CIFE CENTER FOR INTEGRATED FACILITY ENGINEERING

SightPlan: A Blackboard Expert System for Constraint Based Spatial Reasoning About Construction Site Layout

by Raymond E. Levitt, Iris D. Tommelein,
Barbara Hayes-Roth, and Tony Confrey

TECHNICAL REPORT
Number 20

October, 1989

Stanford University

TABLE OF CONTENTS

SightPlan: A Blackboard Expert System for Constraint-Based Spatial Reasoning About Construction Site Layout

Raymond E. Levitt[1], *Iris D. Tommelein*[2], *Barbara Hayes-Roth*[3], *and Tony Confrey*[4]

1. Abstract

SightPlan is an expert system that lays out temporary facilities on construction sites. It demonstrates how one can closely model the steps taken by a person performing layout design, and how interactive graphics combined with an expert system can augment human decision-making. This report describes site layout practice and reviews the state of the art of layout modeling. The work on layout modeling fits within the larger context of spatial reasoning and generic design, so SightPlan can be related to ongoing research in other domains. SightPlan builds upon the domain-independent BB1 blackboard architecture and uses the ACCORD language for constructive assembly.

Three SightPlan strategies are applied to two case studies of power plant construction (the Intermountain Power Project and the American 1 project). An *early-commitment strategy* models the way human experts conduct site layout and produces a layout solution that satisfies all constraints. A *temporal strategy* tests and validates the early-commitment strategy and extends it by explicitly representing and reasoning about objects over time. This allows the global site layout to be animated as a sequence of layouts over discrete time intervals. A *postponed-commitment strategy* takes advantage of computer capabilities by delaying commitments and then heuristically sampling possible positions to find several satisfying solutions. While these solutions meet hard constraints, they appear "chaotic"; that is, a person viewing them is likely to introduce additional constraints. SightPlan can introduce such constraints at run-time, or a user of the SightView graphical interface can introduce such constraints interactively and feed that information back to SightPlan for further reasoning.

KEYWORDS: artificial intelligence; expert systems; knowledge-based systems; blackboard architecture; spatial reasoning; constraints; interactive graphics; site layout; construction engineering; construction management; temporary facilities; design.

[1]Professor, Department of Civil Engineering, Stanford University, Stanford, CA 94305-4020

[2]Assistant Professor, Department of Civil Engineering, University of Michigan, Ann Arbor, MI 48109-2125

[3]Senior Research Associate, Department of Computer Science, Stanford University, Stanford, CA 94305-0107

[4]Research Assistant, Department of Computer Science, Stanford University, Stanford, CA 94305-0107

2. Introduction

This paper presents the goals, approach, experiments, and results of a three-year study aimed at using artificial intelligence techniques to model the kinds of expertise used by construction engineers in laying out the temporary facilities on construction sites. The project was organized as a collaborative effort between civil engineers interested in exploring the use of artificial intelligence techniques for construction site layout, and computer scientists interested in testing and generalizing ideas about architectures for spatial reasoning and opportunistic planning. The research resulted in SightPlan, a knowledge-based system for site layout, that builds upon a blackboard architecture, named BB1.

We give the background of the research project, our reasons for choosing the software architecture and tools, and the rationale for the experiments that we decided to conduct. Next, we describe the organization and operation of the BB1 blackboard expert system architecture used for SightPlan's development. The main body of this paper discusses the implementation and results of a series of SightPlan prototype systems in which alternate problem-solving strategies for site layout are contrasted. These different prototypes model contrasting strategies which exploit human vs. computer cognitive strengths. We conclude with an evaluation of the experimental results and we suggest directions for future research. A more detailed description of this work can be found in [Tommelein 89c].

2.1. Background

Construction site layout consists of identifying the temporary facilities needed to support construction operations, determining their size and shape, and positioning them within the boundaries of the available site or remote areas. It is a multi-objective design problem that has not been effectively addressed by linear programming or related optimization techniques to date.

We were interested in laying out temporary facilities on a construction site because 1) it is a problem that is addressed routinely by construction managers and that substantially influences the efficiency of construction operations on a site, and 2) there do not exist formal methods for construction managers to assist them in this task. We assumed that modeling the layout process used by practitioners would provide us with insights that would facilitate the development of computer decision-support tools for this and related problems.

In this section we first define the site layout problem. Next, we sketch how construction field managers go about laying out sites, and we provide a brief overview of modeling techniques that have been used to solve layout problems.

2.2. Problem Definition

SightPlan positions temporary facilities such as parking lots, construction office trailers, and material laydown areas. These facilities remain on site for periods ranging from a few days to several months or even years, depending upon whether they support a single construction activity, a construction phase, or the entire construction enterprise. At times, "temporary" facilities may not be dismantled after their use in construction. For instance, construction warehouses are often reused as maintenance facilities during operation of a completed facility.

There are many factors that influence what facilities are needed on a given project and how they should be laid out. Neil, among others, describes which factors play a major role in layout decisions, and he provides guidelines on how they can be taken into account [Neil 82]. These factors depend on the construction type and scale, the work organization (the contract structure and company practice of the parties that are involved with the project design, construction, and operation), and the project location (the specified site boundaries, and geological and climatic conditions). Among these factors, several determine the *need* for certain facilities and the *size* of these facilities. Such sizing requirements may be satisfied before any layout is generated. Other requirements affect the *location* of facilities on site and must be taken into account during the layout process. In general, the sizing and the location problem interact with one another: depending upon space availability on site,a facility can vary in size or in number of constituent parts, while meeting its functional requirements.

Managers who lay out construction sites may say that the objectives of their task are to minimize worker and equipment travel time, to keep material handling time low as compared to production time, to meet safety or security objectives, and so on. The multiple objectives, however, do not provide a novice with instructions on how to go about laying out a site; they merely provide a set of specifications or constraints to meet in designing a site layout.. In fact, such a process description is seldom articulated by managers.

Moreover, the role and the level of involvement of the person laying out the site affects the resulting layout. A manager may add his or her personal preferences to the constraints that need to be met while laying out a site. Handa confirmed this by observing differences in the quality of site layouts based on whether the responsible party was the construction manager, the main contractor, or the developer of the project [Handa 87]. In summary, the site layout problem is generally a multi-objective design problem.

The payoff for laying out sites better is potentially very high, but, because site layout is such a tightly intertwined part of the construction process, it is hard to attribute project savings or avoided costs directly to it. Benefits that might be gained by laying out a site well are: ease of inventory control (e.g., no materials are lost in piles of dirt, theft is limited), reduced travel times, reduced noise and dust, avoidance of obstructions and interferences, increased safety and security, prevention of theft, and improved site access. Because we were interested in learning how field managers address this problem, we based the knowledge that is represented in SightPlan on data from actual field practice. SightPlan models how managers construct layout arrangements; however, we did not learn how they evaluate layouts.

2.3. Construction Practice for Site Layout

The current practice is that a project engineer or a superintendent is assigned to plan the layout of temporary facilities on a project at the beginning of construction, because such a person can take full responsibility for implementation and execution of the layout plan. Site layout is mostly done as a pre-planning task, following substantial completion of design drawings, often after civil works (such as clearing and grading of the site, or excavating and installing foundations for the project) have already commenced, but at a time preceding the arrival on site of major contractors requiring substantial laydown areas. The assigned field manager has access to specifications that describe the scope of the project, a milestone schedule, site arrangement drawings showing the permanent facilities, contract documents, and other information about the construction processes, methods, and sequences of the components to be built. All of these pieces of information help in generating the site layout.

Because layout is closely tied to other site management concerns, such as planning and scheduling, and because of human cognitive limitations (see for instance [Miller 56]), humans cannot possibly keep track of all the factors that could affect the selection, dimensioning, and location of temporary facilities. Therefore, several modeling tools and methods have been developed by managers and researchers to help them conceptualize and solve the layout problem.

2.4. Models and Methods to Assist with Site Layout

Models and methods have in common that they force their users to select a limited number of factors or variables for representation of the problem they represent or solve. Models for site layout often consist of drawings or physical cut-outs created from materials such as paper, cardboard, or plastic. They are embodied by, e.g., icons, two-dimensional templates, or three-dimensional scale models [Henderson 76]. Models allow people to inspect a layout visually while it is being developed. The process of manipulating the individual parts of the model in order to achieve a satisfactory layout, however, is left to a person.

Methods have also been developed to automate the layout process, i.e., they provide a mechanism to achieve a layout when given a specified input. One may distinguish mechanisms that attempt to replicate the decision-making processes of human experts (employing artificial intelligence (AI) techniques) from more traditional satisfaction or optimization approaches (employing operations research (OR) techniques) which do not necessarily mimic human decision-making.

