1984

Design of a pedagogical aid for civil engineering education

Daniel R. Rehak
Carnegie Mellon University

Thomas V. Schields

DESIGN OF A PEDAGOGICAL AID FOR CIVIL ENGINEERING EDUCATION

by

D.R. Rehak and T.V. Schlelds

DRC-12-21-84

December, 1984

Design of a Pedagogical Aid for Civil Engineering Education

Daniel R. Rehak
Assistant Professor, Department of Civil Engineering
Camegie-Meilon University, Pittsburgh, PA. 15213

Thomas V.SchiekJs
Engineer, Bechtef Power Corporation
Gaithersburg, MO.
formerly Graduate Research Assistant, Department of Civil Engineering
Camegie-Mellon University, Pittsburgh, PA. 15213

Abstract

One major aspect of Civil Engineering education is the presentation of the behaviors of physical systems or models of systems. The principles of these systems are repeatedly demonstrated through a wide variety of problem applications. Learning is enhanced through increased problem solving opportunities, but instructors have a limited ability to generate a variety of realistic, unique problems for drill, homework, and examinations. The pedagogical aid is presented as one approach to computer based education, designed not to instruct, but to provide a self-paced, interactive problem generation and problem explanation facinty which supplements end-oNchapter textbook assignments. The intent is not to substitute for an instructor, or to provide programmed learning, but to provide access to unlimited problem solving drill, addressing the need outlined above.

Current pedagogical aids suffer from two drawbacks. First, they are custom built. Thus the costs associated with developing a new set of courseware is prohibitive. Second, those systems which have been developed operate on targe time-shared computers. They suffer from ail the drawbacks of such a computing environment, including lack of access to the computing resource due to high demand. With the need for pedagogical aids and their current status as background, the architecture of a domain independent program which is designed to operate in a networked, powerful personal computer environment is presented.

Using the pedagogical aid, the student is presented a custom generated problem with a specific level of difficulty (similar to an end-of-chapter textbook assignment). The problem generation paradigm is based on a hierarchical selection of problem topology, components, and parameters. The range of valid selections, and constraints on the selection process, are problem domain dependent and this data is used to drive a general problem generation algorithm. The problem and a variety of questions are presented, utilizing graphics for problem visualization whenever possible. The computer then serves as as electronic desk, providing a calculator and sketchpad for use in problem solving. The student's results are then submitted to the program for checking, and the process proceeds through a set of questions and problems. The program incorporates additional facilities for computer managed instruction, and an instructor interface. The various components which comprise the pedagogical aid, its operation, and the computer environment in which it is used are described.

Keywords: Computers; Education; Software; Pedagogical aid; Computer aided instruction; Personal computer, Problem generation; Workstation.

Prepared for the *Third Conference on Computing in Civil Engineering,* American Society of Civil Engineers, San Diego, California, April, 1964.

Design of a Pedagogical Aid for Civil Engineering Education

Daniel R. Rehak[1] and Thomas V. Schields[2]

1. Introduction

One major aspect of Civil Engineering education is the presentation of the behaviorsof physical systems or models of systems. The principles of these system are repeatedly demonstrated through a wide variety of problem applications. Learning is enhanced through increased problem solving opportunities, but instructors have a limited ability to generate a variety of realistic, unique problems for drill, homework, and examinations. Pedagogical aids provide assistance to the instructor in generating unique problems for drill, homework, and tests. Current aids are limited in that each can only generate problems from a single domain. A methodology and a prototype system which generates many unique, realistic problems from different domains within Civil Engineering is described herein [8]. Students access the pedagogical aid through an educational workstation. Each workstation, a powerful personal computer with a high resolution graphics display and graphical input device, is connected via a high speed communications network [5]. This computing environment provides new opportunities in the area of computer based education due to the dedicated resources and the high quality graphics display usually not available on a time-sharing system [6].

1.1. Computer Based Education

Computer Based Education *(QBE)* describes the use of computing resources in the educational process. Computer Aided Instruction (CAI) tools, one aspect of CBE, are designed to assist the instructor. One major goal for any CAI tool is to provide opportunities for problem drill, exercise generation, and test generation. Within the context of a true teaching aid, the program must exhibit some intelligence. Several significant CAI systems relevant to this work are discussed below.

