identifying interactions is straightforward because components can only interact through local connections, and each connection specify exactly what and how information is propagated between the components. In order to model a behavior correctly in QPT, one must make sure that all relevant processes and their interactions are found. In many domains such as analysis of building structures and mechanical devices, the component-oriented approach alone is not sufficient since identifying the types and locations of interactions is an important part of reasoning about behavior in these domains.

D.2.4 Causal Ordering

The work on Iwasaki and Simon [Ref. 11] on causality in device behavior has a different goal from the three discussed so far. The purpose of the work is to provide a means to determine the causal dependency relations among variables in a model comprised of equations, which are, by themselves, "acausal" representations of functional relations among variables. Their approach is based on the theory of causal ordering originally proposed by Simon in the field of econometrics [Refs. 16, 17]. The work by Iwasaki and Simon extended the scope of the theory and applied it to the domain of physical systems.

In order to apply the method or causal ordering to determine causal relations, one must have a model in the form of a self-contained set of n equations in n variables. Each equation in this set must represent a conceptually distinct mechanism, such as a process, a physical law, or a component function, in the system being modeled. Such equations are called *structural equations.* This set of equations may also include equations of the form vi = c, where c is some constant, for each variable vi that is determined by forces external to the system. Such variables are called *exogenous variables.* Given a self-contained system of structural equations including exogenous variables, the theory of causal ordering allows one to determine the causal dependency relations among the variables in the system by repeatedly identifying the subset of variables whose values can be determined independently of other variables and removing them from the system.

Since this procedure for determining causal ordering looks only at what variables appear in each equation but not at the exact form of the equation, it applies to both qualitative and ordinary equations. At the same time, the resulting causal dependency relations depend strongly on the choice of equations included in the model. Iwasaki and Simon show that different sets of equations produce different causal relations even if the sets are mathematically equivalent. However, if each equation is a structural equation, the derived causal relations will agree with the model builder's intuitive notion of causality in the situation. Therefore, the method of causal ordering can be used to reveal the causal dependency relations implicit in the choice of structural equations and exogenous variables.

Since the directed graph produced by applying the method of causal ordering also indicates the order in which the set of equations can be solved, we have used causal ordering in QStruc as a way to identify a solution path for solving qualitative equations to derive the qualitative behavior of the structure. We have also found that causal ordering helps the user to visualize the dependency among variables and to understand the impact of a desired change in a variable's value.

D.3 Summary

The final issue that we discuss in this section is the problem of ambiguity in qualitative reasoning. Ambiguity can arise for several different reasons:

(1) There may not be sufficient number of qualitative equations for the number of variables.
(2) Qualitative arithmetic operators do not necessarily produce unique values.
(3) During qualitative evaluations, since it is not known which one of the two variables that are both changing in one state will reach a landmark value first, the prediction may be ambiguous allowing the possibilities of either or both of them reaching a landmark value in the next state

In case (1), where there are fewer equations than unknown variables, some of the variables must be determined by mechanisms external to the system in question. Identifying which variables are exogenous allows one to make appropriate assumptions to reduce ambiguity, such that their values are not changing unless known otherwise.

Cases (2) and (3) are inherent problems in reasoning without precise numerical data. Case (3) is not a concern to us here because QStruc does not predict behavior over time. Ambiguity of Case (2) arises, for example, when we try to determine the value of (z) given

$$(z) = (x) - (y),$$

and

$$(x) = (y) = +.$$

We have tried in QStruc to reduce this type of ambiguity by using Quantity Lattice [Ref. 18] to reason about the relative magnitudes of the variables.

Another type of possible ambiguity is ambiguity in causal relations. When qualitative equations are solved by value propagation and when the direction of causality is determined based on how values are propagated, the same set of equations can give rise to different causal dependency structures, especially when there is a set of inherently simultaneous equations that cannot be solved by simple value substitution. Simultaneous equations indicate the existence of a feedback loop in the system. Such equations must be solved by making assumptions about the value of one or more of the variables. If the assumptions lead to a set of values that are consistent with all the equations, the set represents a solution. If the assumptions lead to a contradiction, they must be discarded. Due to ambiguity of Case (2) discussed above, there may be multiple solutions. Also, since one can choose the variables to make initial assumptions about, and since the causal interpretation produced depend strongly on the assumptions, there can be multiple causal interpretations even for one solution. deKleer and Brown employ heuristics for deciding what assumptions to make in these situations, which also reduces the number of causal interpretations produced. The theory of causal ordering, on the other hand, does not assign causal relations among variables in an inherently simultaneous set of equations. In general, determination of the direction of causality around a feedback loop requires more detailed knowledge of the dynamic behavior of the mechanisms involved in the loop.

In this appendix we introduced some of the basic concepts in qualitative physics and discussed some notable approaches and their relevance to our research. We have built on ideas from these earlier work in qualitative physics in developing QStruc. QStruc takes a component-oriented approach in representing building structures in terms of components (structural members) and connections. However, connections in QStruc have much more complex behavior and are much more varied in types than connections in ENVISION, reflecting the variety of types of joints and their significance in determining the behavior of the overall structure. QStruc also employs the concept of physical processes specific to the structural analysis domain to link the structural features to their behavioral characteristics. Detection of certain processes also prompts QStruc to refine the structure model by decomposing a structural component into multiple sub-components, where such decomposition facilitates the analysis even though no corresponding physical decomposition exists.

The concept of causality plays an important role in all the works discussed in this section. QStruc performs causal ordering analysis on the model it generates to help the user understand the dependency relations among variables. The result is also used to determine the order for solving qualitative equations to derive the behavior.