There are two main types of heuristic methods for layout: "construction" and "improvement." Construction methods incrementally build a layout by adding pieces to be laid out one at a time. Improvement methods start with a complete initial layout and swap pieces in order to obtain a layout that better meets the requirements.

One may further distinguish layout from location methods. Whereas layout takes into account the dimensions of the pieces to be laid out, location ignores the dimensions of individual objects because the dimensions of the pieces are small compared to the distances between them. Location methods might be used, e.g., for selecting the locations of warehouses for a grocery chain; SightPlan uses a layout method. The interested reader can obtain a more detailed description of layout and locations methods in [Francis 74] and can find a survey of layout systems currently on the market in [Driscoll 86].

Despite several attempts to formalize the site layout process, field construction managers have resisted using operations research-type methods to solve their layout problems, whereas cut-out templates and other models do get used to help them in their task. Although facility location problems have found some acceptance in construction ([Reinschmidt 75], [Gates 78], [Mayer 81], [Rodriguez-Ramos 82, 83], [Stark 83]), neither layout nor location methods are widely used in practice. Our rationale for this is that managers find the formulation of the input for such methods tedious and difficult to achieve with the limited amounts of information at hand, and that they mistrust the solutions generated by the "black box" that constitutes the implementation. Indeed, this "black box" does not provide any means for them to get insight into the process, or to intervene and follow intuitions that might lead to an acceptable solution. Because on the one hand, it is expected from managers that they virtually blindly accept such a program's solution and implement it on the site, while on the other hand, they remain responsible for the program's results, this use of "black box" computer programs understandably leads to resentment.

In this respect we believed that programming techniques from AI could contribute to the development of better tools. First, expert systems represent human problem solving declaratively and thus their reasoning process should be easier to understand and follow by a human observer. Second, a user can be permitted to interrupt the system and introduce her or his own actions instead, to which the system can subsequently react. Our SightPlan system features both of these behaviors.

A proof of concept of a construction method for layout, implemented with an AI approach, was given by Eastman as early as in 1972 [Eastman 72]. Applications of this method to construction site layout were given by [Tommelein 87a, 87b] and [Hamiani 87, 88]. The research described in this paper completes and extends the prototype described by Tommelein.

3. SightPlan Research Approach

We review the goals, scope and research approach of the SightPlan project in this chapter.

3.1. Research Goals

Our work on SightPlan had four major research goals:

1 Understand the knowledge used by construction managers for laying out construction sites, in order to formalize this area of field construction practice.

2 Build a model to reproduce the problem-solving strategies employed by these human experts in designing site layouts.

3 Test the power and generality of BB1, a generic blackboard architecture, on a realistic design problem in a new and challenging domain.

4 Explore the potential of computer-based strategy (as opposed to a human-based strategy) and interactive graphics to integrate the bookkeeping and computational strengths of a computer system like SightPlan with the spatial perception and reasoning strengths of a person engaged in this spatial layout task.

3.2. Research Approach

We delimited the scope of SightPlan in ways which make the problem it addresses tractable, yet keep it realistic. This would enable us to make significant progress, while still proving our concepts adequately. Next we explain the rationale for selecting the BB1 blackboard expert system architecture as a foundation for SightPlan. We conclude this section with a brief chronology of the steps in development of a series of SightPlan models, each embodying a different problem-solving strategy.

3.2.1. Defining and Delimiting Problem Scope

Our assumptions on SightPlan's scope and our assessment of their impact on the realism of the system are the following.

- **Develop SightPlan for a single class of projects:** First, we selected a class of projects for which site layout was an important problem. After consulting several industry experts we decided that remotely-located fossil power plants would be our target application. Several such projects were under construction and site managers appeared to be available to us as potential sources of site layout knowledge. We would choose one such project and study its layout in detail. In addition to this, we had found empirical guidelines [Dressel 63; Popescu 78, 80, 81, 86; Neil 80, 82; Handa 87, 88, 89], field construction manuals [Bechtel 88, Stone&Webster 78, 79], and a few articles [Tatum 81; Weidemier 86] describing concerns for site layout on this kind of projects, and could incorporate parts of these in our knowledge base as well.

- **Arrange about 50 predefined temporary facilities:** Second, we would locate approximately 50 temporary facilities on the site. This is about the level of detail at which designers and construction managers currently lay out temporary buildings and long-term laydown areas on such sites. Our system would not reason about what facilities were needed, but we would illustrate how the sizing and shaping of facilities is tightly intertwined with layout design. We believed that our hardware and software tools would be capable of tackling this dimension of the layout problem at full scale, so we would get a good handle on the computing needs for solving a realistic layout problem.

- **Ignore topography:** Third, we decided to limit our layout to two dimensions, ignoring topography on the site. This simplification would prevent us from satisfying some constraints that explicitly relate to the third dimension of objects. We found that there were not many such constraints for the cases that we would model, and assumed that the gain in computation time obtained by this simplification would justify it.

- **Assume rectangular objects and rectilinear constraints:** Fourth, we required that objects be rectangular in shape to keep computation costs low. This conceptual limitation is easily overcome since we can create arbitrarily complex shapes by defining groupings of adjacent rectangles. In similar vein, we limited all constraints to be rectilinear, i.e., minimum and maximum distances between objects are measured horizontally or vertically. This is a simplification of distance constraints where true radii might be more accurate but, as will be described later, the computation of the geometry engine developed to manipulate constraints in SightPlan was greatly simplified thereby. The savings in computation would more than make up for any loss of realism in this proof-of-concept application.

- **Consider time implicitly at first, and then in discrete intervals:** Finally, we started out by considering time only implicitly. That is, when locating an object needed only for part of the project, we would not reason about re-use of its assigned area at other times. After we completed the first SightPlan model, we extended it to reason about time explicitly. That is, areas no longer needed by a facility can be reused at other times during construction. Our temporal model is a simple one in that we did not address all the issues involved with planning. This is one of the areas which we identify as an opportunity for future extensions of the system.

We contend that the limitations imposed on SightPlan are all quite acceptable. They leave us with a realistic representation of site layout that goes far beyond toy-problem dimensions, and approaches the grain size that would be needed in a practical decision-support tool for construction site layout.

3.2.2. Reasons for Selecting the BB1 Blackboard Architecture

We had several reasons for selecting a blackboard architecture in which to implement SightPlan:

- **Opportunistic Reasoning Approach:** Spatial arrangement of objects under constraints form a class of problems which humans find very challenging. Since humans cannot hold more than a small number of constraints in short-term memory at any one time, they must generally solve such problems by assembling solutions incrementally, examining partial solutions for acceptance, and changing strategy along the way. The BB1 control architecture was specifically designed by Dr. Hayes-Roth to represent and reason opportunistically with the kind of meta-knowledge (strategic knowledge about how to order steps in problem solving) that humans use to guide such planning and design tasks [Hayes-Roth 85].

- **Hybrid Architecture:** BB1 uses a frame-based representation. Much of SightPlan's knowledge falls naturally into this representation in which attributes and links can be inherited from one or more related frames. BB1 also provides excellent facilities for browsing and editing knowledge bases. Furthermore, earlier application developments in BB1 established the idea of layering the various knowledge bases in the system hierarchically, so other applications—including SightPlan—could easily reuse the appropriate frame hierarchies [Hayes-Roth 87].

  BB1 molds rule-like structures into frames, but BB1's inference mechanism does not employ conventional forward or backward chaining of these. Instead, these frames represent independent units, called knowledge sources (KSs). Knowledge sources are triggered by information changes on the blackboards, which is maintained and updated at every cycle of problem solving. This provides opportunistic reasoning capabilities and modularly distributed knowledge, both of which fit SightPlan's needs.

- **Explicit Representation of Layout Strategies:** BB1 separates domain KSs (e.g., in SightPlan those that describe that a specific object can be positioned in a specific arrangement with a specific constraint) from control KSs (e.g., in SightPlan the set of KSs that capture the strategy pursued by a field manager laying out a site), so this would allow us to explicitely formalize a layout strategy.

An important ingredient of our research approach was to decompose the problem so that we could avoid manipulating massive constraint sets — the Achilles heel of several prior attempts to employ constraint satisfaction in solving complex design problems. We therefore wanted to define compound objects (consisting of groups of facilities highly constrained with respect to each other), arrange them internally, locate them as a single object on the site, and then consider remaining detailed constraints to objects on the site from within the compound object. The BB1 control architecture facilitates modeling multi-level control strategies for this kind of problem solving, and is able to manipulate multiple and hierarchical partial arrangements to store multiple descriptions of the problem's solution.