One CAI system using artificial intelligence (AI) techniques is *SOPHIE* [2], an electronic trouble shooting educational monitor. The student is presented a circuit with a fault. He asks for information about the circuit and then makes a hypothesis to correct the fault *SOPHIE* checks the hypothesis and provides help based on the information the student requested. *SOPHIE* demonstrates how AI techniques can be used to implement an intelligent pedagogue. As a teaching aid it has one severe drawback in that it understands only one predefined, preprogrammed circuit

Steamer is a program developed to instruct U.S. Navy personnel in the use of steam propulsion [9]. *Steamer's* major goal is to produce an inexpensive desktop computer system for use in propulsion plant operation instruction. It dedicates substantial computational resources to a single user while incorporating high-quality color graphics, knowledge-based instruction and comprehensive simulations, and provides a better computational environment for education than a conventional time-shared machine.

Other programs provide more problem drill. These programs are not intelligent pedagogues, but merely pedagogical aids, programs which provide problem drill, exercise generation, and test generation, but not instruction. They provide a self-paced interactive problem generation facility which supplements end-of-chapter textbook assignments. One such system is SSSDS [3], which generates statically

[1] Assistant Professor. Department of Civil Engineering, Carnegie-Mellon University. Pittsburgh. PA. 15213

Engineer. Becthel Power Corp.. Gaithersburg, MD., formerly Graduate Research Assistant. Department of Civil Engineering. Carnegie-Mellon University. Pittsburgh. PA. 15213

determinate structures problems. A similar program is *Ustaad* [7], wherein the student can test his knowledge of d.c. electrical circuit analysis at a self-determined pace. Like SSSDS, *Ustaad* contains a random synthesis mode which generates many unique circuit problems. Both SSSDS and *Ustaad* have improved on the major shortcomings of other CAI tools in that they provide assistance to the instructor in generating problems. However, ail current pedagogical aids are course domain specific, and the costs associated with developing a new set of courseware are prohibitive.

2. Architecture of a Pedagogical Aid

A pedagogical aid must offer the student unlimited drill opportunities, and the instructor the facility for generating a wide variety of realistic problems. From the standpoint of the student, the pedagogue must offer

- *Problem generation* ~ the student must be presented with a realistic problem of a certain hardness,
- *Problem visualization* •- the graphic presentation of many engineering problems is essential "to the learning process, and
- *Problem solving assistance* •- an interactive question and answer and explanation facility must be a primary feature of any aid designed to replace or assist the instructor.

From the standpoint of the instructor, control within these areas must be provided:

- *Problem generation* -- the instructor must be able to specify many unique problems,
- *Problem visualization* ~ the instructor can specify various graphical presentation aspects of the problem, and
- *Problem solving assistance* ~ the instructor can embody some of his knowledge, including the correct solution technique and clear, concise descriptions of various aspects of the problem.

The pedagogical aid described below is designed to provide these features. *

2.1. Problem Representation

Figures 1 shows a number of typical end-of-chapter problems from different texts [1,4]. As one examines many such problems, a distinct pattern of similarities and differences is discernible. The problems in Figures 1-a through 1-d are statically determinate one span beam problems, while the problem in Figure 1 -e is a statically indeterminate two span problem. This problem is conceptually more difficult than the first four problems, due to the additional span. The problems in Figures 1-c and 1-d appear similar in overall form. The support conditions on the left end are either a hinge or a fixed support, and on the right end either a roller or a free end. Finally, the problems presented in Figures 1 -a and 1 -b also are similar, differing only in the magnitude and location of the loadings and in the overall dimensions of the beam. Therefore, the similarities and differences can be classified as:

- *topology* - the overall configuration of the problem, or the relationship and connections between the individual components of the problem differs. The first four problems are topologically similar, while the last problem is different due to the additional beam, support, and load.
- *component* -- the physical components of a topology may vary. For the problems in Figure 1, the set of components includes - beams, point loads, uniform loads, and roller, hinge, fixed, and free supports. In Figures 1-c and 1-d, the hinge, roller, free end, and fix support are all examples of components which vary for the same problem topology.
- *parametric value* -- the problems in Figures 1-a and 1-b are topologically identical, and have the same components, but the loadings are of a different magnitude, and the beam dimensions vary. This numeric change of parameter values makes the problems distinct.

Analogies to the above can be found in.many Civil Engineering problem domains [8].