- **ACCORD Language for Constructive Assembly:** A previous application of BB1 solving a spatial arrangement problem was PROTEAN [Brinkley 86; Buchanan 86; Hayes-Roth 86]. PROTEAN "designs" hypothetical three-dimensional protein structures to satisfy the constraints on its constituents. Resulting from PROTEAN a set of primitive operators for assembling compound objects in space under constraints were formalized into the ACCORD language framework. ACCORD provides KSs with an English-like language to express their actions related to the task of arrangement assembly. This facilitates control reasoning by providing an unambiguous language for describing design strategies and supports explanation by providing a human-interpretable reasoning language. We thought that ACCORD and other parts of PROTEAN could be adapted to SightPlan. Our exploratory attempts to test this hypothesis were quite successful, and permitted us to propose the use of BB1/ACCORD for our problem.
- **Application of Multiple Kinds of Expertise in Parallel:** Site layout involves different kinds of expertise possessed by the various participants involved with the construction project. Expertise about personnel movement, equipment access, materials management, safety, and environmental concerns are just some of the important kinds of knowledge used to lay out a large construction site well. Blackboard architectures represent a proven computational paradigm for bringing multiple kinds of knowledge to bear on a complex problem with strategic mediation and control.
- **Availability of BB1:** When the civil engineers in our group launched the idea of modeling site layout in a discussion with the computer scientists, it was warmly welcomed as an opportunity for further validation and generalization of their architecture. They felt that the direct support of BB1's developers would be an invaluable resource to the project. This hunch proved correct.

All the above reasons explain why the BB1 software provided a good fit with the organizational and technical requirements of SightPlan's problem.

3.2.3. Steps in Implementing, Validating, and Testing SightPlan

The SightPlan modeling started with the implementation of a simple prototype layout system, after which we selected a case for more detailed study. The SightPlan Expert Model is an implementation of the "early commitment" way in which we believe humans address the site layout problem. We contrasted this model with the SightPlan Computational Model that follows a "postponed commitment" approach that we argue is possible with computer tools of the kind being developed here. Experiments using SightPlan to shed light on the strengths and weaknesses of human vs. machine problem solving strategies for space layout became a significant thrust of the artificial intelligence research on this project.

After validating the Expert Model with knowledge about a second site, we augmented the expert model to reason explicitly about time. Finally, in parallel with the development of these models, we developed an interactive graphical user interface so that a human expert can collaborate with SightPlan to solve spatial layout problems. The joint cognitive system can thus exploit the strengths of each.

4. The SightPlan System Architecture

The blackboard architecture is widely recognized as an extremely general and flexible framework for problem solving [Engelmore 88]. It has been applied successfully to a range of problems, including planning [Hayes-Roth 79], speech understanding [Erman 80], and sonar interpretation [Nii 82].

The CS members of our research group developed a domain-independent implementation of the blackboard architecture, called BB1 [Hayes-Roth85], to support a variety of domain-specific applications. BB1 allows diverse knowledge sources, functioning independently, to solve problems cooperatively by recording and exchanging solution elements on a global data structure, the blackboard. BB1 further allows control knowledge sources dynamically to post and modify the strategy that the system uses to choose knowledge sources for execution on each problem-solving cycle.

4.1. The BlackBoard Metaphor

The BB1 architecture draws on the blackboard metaphor. Imagine a meeting situation in which a number of participants—here called knowledge sources (KSs)—are faced with a problem that is described on the blackboard (BB). None of the KSs can solve the entire problem on its own, but each of them may be able to contribute problem-solving steps that, when combined in a reasonable sequence, lead to a solution. By looking at the BB, KSs know when it is appropriate for them to focus their attention, and they know when it is proper to propose to take action. The only way for KSs to communicate with each other is by making changes on the blackboard. In each step towards the solution, one—and only one—KS gets to execute its proposed action, that is, is allowed to make changes to the BB. In reaction to such changes on the BB, other KSs may now focus their attention or propose to take action. It is the moderator in the meeting—here called the scheduler—who, at each cycle, listens to the contributions that KSs propose and who selects the best KS, which then gets to execute its action. Thus the inference mechanism of BB1, as it is embodied by the scheduler, is both incremental and opportunistic.

4.2. BB1's Layered Environment

Concepts in the BB1/SightPlan world are represented in a conceptual graph [Sowa 84]; the underlying BB1 scheduler then makes the appropriate inferences. Concepts are represented by frames that can have any kind of user-defined attributes or links to other objects, and that can inherit attributes over specific links.

For the sake of clarity and flexibility the conceptual graph is layered, that is, concepts specific to a particular application domain are grouped in what is termed a blackboard (BB), itself part of a knowledge base (KB). A schema representing various layers and how they relate conceptually to each other is given in Figure 4-1. This figure illustrates how multiple application systems can co-exist in the same conceptual graph representation. PROTEAN is an application in the domain of biochemistry. Intermountain and American 1 are two separate SightPlan implementations in the domain of construction management. By dividing the system up in layers it is easy to build a new application by substituting the needed BBs only.

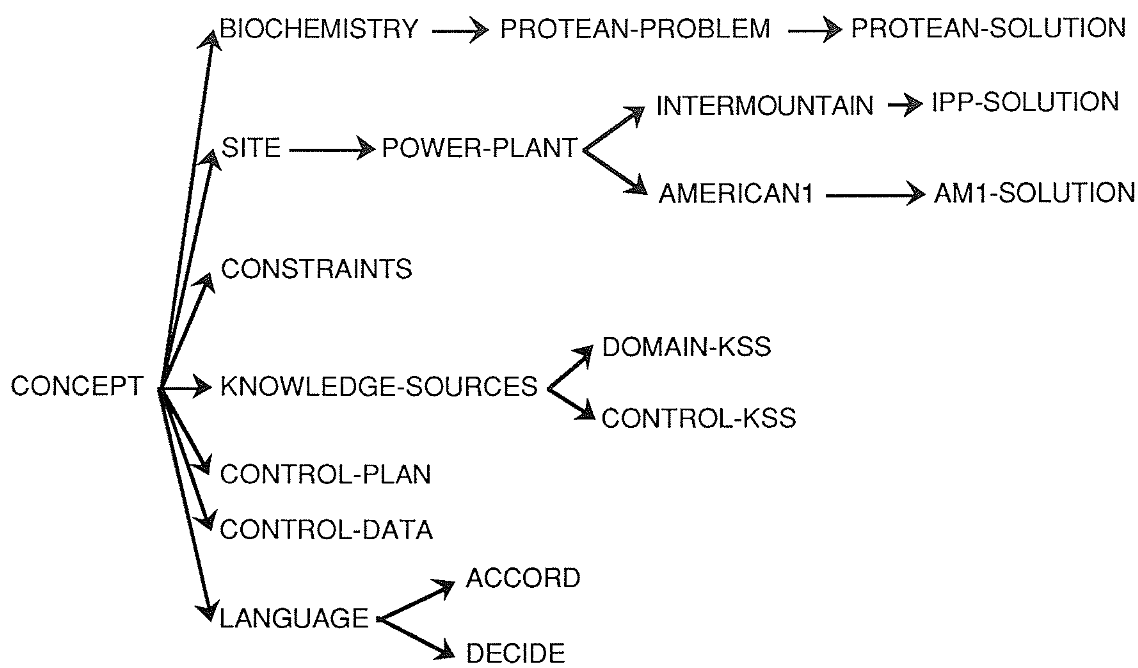


Figure 4-1: Schema of Layered BBs in SightPlan's Conceptual Graph. Each of the nodes in this graph represents a Blackboard that groups frames representing concepts. The nodes in the topmost line represent PROTEAN BBs that are not used by the SightPlan system.

The most abstract concepts, from which domain-specific concepts stem, are grouped on the CONCEPT-BB. Those that define objects, for instance those related to site layout and construction management, are grouped on the SITE-BB, and objects on the SITE-BB, in turn, can be specialized to concepts related to power plant construction on the POWER-PLANT-BB. Finally, those examples of specific objects that exist on one particular site—here for instance the site of the Intermountain Power Project—are on the INTERMOUNTAIN-BB, those that exist on another site—for instance the American 1 site—are on the AMERICAN1-BB. Solutions to the problem the system is solving are generated in terms of instances of example objects, and these instances are on the respective solution BBs (PROTEAN-SOLUTION, IPP-SOLUTION, AMERICAN1-SOLUTION). In this way, the concepts on the CONCEPT-BB can be specialized to accommodate any representation of more specialized worlds. This specialization hierarchy makes use of CAN-BE-A, EXEMPLIFIED-BY, and INSTANTIATED-BY links and their inverse links for inheritance of attributes and links.

As an example of how different pieces of knowledge about a problem domain may be implemented in BB1 the following paragraphs will elaborate on what is represented on some of the SightPlan blackboards.