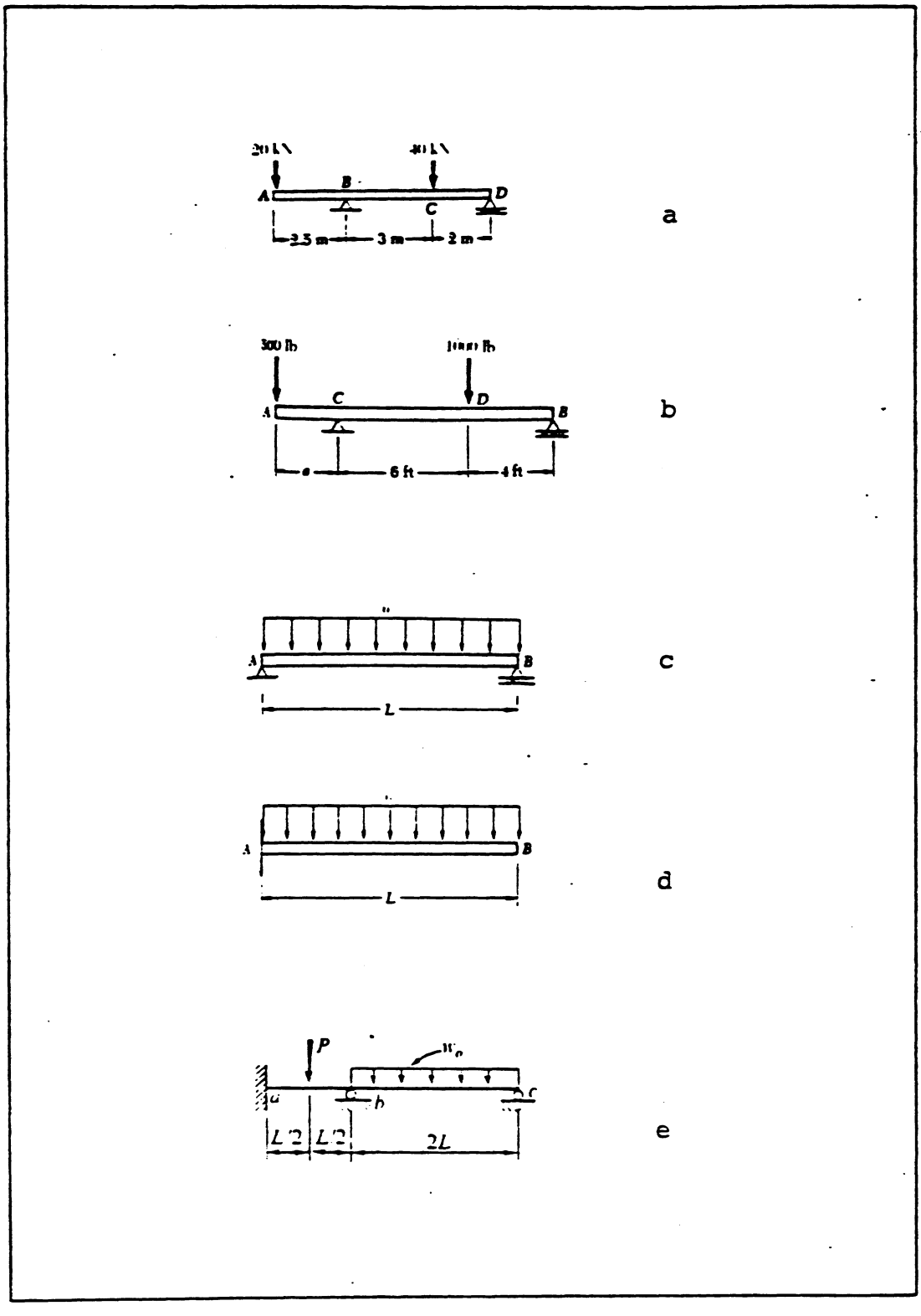


Figure 1: Structural Mechanics Textbook Problems

2.2. Hierarchical Problem Generation and Representation

Many unique engineering problems can be generated using the hierarchical generation model illustrated in Figure 2.

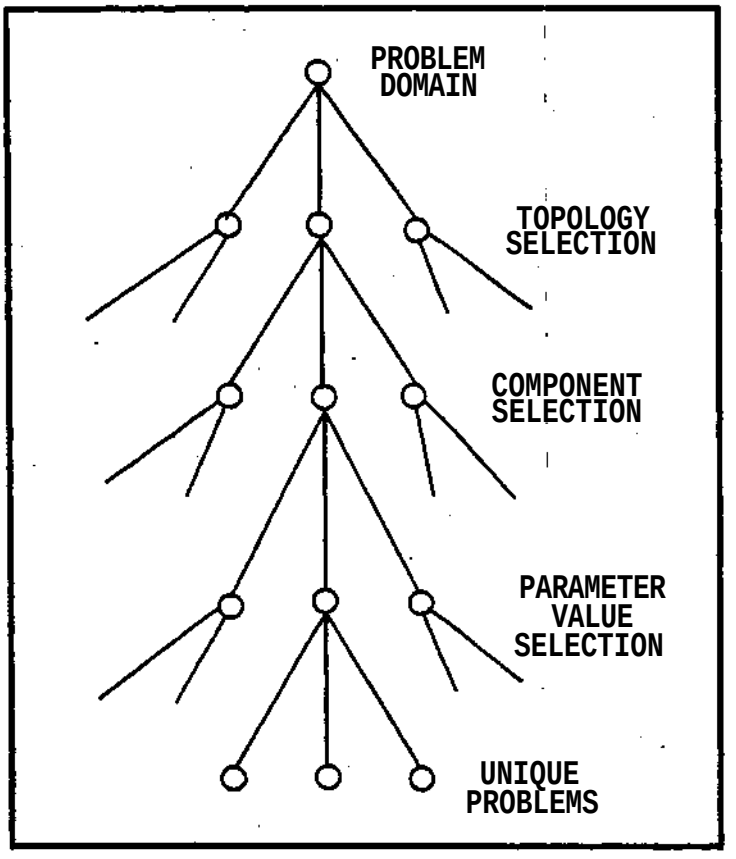


Figure 2: Hierarchical Problem Generation Model

A problem can be represented by a certain topology. The number of components and connections in the problem defines the topology, however the types of components and connections may vary tor the same topology. In addition, for any given component of a topology, the numerical values of parameters associated with the components may vary. Therefore, generation of a unique problem requires: (1) selection of a topology, (2) for that topology, selection of component and connection types, and (3) for those components, selection of parameter values. By traversing the tree structure, a unique problem can be generated. As the number of topologies and available components increase, the combinations increase accordingly. In addition, the selection of different numerical values at the parameter level increases the number of unique problems.

As an example of this traversal, consider Figure 3. It depicts, in general terms, the tree structure required to generate the problems in Figure 1. The first step is to select a topology. Choices are four, five and seven component topologies. Once the topology is selected, each type of component must be chosen. Consider the four component topology. By varying the choice of components, the problems in Figures 1-c and 1-d are generated. Finally, for each component numerical values are selected. By varying the actual parameter values for the point loads in the five component topologies, different problems are generated (Figures 1-a and 1-b).

Problem Topologies. The mechanism used to describe a particular topology is based on formal network techniques, where a problem is represented as a series of *nodes* and *links.* A problem may be thought of as a number of pieces, or components, and their connections. Each component is

represented as a node, while each link describes the manner in which these components or nodes are joined. The instructor specifies the topology used to represent a problem.