4.2.1. Site, Power-Plant, and Problem BlackBoards

The SITE-BB contains representations of concepts that are generally talked about on construction sites, such as trailers, laydown areas, and roads. These concepts are represented by nouns in spoken language. The SITE-BB also has representations of adjectives that field practitioners commonly use, such as: a "large" laydown area as opposed to a "small" laydown area. Concepts are classified by level to make the BB more comprehensive (Figure 4-2). So, the PHYSICAL-OBJECTS level contains the representation of the physical objects laydown-area, road, and trailer, whereas the MODIFIERS level groups the adjectives.

PHYSICAL-OBJECTS

BUILDING	ROAD	RAILROAD
LAYDOWN	SITE-PHYSICAL-OBJECT	TRAILER
WAREHOUSE	PARKING	

MODIFIERS

LARGE	SMALL	LONG-TERM
IMPORTANT	EFFICIENT	PERMANENT

CONTEXTS

SITE	SUB-AREA

CONSTRAINTS

CLOSER-THAN	FURTHER-THAN	ADJACENT-TO
AS-CLOSE-AS-POSSIBLE	SITE-DISTANCE-CONSTRAINT	

Figure 4-2: Excerpt of the SITE-BB with Objects Ordered by Level

Modifiers have the attribute "function-definition" which contains a function definition. When applied to the object that the modifier modifies (i.e., the object linked to the modifier with link "modifies") the function returns a numeric value.

Simple and **aggregate objects** are defined on SightPlan's PROBLEM-BB. Aggregate objects distinguish themselves from simple objects in that they *include* other objects. These included objects, in turn, can be simple or aggregate objects.

It is possible that some objects included in the problem already have a location in a given context before problem-solving starts. This is for instance the case with the object POWER-UNIT-1 in Figure 4-3. The object has the attribute **location**, whose value is the name of the site at which it is located (here, the site is in Delta, Utah, and is therefore called delta-site), and the coordinates of the upper left corner of the rectangle representing it on that site.

POWER-UNIT-1
level:
INTERMOUNTAIN.POWER-PLANT-FACILITIES
attributes:
dimensions: (1040 480)
location: (delta-site 845 1067)
links:
exemplifies: POWER-PLANT.POWER-PLANT-FACILITIES.POWER-UNIT

LARGE
level: SITE.MODIFIERS
attributes:
function-definition: (lambda (...) ...)
links:
is-a: CONCEPT.NATURAL-TYPE.MODIFIER
modifies: SITE.PHYSICAL-OBJECTS.SITE-PHYSICAL-OBJECT

POWER-UNIT
level:
POWER-PLANT.POWER-PLANT-FACILITIES
attributes:
links:
is-a: SITE.PHYSICAL-OBJECTS.BUILDING
exemplified-by: INTERMOUNTAIN.POWER-PLANT-FACILITIES.POWER-UNIT-1

BUILDING
level: SITE-BB.PHYSICAL-OBJECTS
attributes:
links:
is-a: SITE.PHYSICAL-OBJECTS.SITE-PHYSICAL-OBJECT
can-be-a: POWER-PLANT.POWER-PLANT-FACILITIES.POWER-UNIT

Figure 4-3: Frames Representing a Physical-Object and a Modifier in the SightPlan Application

In SightPlan's current implementation, all simple objects are represented by single rectangles. POWER-UNIT-1 in Figure 4-3 is a simple object: its attribute **dimensions** give the dimensions of the rectangular area it occupies.

While simple objects have a predefined area and shape, aggregate objects may be undimensioned and SightPlan then reasons about their size and shape before positioning them in an arrangement. An example is the CONSTRUCTION-ENTRANCE aggregate, including the craft parking lot, two security offices, two brass alleys, a welder qualification building, and a weld test building. An aggregate object includes several parts that have constraints with each other. Depending on the type of constraint, SightPlan may decide to meet them by building a separate arrangement, including the parts, and solving this layout independently of possible constraints between the parts and other objects not in the aggregate or of possible constraints between the aggregate and other objects. This results in a layout of which the boundaries provide the size and shape of a single rectangle for the aggregate. Alternatively, SightPlan may heuristically estimate the dimensions and the shape of that rectangle.

4.2.2. Constraints BlackBoard

The CONSTRAINTS-BB contains representations of concepts used as spatial prepositions or adverbs in common English. Since these concepts express unary, binary, or n-ary relations between objects they are represented by means of frames that name constraint types (Figure 4-4). These types are exemplified by the actual example constraints that involve the related example objects.

CLOSER-THAN	CLOSER-THAN-45-1
level: SITE.CONSTRAINTS **attributes:** function-definition: (function (lambda (...) ...)) description: "A first object is closer than a given distance to another object if the distance between their nearest edges is smaller than or equal to that distance. The constraint is symmetrical in its arguments." **links:** is-a: SITE.CONSTRAINTS.SITE-DISTANCE-CONSTRAINT exemplified-by: CONSTRAINTS.CLOSER-THAN.CLOSER-THAN-1	**level:** CONSTRAINTS.CLOSER-THAN **attributes:** arg1: INTERMOUNTAIN.POWER-PLANT-FACILITIES.POWER-UNIT-1 arg2: INTERMOUNTAIN.CONTRACTOR-LAYDOWN-AREAS.TURBINE-GENERATOR **links:** exemplifies: SITE.CONSTRAINTS.CLOSER-THAN involves: (INTERMOUNTAIN.POWER-PLANT-FACILITIES.POWER-UNIT-1 INTERMOUNTAIN.CONTRACTOR-LAYDOWN-AREAS.TURBINE-GENERATOR)

Figure 4-4: Frames Representing a Constraint Type and a Constraint Example

The CONSTRAINTS level contains constraint types such as closer-than, further-than, and each constraint type in turn labels its separate level that groups the constraint's examples This is shown in Figure 4-5.

Whereas SightPlan performs the task of reasoning about the selection of objects to position and about the constraints that need to be satisfied, the system relies upon a separate constraint engine to process the constraint. That is, SightPlan passes the initial sets of possible locations of each of the involved objects and the description of the constraint to that engine, and receives new sets of legal locations for the involved objects to be applied after the constraint engine has reduced them to satisfy the constraints.

In the definition of each constraint-type, a function is defined as the value of an attribute, and that function calls one or more functions in the constraint engine. When an example-constraint is to be met, it inherits the function from its constraint-type, and that function is called with the example's arguments.

Since the scope of SightPlan is restricted to spatial layout, the system deals with geometrical, topological, and sampling constraints. For example, a geometrical constraint may express that two objects need to be closer than 100 yard from each other; a topological constraint may require that an object be zoned within a certain area; a sampling constraint may express that if one is to pick a point position from a set of positions this can be done by picking the first corner of the first rectangle in the essential area. SightPlan reasons about which constraints to apply and when to apply them. It calls upon its underlying constraint engine to compute constraint satisfaction. Figure 4-6 graphically represents some of the constraint types implemented in SightPlan.

CLOSER-THAN

CLOSER-THAN-45-1	CLOSER-THAN-105-106	

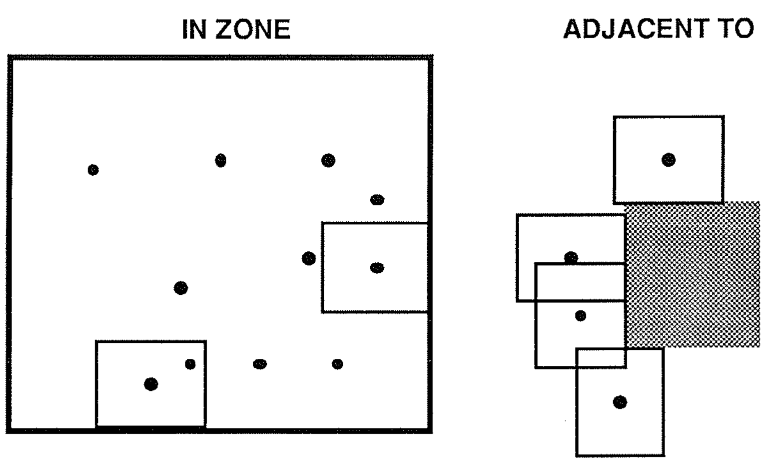
IN-ZONE

IN-ZONE-45-CONSTR IN-ZONE-103-COAL	IN-ZONE-104-CONSTR	IN-ZONE-101-CONSTR

AS-CLOSE-AS-POSSIBLE

ACAP-43-2 ACAP-32-4	ACAP-33-1	ACAP-103-1

Figure 4-5: Excerpt from the CONSTRAINT-BB Displayed by Levels



LEGEND

- represents a possible location for the centerpoint of a rectangle
  The thin boxes show the rectangle at some of its possible locations.