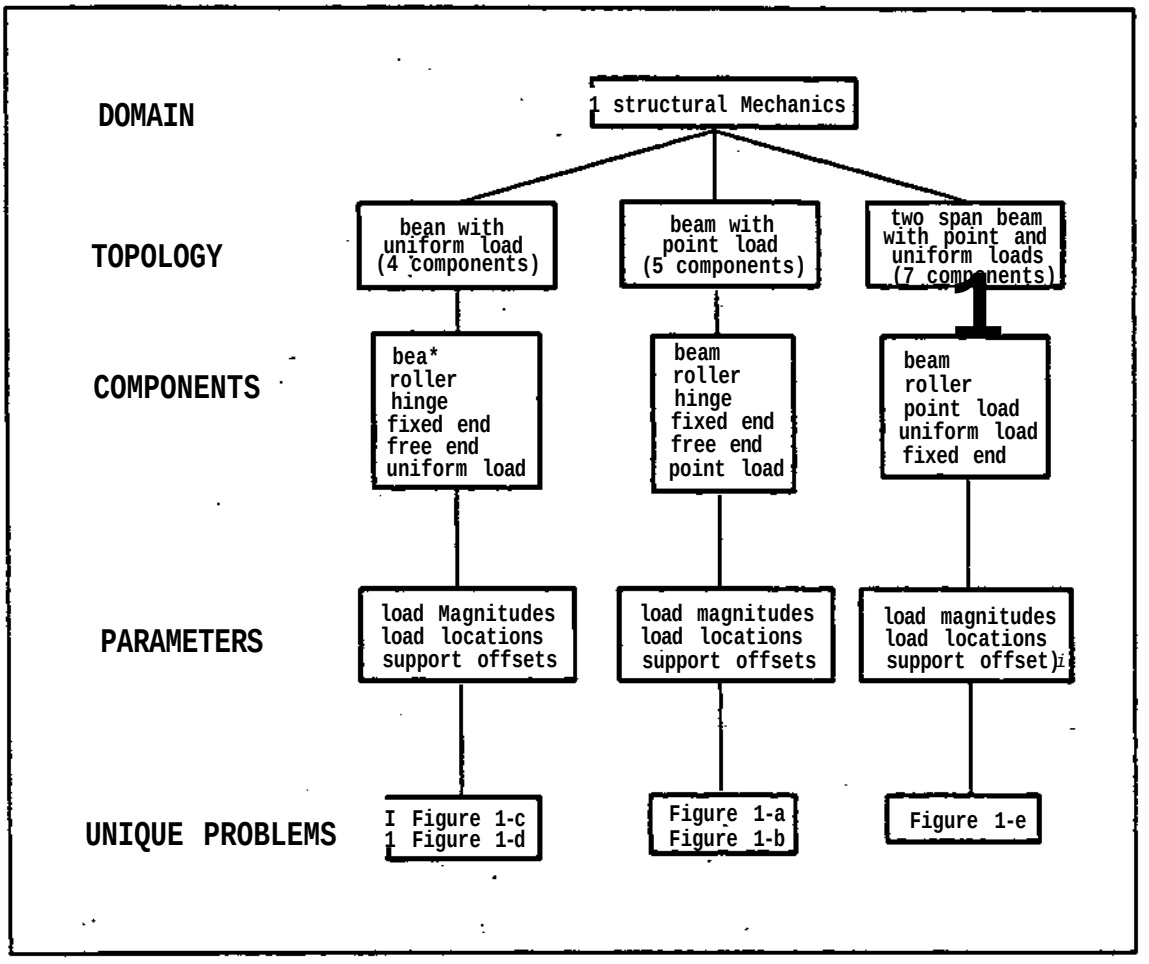


Figure 3: Structural Mechanics Generation Example

Problem Components. Each node or link is an *instance;* the specific occurrence of a component within a problem. Each particular node or link of a topology may have a number of available instances. By selecting one instance from the available choices, a unique problem is defined. The problem generator contains a number of predefined instance types: points, lines and bodies, with classes withtn each type (i.e. straight line, curved line, rectangular body, circular body, etc.). Actual components, such as a beam, are instances of some type and class, e.g. a beam is a straight line node. Similarly, a set of link instances define the geometric details of all nodal connections.

Problem Parameters. There are two types of parameters, *global* and *instance* parameters. Global parameters deal with some global aspect of the entire problem, not just some aspect of one of the components of the problem. An example is the maximum bending moment in a structure. Instance parameters can be classified as *geometric* or *internal* parameters, and *external* parameters. Geometric parameters are related to the type of instance. Certain parameters are associated with the nodal and link instances simply due to the geometric interpretation of each instance. For example, a line type node has geometric parameters length and orientation. External parameters are problem specific, e.g. a beam instance might have a moment of inertia parameter because it is a beam, not because it is a line type instance. Each parameter has an associated numerical value. The value is based on the geometry of the problem, it is computed, or it is specified externally by the instructor.

2.3. Unique Problem Generation

The process of generating a problem is a tree traversal, selecting one path from the root node to a unique problem leaf node. First a topology is selected. For that topology, one instance is selected for each node and link. For each node and link instance, the parameter values are selected. Selection at each level is based on *hardness.* Each alternate topology, component, and range of parameter values has an associated level of difficulty. The problem generator selects from the available options those items which yield a problem close to the desired hardness. Once selected, the problem and a set of related questions are presented to the student for problem solving and drill.

3. Example

This section presents an example using the problem generation paradigm concepts discussed above [8]. The example is a one span beam with one varying length overhang, different types of supports, and two alternate loading conditions. The example focuses on one topology with two nodes having multiple instances (the alternate left support and loading condition), all other nodes and links having one available instance, and different node and link parameter values (the right support location and load location). The instances and their locations are shown in Figure 4 (double arrows imply the instance may have different locations). The topology and the nodal instances are shown in Table 1.

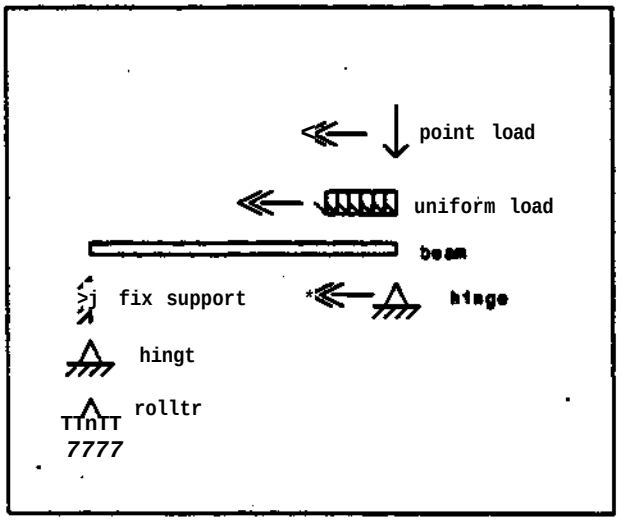
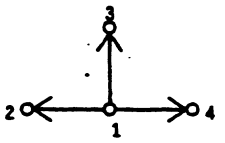


Figure 4: Available Instances and Locations

Node 1, the beam, is a line instance. Nodes 2 and 4 are points which correspond to the supports and are linked to the beam. Node 3, either a uniform load (a line instance) or a point load (a point instance), also connects to the beam. Table 2 lists some parameters associated with this problem. External parameter values which are specified include the magnitudes of the point and uniform load components. Internal parameters which are specified include the length of the beam and uniform load components, and the offset parameters associated with each line to point adjacent connection (the support to beam connection location and the load to beam connection location). Internal computed parameters such as the support reactions also exist.

Some unique problems generated by the problem generator are shown in Figure 5. The example allows two of the four nodal instances to vary, along with the provision of different numerical parameter values for the location of the instances. This results in sixty unique problems. This example does not exploit the paradigm to its fullest potential. The addition of multiple topologies, providing more alternate instances for each node, and increasing the allowable values for each parameter would yield additional unique problems. If the paradigm were exploited to its fullest potential, virtually an infinite number of unique problems would result.



NETWORK REPRESENTATION

	NODE			
	1	2	3	4
AVAILABLE INSTANCES	beam	fix support	point load	hinge
		roller	uniform load	
		hinge		

Table 1: Problem Topology with Available Instances

PARAMETER NAME	CORRESPONDING INSTANCE	PARAMETER TYPE	PARAMETER VALUE
W	uniform load (node 3)	external specify from set	one set element 10 k/ft
P	point load (node 3)	external specify from set	one set element 100 k
length	uniform load (node 3)	internal specify from set	one set element 3.33 ft
	beam (node 1)	internal specify from set	one set element 10.0 ft
orientation	uniform load (node 3)	internal specify from set	one set element 0 degrees
	beam (node 1)	internal specify from set	one set element 0 degrees
off1	link 1 -> 3 beam to point load OR beam to uniform load	internal specify from set	two set elements 0.333 or 0.667
	link 1 -> 4 beam to hinge	internal specify from range	range 0 to 0.4 by 0.1