Figure 4-6: Graphical Representation of SightPlan Constraint Types

Several assumptions were made to limit the computational effort required of SightPlan's constraint engine. Two key assumptions are:

1) Only single rectangular entities are represented, and
2) Entities can be positioned only in two-dimensional orthogonal and continuous space (that is, they can have only 0 and 90 degree orientation, hence 2.5 degrees of freedom)

The set of possible locations of an entity is represented by the entity's so-called **essential area**. Figure 4-7 shows our convention for the representation of an essential area.

```
((length  width)                                  —> the dimensions of the entity
((0) (    ((xmin-1  xmax-1)(ymin-1  ymax-1))      —> the possible rectangles
  ((xmin-2  xmax-2)(ymin-2  ymax-2))                 within which the centerpoint
     ...)  )                   of the entity must lie when it is at 0-degree orientation
((90)(    ((xmin-n  xmax-n)(ymin-n  ymax-n))      —> similar, but at 90 degree
     ((xmin-m  xmax-m)(ymin-m  ymax-m))                                orientation
          ...)    ) )
```

Figure 4-7: Convention for the Representation of an Essential Area

Figure 4-8 is an example of how a set of possible locations for a rectangle is represented using this convention.

4.2.3. Knowledge Sources BlackBoards

Knowledge sources (KSs) in BB1 are similar in structure to if-then rules, so the Knowledge Sources Blackboard contains the knowledge that BB1 will apply to make its inferences from the current state of the BBs. These KSs however are not designed to "chain" together as is traditionally the case in rule-based systems; rather, they are independent entities whose if-part can become true based on facts stated on any of the BBs, and whose then-part—upon execution—posts new facts on any of the BBs. Thus, KSs need not be "aware" of each other's presence. KSs, as all other concepts in BB1's conceptual graph, are represented by means of frames.

The if-part of a KS in BB1 distinguishes three different conditions: triggerconditions, preconditions, and obviationconditions (Fig4-9 Triggerconditions state when the KS becomes applicable. In addition, since the conditions of KSs are stated in terms of concept-types, the KS makes use of context variables to specify which of the examples in the specific domain of application apply. When more than one example of the concept-type is applicable, multiple knowledge source activation records (KSARs) are generated, one for each example. Figure 4-10 shows a set of KSARs generated from a single knowledge source's action sentence. Preconditions state when the KS is executable, and obviationconditions state when the KSAR is no longer applicable.

```
(   (60 30)
    ((0)    (((30 270)(15 85))))
    ((90)   (((15 285)(30 70))))   )
```

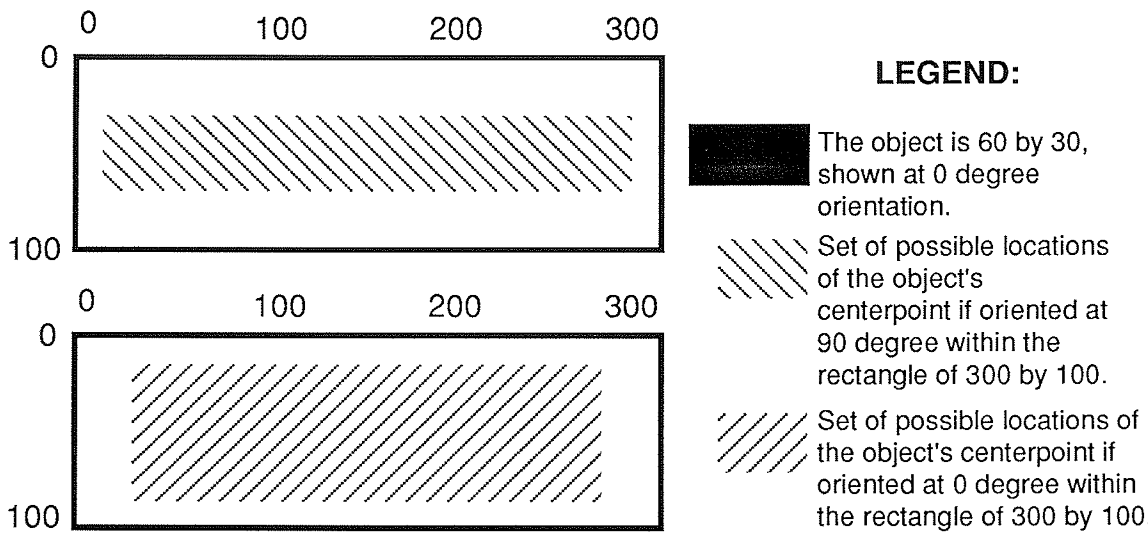


Figure 4-8: The Essential Area of a Rectangle and its Graphical Display The rectangle has dimensions 60 by 30 and is positioned in an area of dimensions 300 by 100.

The then-part of a KS tells BB1 what changes to make to BBs when the KS in question is executed. Although at any time multiple KSARs may compete for execution, the BB1 scheduler embodies an incremental problem solver, so it picks only one at a time to execute. An action's execution causes the events of adding or modifying objects on BBs. Because an object was changed a new state is promoted (Figure 4-11). By definition, triggerconditions depend on events, preconditions depend on states, and obviationconditions (usually) depend on states.

KSs in BB1 contain additional information, most of which is for bookkeeping purposes and is currently not used for inference. A full description of KSs and their operation is provided in the BB1 Manual [Garvey 87; Hewett 88a, 88b, 88c].

CONVENTIONAL RULE	KNOWLEDGE SOURCE	Explanation
IF	TRIGGERCONDITIONS	say when a KS becomes applicable or relevant to the current state of problem-solving
	CONTEXTVARS	allow the system to instantiate types in the KS' triggerconditions
AND	PRECONDITIONS	say when a KSAR becomes executable
AND NOT	OBVIATIONCONDITIONS	say when a KSAR should no longer be considered
THEN	ACTIONS	

Figure 4-9: Structure of a Knowledge Source in BB1

KS's action sentence:

ANCHOR ANCHOREE TO ANCHOR IN PARTIAL-ARRANGEMENT WITH CONSTRAINT

Two KSARs' instantiated action sentences:

ANCHOR TURBINE-GENERATOR TO POWER-UNIT-1-1 IN PA1 WITH CLOSER-THAN-110-1

ANCHOR MECH&PIPING-UNIT1-1 TO POWER-UNIT-1-1 IN PA1 WITH CLOSER-THAN-112-1

Figure 4-10: Instantiation of Multiple KSARs from one KS

Figure 4-11: The Relationships among Actions, Events and States.

Two types of knowledge sources are distinguished in BB1: domain knowledge sources and control knowledge sources.

4.2.3.1. Domain Knowledge Sources

Domain Knowledge Sources are application-dependent and are specific to the problem-solving method that is used. In SightPlan, whose goal is to locate objects within some partial arrangement while satisfying constraints, they describe actions such as:

- create a new arrangement,
- start another partial arrangement,
- include certain objects,
- meet certain constraints between objects,
- provide display for the object that was just positioned, and so on.

Figure 4-12 shows an example of a domain KS, and Figure 4-13 gives the type hierarchy of domain KSs on the SIGHTPLAN-KS-BB.

```
                         ANCHOR-OBJECT

level: SIGHTPLAN-KS.DOMAIN
attributes:
triggerconditions: (DID-ORIENT PARTIAL-ARRANGEMENT (THE-PA) ABOUT
    ANCHOR (THE-ANCHOR))
ksarcontexts:
    ((THE-ANCHOREE (REMOVE-IF-NOT
                (FUNCTION (LAMBDA (OBJ) ($PLAYS OBJ 'ANCHOREE))))
                         ($OBJECTS THE-PA 'INCLUDES)))
    (THE-CONSTRAINT (CONSTRAINTS-BETWEEN THE-ANCHOR THE-ANCHOREE)))
preconditions: (T)
obviationconditions: NIL
actions: (ANCHOR THE-ANCHOREE TO THE-ANCHOR IN THE-PA WITH THE-CONSTRAINT)
description: "Apply one constraint at a time to anchor an anchoree to the anchor in the partial
                arrangement"
links:
is-a: SIGHTPLAN-KS.SKILL.SIGHTPLAN-DOMAIN-KS
implements: SIGHTPLAN-KS.SKILL.SIGHTPLAN
```

Figure 4-12: Example Domain Knowledge Source

Domain KSs use the ACCORD language for expressing the action they want to take. ACCORD will be discussed in the following section on Language BlackBoards. A domain KS's action makes changes to the SOLUTION-BB.

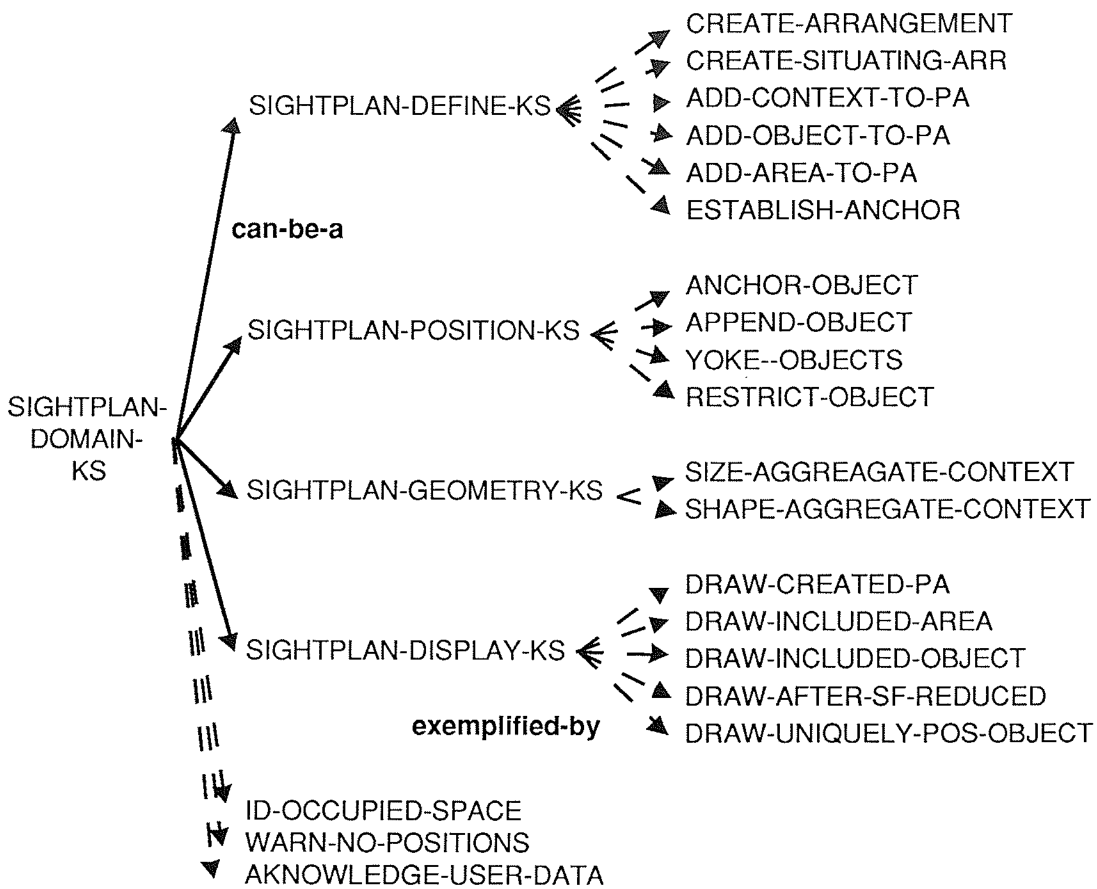


Figure 4-13: Type Hierarchy of Domain Knowledge Sources on the SIGHTPLAN-KS-BB

4.2.3.2 Control Knowledge Sources

Control knowledge sources contain so-called meta-knowledge, which takes the form of BB information that will allow the BB1 scheduler to assign priorities on KSARs. For example, control KSs express strategic information on the desirability of domain actions, as well as—for closure—on the desirability of control actions. Thanks to control knowledge sources, a BB1 application can dynamically alter its strategy and opportunistically select its actions.

Control KSs make changes to the CONTROL-DATA-BB, on which they can post or modify one of three things: a strategy, a focus, or a heuristic. Figure 4-14 displays a control plan part way through the execution of a SightPlan run. Strategies provide high-level statements of what needs to be done to solve the problem. Foci do the same, but they describe the preferred steps in more detail. Foci are used by the scheduler to determine which of the executable KSARs is most desirable at that cycle. Heuristics, which implement foci, are the ones that prescribe what function should be used by the scheduler to compute this desirability. We will return to the question of how the scheduler determines desirability when we describe the rating mechanism in the BB1 control loop.

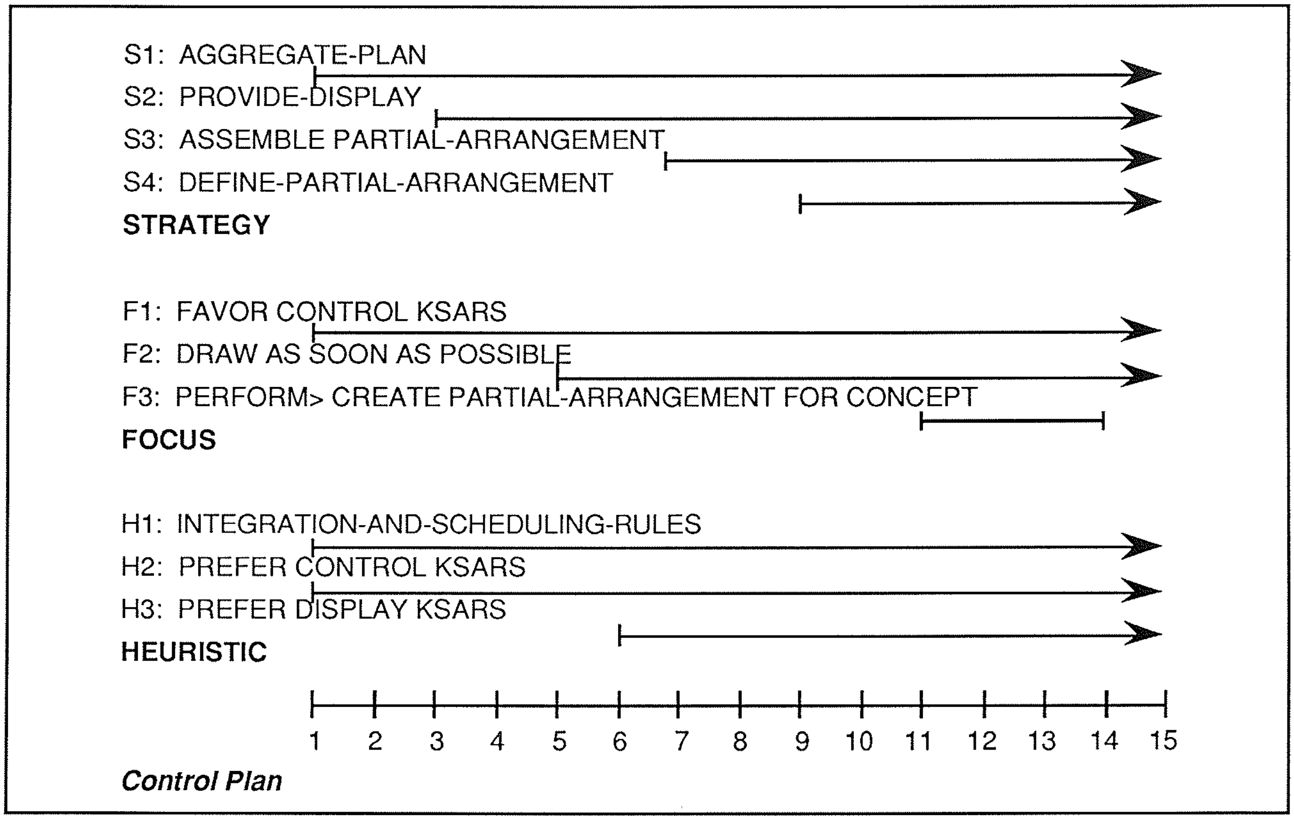


Figure 4-14: Control-Plan BB Displayed by Cycle

Control KSs (see Figure 4-15) make use of their own language, called DECIDE, for expressing the types of actions to take. This language is discussed briefly in the following section on Language BBs.

```
                          POSITION-ALL-OBJECTS

level: SIGHTPLAN-KS.CONTROL
attributes:
triggerconditions: (($EVENT-LEVEL-IS CONTROL-PLAN.STRATEGY)
                   ($EVENT-TYPE-IS MODIFY)
                   ($CHANGED-ATTRIBUTE-IS CURRENT-PRESCRIPTION)
                   (MEMBER 'POSITION-ALL-OBJECTS
                        ($VALUE $TRIGGER-OBJECT 'CURRENT-PRESCRIPTION)))
ksarcontexts: NIL
preconditions: (($SET THE-PA ($SHORT-NAME
                        ($NEWEST-OBJECT 'SOLUTION.PARTIAL-ARRANGEMENT))))
obviationconditions: NIL
actions: ((FOCUS-ON (PERFORM> POSITION LARGE TIME-CRITICAL OBJECT IN
                   ($NEWEST-OBJECT 'SOLUTION.PARTIAL-ARRANGEMENT) WITH
                   IMPORTANT CONSTRAINT)))
description: "Position those object first that must meet important constraints."
links:
is-a: SIGHTPLAN-KS.SKILL.SIGHTPLAN-CONTROL-KS
implements: SIGHTPLAN-KS.SKILL.SIGHTPLAN
```

Figure 4-15: Example of a Control Knowledge Source

All KSs in BB1 are treated in the same manner: as independent agents that generate proposed actions (KSARs) which compete for execution. Control KSs have the same format as Domain KSs (compare Figure 414 and Figure 415) so they first trigger, then generate one or more KSARs, whose actions can be executed when it becomes executable. Sometimes, though, an application designer may know in advance that a sequence of strategic steps needs to be executed in a fixed order. Thus we could short-cut the triggering mechanism and put these KSs into the format of a skeletal plan (Figure 4-16) A skeletal plan is a tree-hierarchy of concepts that prescribe strategies, foci, and heuristics. These concepts have an attribute "strategic-generator" in which it is prespecified which part of the plan is to be posted next. In the generic layer of the BB1 architecture where generic control KSs are defined, there are KSs that know how to initialize, update, and terminate prescriptions of such skeletal plans.