Table 2: Parameters for Example

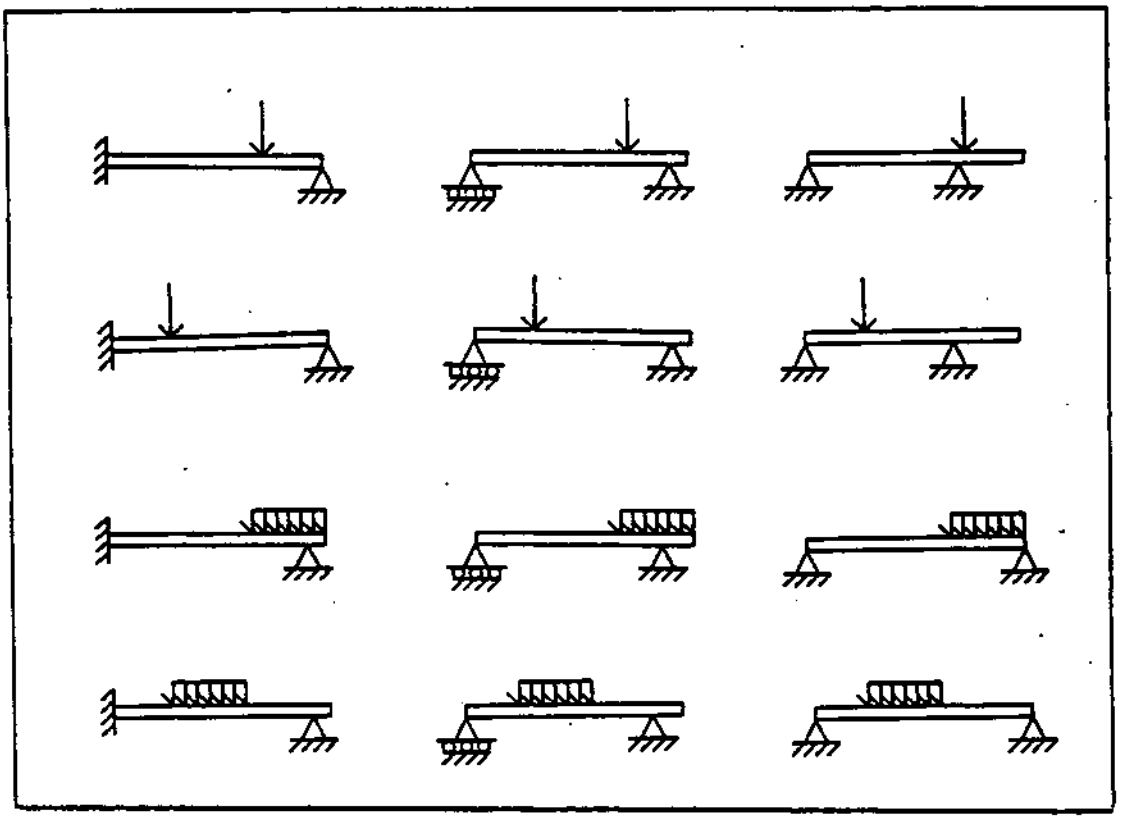


Figure 5: Unique **Examples**

4. Educational Workstation

The pedagogical aid is part of a complete educational workstation which provides an electronic desk for problem solving and drill [6]. The software is implemented on a networked powerful personal computer (PERQ) with the following characteristics:

32 bit virtual address space
1 ** primary memory
30** disk

1024*768 bitmap raster display
digitizer tablet with puck
10 Mbit ethemet connection

The display and graphical input supported by the the digitizer and puck are used to implement a "seeing and pointing- environment. A set of icons, representing possible actions such as sending or receiving electronic mail via the network, using a calculator, and using a electronic sketchpad are provided. The student can access these and other capabilities simply by pointing to the corresponding icon. The dedicated computing resources and utilization of high resolution, interactive graphics provide a computing environment which is more responsive, user friendly, and better suited to the needs of an educational application than a traditional time-sharing machine.

An example of the display of an educational workstation is illustrated in Figure 6. The screen is divided into several windows. The central screen area is a set of overlapping windows, representing pieces of paper on a desk top. Since the quantity of information to be displayed exceeds the screen size, all the windows scroll and can be moved. The corresponding scroll bars indicate the relative portion of the display which is currently visible. An interactive typescript (labelled text window) with screen editor, shown in the bottom of the display, handles all textual input/output. Icons for electronic mail, the calculator, the sketchpad and other functions are shown on the right of the display in the icons window.