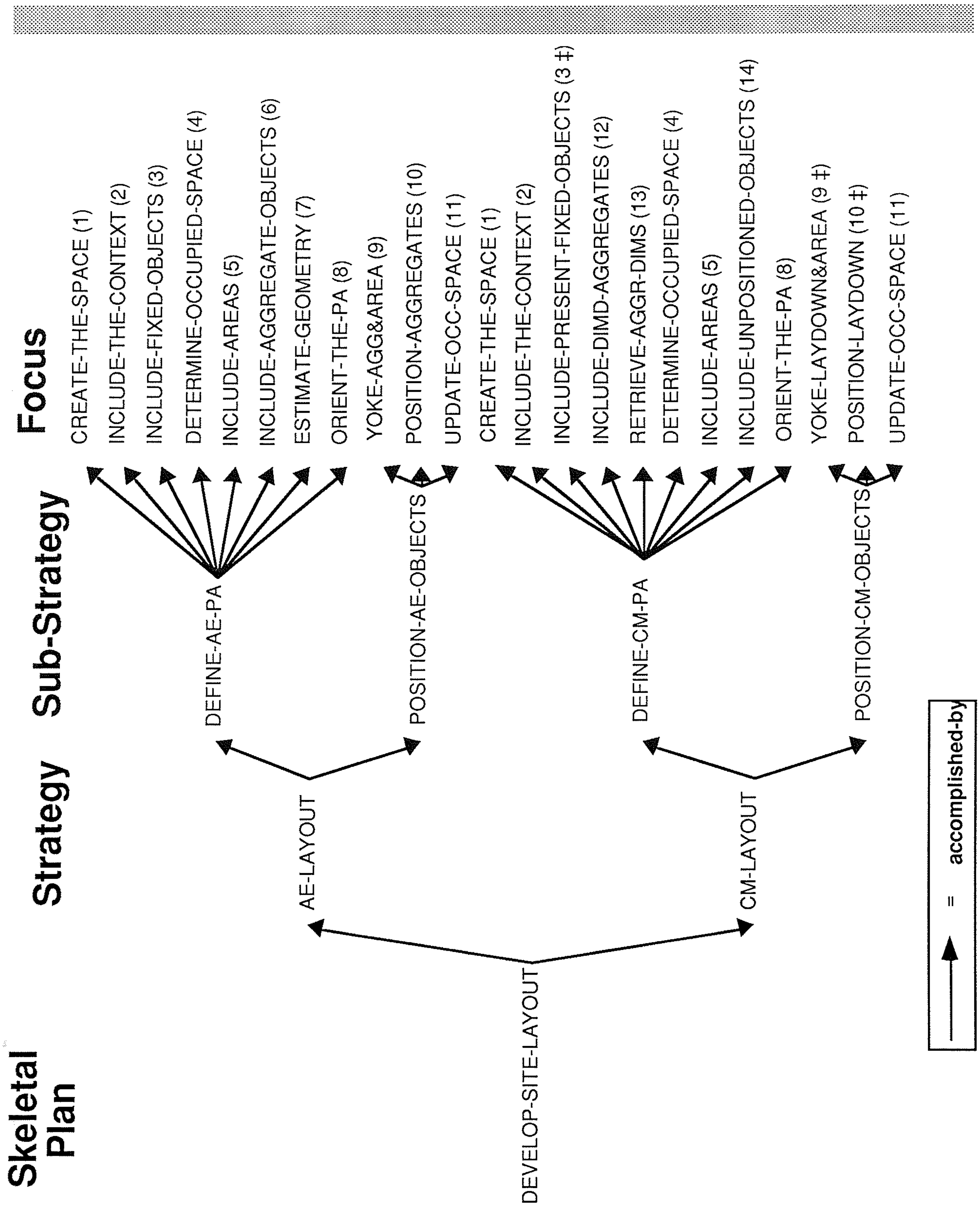


Figure 4-16: Skeletal Plan for Control. Numbers in parentheses refer to the BB1 cycles in which each focus became active.

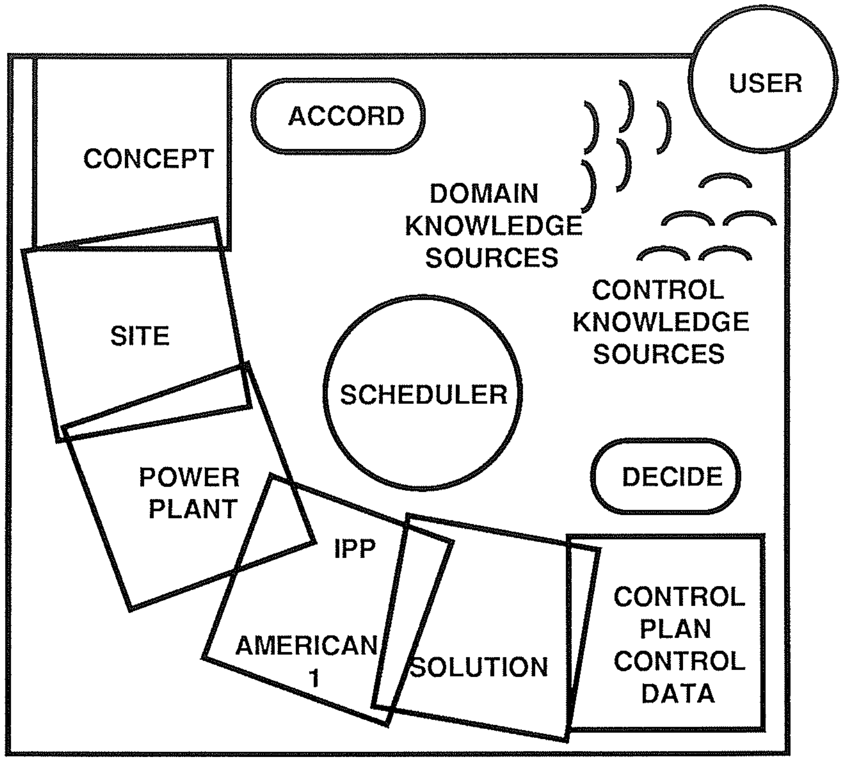


Figure 4-17: Domain and Control Knowledge Sources Joined in BB1 This Figure provides a good overview of the architecture of the SightPlan system. Note the layered blackboards, the domain and control knowledge sources that read from and write to them, the ACCORD and DECIDE frameworks which create high level languages for domain and control actions, and the scheduler which selects an action to execute on each cycle.

4.2.4. Language BlackBoards

Language BBs contain verbs that KSs can use to express their proposed actions. SightPlan's domain KSs make use of the ACCORD language for constructive assembly and its control KSs make use of the DECIDE language to post prescriptions.

4.2.4.1. ACCORD: A Language for Constructive Assembly

ACCORD is the language that permits KSs to express their triggerconditions, preconditions, or obviationconditions in terms of a vocabulary established for application of the constructive assembly method. As mentioned before, the action of a KS consists of adding or modifying object(s) on BB(s). Such actions can be expressed with calls to the native BB1 low-level functions, but these descriptions are rather cumbersome to write or read by people who author or use KSs. Also, because it turned out that the same combination of such calls appeared in several KSs, it was natural to abstract those low-level function calls out to a higher level language [Hayes-Roth 88]. Another advantage of using a language is that it makes the matching between desired action types (foci) and possible actions (executable KSARs) very easy (as we will show in Section

4.3). Besides providing a vocabulary to express actions, the ACCORD language encompasses vocabulary to express caused events and promoted states (Figure 4-18).