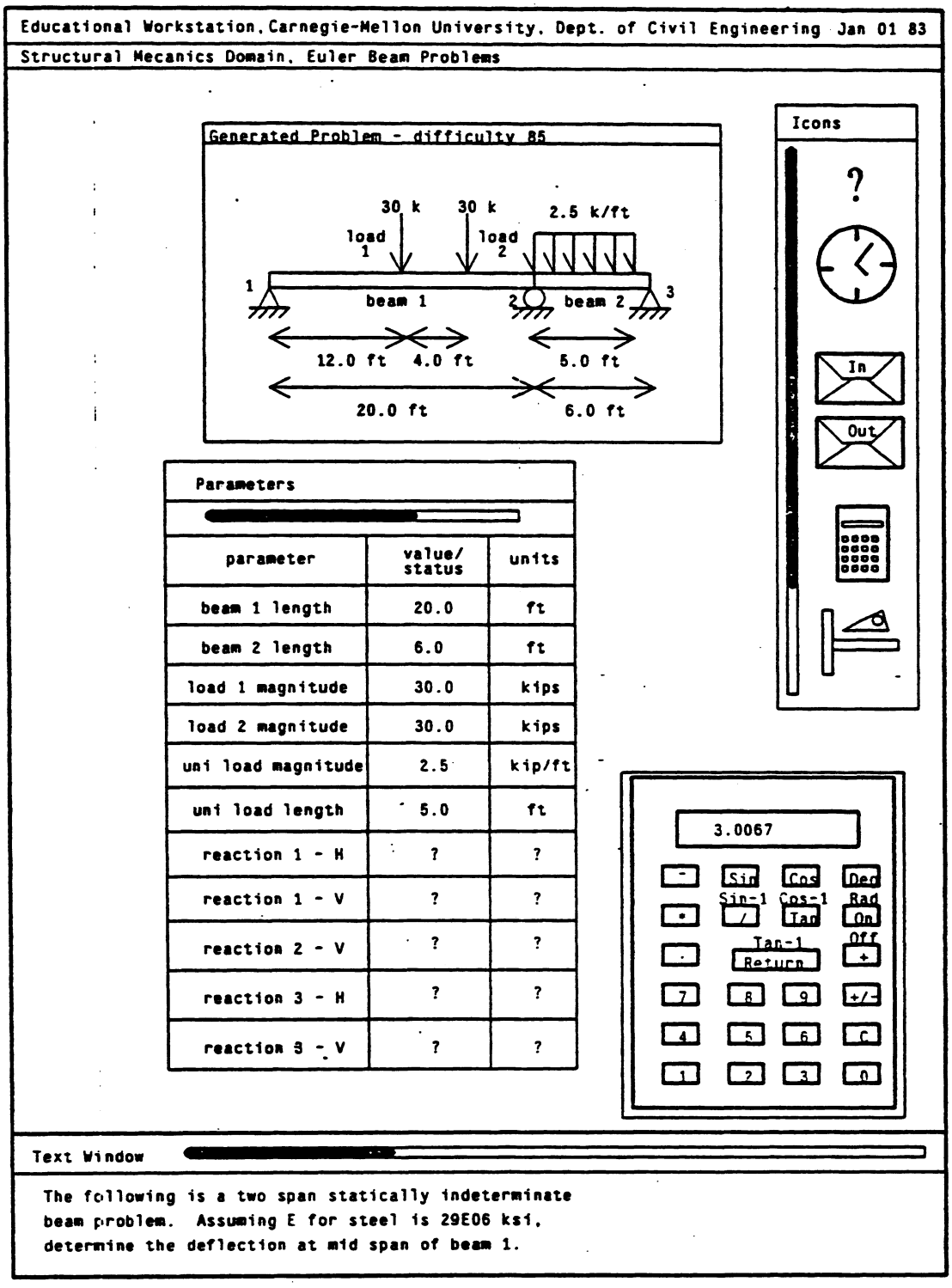


Figure 6: Display of an Educational Workstation

Using the workstation, the student is presented with a unique problem from a particular domain, as shown in the upper portion of the display. Below the problem, parameters applicable to the problem are displayed. As the student computes parameter values, these are placed in the table and are checked. Any value can be used in further computations. A variety of questions regarding different aspects of the problem are generated and presented to the student. All such text input/output is handled by the text

editor. A graphical calculator is provided to assist in numerical computations. The student operates the calculator with the digitizer puck, pointing to keys and pressing buttons, just as on a real calculator. Values may be freely interchanged between the calculator, the parameter table, and the other displays via the puck, eliminating the need to manually type or copy values. As the student proceeds through a set of problems, his progress is recorded and automatically forwarded to the instructor via network electronic mail. Thus the workstation provide a complete electronic environment for educational problem solving and drill.

5. Summary

The problem generation strategy presented appears to be applicable to many problems in Civil Engineering. The idea of problem representation using the concepts of topologies, instances, and parameters has been applied to problems in structural mechanics, statics, solid mechanics, and engineering economics. The representation of many typical end-of-chapter textbook problems has been reproduced using this prototype system [8]. In addition, the problem generation paradigm provides the capability for the instructor to generate a virtually limitless number of unique realistic problems for drill, homework, and examinations. The prototype, along with the Educational Workstation, provide an example of a next generation, personal computer based educational environment

Acknowledgements. This work is supported in part by the National Science Foundation under Grant SPE-8263002.

6. References

1. Beer, *P.P.,* and Johnston, E.R. *Vector Mechanics for Engineers - Statics,* third ed., McGraw-Hill, Inc., 1977.
2. Brown, J.S., Burton, R.B., and de Kleer, J. "Pedagogical, Natural Language and Knowledge Engineering Techniques in SOPHIE I, II, and III,** In Sieeman, D. and Brown, J.S. (editors), *Intelligent Tutoring Systems,* Academic Press, 1981.
3. Maher, ML. "SSSDS: A Computer Aided Instruction Tool for the Synthesis and Solution of Statically Determinate Structures," Technical Report DRC-01-06-81, Design Research Center, Carnegie-Mellon University, October, 1961.
4. McLean, W.G., and Nelson, E.W. *Schaum's Outline of Theory and Problems of Engineering Mechanics Statics and Dynamics,* third ed., McGraw-Hill, Inc., 1978.
5. Newell, A., Kiesier, S., Sproull, L, and Walter, B. "Research on Computing Environments at Camegie-Meilon University," Carnegie-Mellon University, February 1983.
6. Rehak, D.R. "The Use of Powerful Personal Computers in Engineering Education," 1963 ASEE Annual Conference Proceedings, Vol. 1, pages 165-169.
7. Samad, T. "Ustaad - An Interactive Pedagogical Aid for Elementary Electronic Circuit Analysis," Technical Report DR018-43-82, Design Research Center, Camegie-Meilon University, April, 1982.
8. Schields, T.V., "Problem Generation Strategies in a Computer Based Pedagogical Aid," Technical Report DRC-01-16-84, Design Research Center, Carnegie-Mellon University, 1984.
9. Stevens, A., Roberts, B., Stead, L "The Use of a Sophisticated Graphics Interface in Computer-Assisted Instruction," *IEEE Computer Graphics and Applications,* pages 26-30, IEEE, March/April, 198a