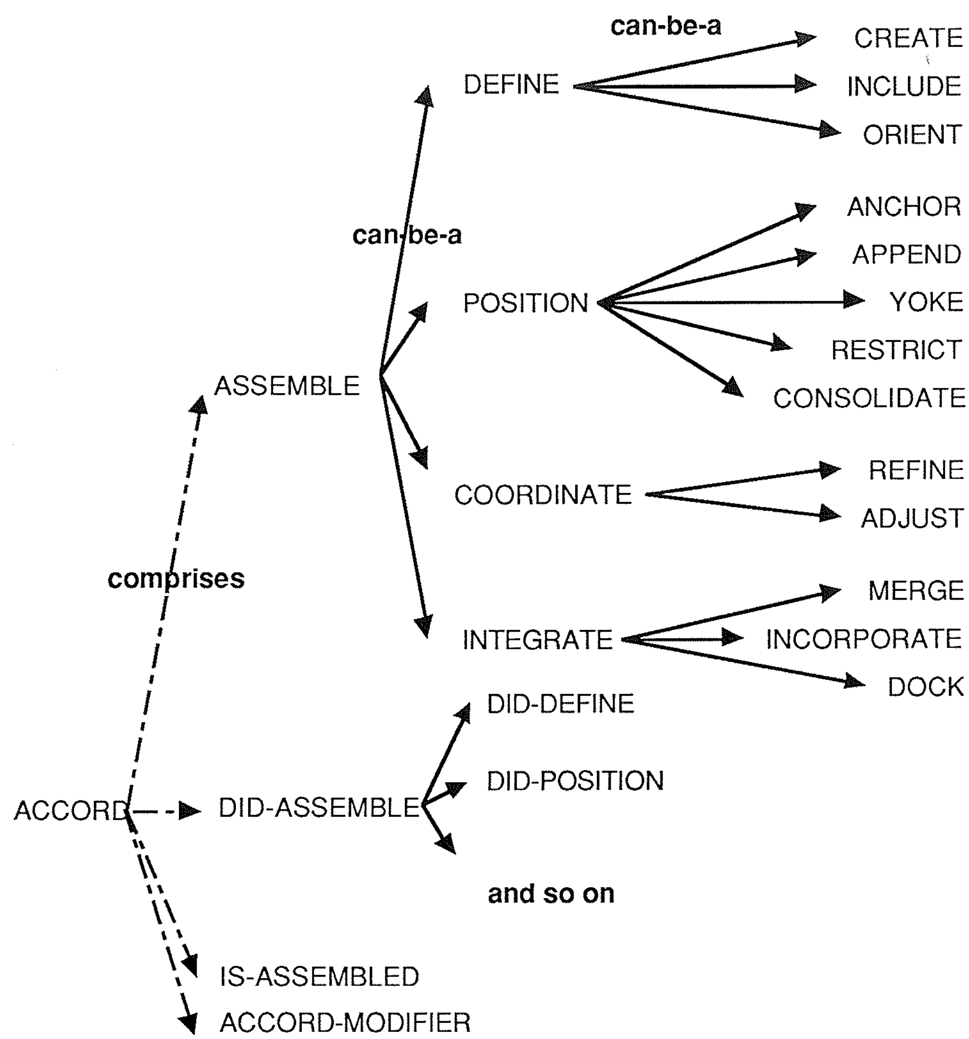


Figure 4-18: Type Hierarchy of ACCORD

Language is used in the following way. On the ACCORD-BB all possible verbs to express Actions, Events, and States are predefined. Actions have an attribute that is the action *template* and another attribute that represents the implementation of that action in terms of the low-level blackboard function calls (attribute *bbactions*). Action templates and their low-level translations share variables, namely those that are marked with a "@" preceding their name as specified in the template (e.g., @anchor, @anchoree). Thanks to such definitions, KSs can make use of the template without having to worry about the translation, which is taken care of by a language translator. Figure 4-19 shows a knowledge source prescribing the ANCHOR action, and shows how that action is defined and translated into native BB1 low-level functions.

```
                                ANCHOR
level: ACCORD.ACTION
attributes:
template:  (ANCHOR @ANCHOREE TO @ANCHOR IN @PA WITH @CONSTRAINTS)
description:  "Generate the family of positions in which an anchoree satisfies a particular
    constraint (or set of constraints) with its anchor."
bbactions:  ((1 (T) ((EXECUTE ($SET @LIST-OF-CONSTRAINTS (IF (ATOM @CONSTRAINTS)
                                         (LIST @CONSTRAINTS)  @CONSTRAINTS)))
(EXECUTE  ($SET CSS-ANCHOR-RESULTS
       (APPLY (BB1::USER-PACK* 'CSS-ANCHOR- *BB1-SYSTEM-NAME*)
          (LIST @ANCHOREE @ANCHOR @PA @LIST-OF-CONSTRAINTS))))
 (EXECUTE ($SET ANCHOREE-RESULTS (CAR CSS-ANCHOR-RESULTS)))
 (EXECUTE ($SET STATE-FAMILY-RESULTS (CADR CSS-ANCHOR-RESULTS)))))
(2 (ANCHOREE-RESULTS)
 ((EXECUTE ($SET ANCHOREE-ATTR (CADR ANCHOREE-RESULTS)))
 (EXECUTE ($SET ANCHOREE-LINK (CADDR ANCHOREE-RESULTS)))
 (PROPOSE CHANGETYPE MODIFY OBJECT @ANCHOREE ATTRIBUTES ANCHOREE-
 ATTR LINKS ANCHOREE-LINK COMMENT "If the Constraint Satisfaction System comes
 back with modifications for the Anchoree, then modify the Anchoree.")))
(3 (STATE-FAMILY-RESULTS)
 ((EXECUTE ($SET SF-ATTR (CADR STATE-FAMILY-RESULTS)))
 (EXECUTE ($SET SF-LINK (CADDR STATE-FAMILY-RESULTS)))
 (PROPOSE CHANGETYPE ADD LEVEL SOLUTION.STATE-FAMILY NAME (CAR STATE-
 FAMILY-RESULTS)  ATTRIBUTES SF-ATTR LINKS SF-LINK COMMENT "If the Constraint
 Satisfaction System comes back with a state family object, then add it to the solution
 blackboard."))))
links:
is-a:  (ACCORD.ACTION.POSITION)
causes:  (ACCORD.EVENT.DID-ANCHOR)
```

Figure 4-19: Definition of the ACCORD Action "ANCHOR"

4.2.4.2. DECIDE: A Language for Control Reasoning

DECIDE is the language used by control KSs to post whether foci on the control plan should apply to actions, events or states. Figure 4-20 shows its type hierarchy.

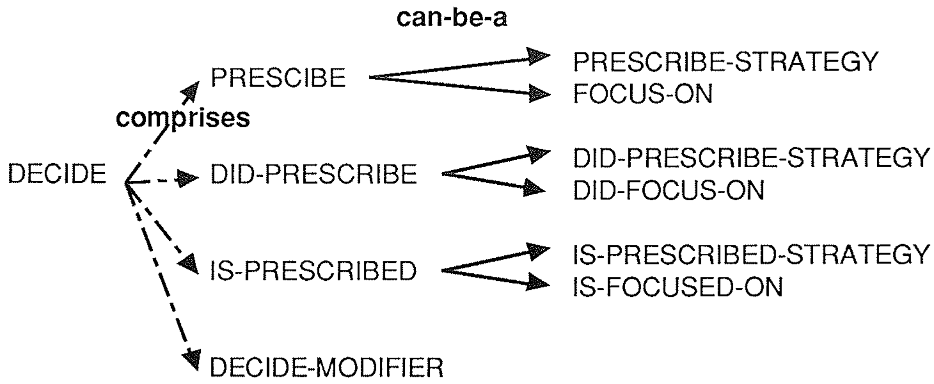


Figure 4-20: Type Hierarchy of DECIDE

4.2.5. Control Knowledge Base with the Control Data BlackBoard and the Control Plan BlackBoard

The CONTROL-KB is used by BB1 for record-keeping purposes. It is split into two BBs: the CONTROL-DATA-BB and the CONTROL-PLAN-BB. The first one maintains information on KSARs, events, language events, and scheduler decisions during a problem-solving run; the second one maintains information on strategies, foci, and heuristics, as well as the overall agenda of the system. The agenda is a frame whose attributes list the triggered, executable, and obviated KSARs; it is updated at every cycle of the run.

4.3. The BB1 Execution Cycle

Having explained SightPlan's use of blackboards and knowledge sources in BB1, we now explain the execution of the system as it steps through problem solving.

4.3.1. BB1 Control Loop

The BB1 scheduler goes through the execution cycle described in Figure 4-21 A cycle starts with the execution of one action. The action causes events. Each of the events of the current cycle is used to try to trigger any of the KSs that implement the system. For each KS that is triggered, the scheduler generates the example objects in the context of the run and generates appropriate KSARs. All of the KSARs—whether just generated or existing from previous cycles—have their preconditions and obviationconditions checked. If preconditions are found to be true and obviationconditions are found to be false, then the KSAR is put on the list of executable KSARs on the agenda. The scheduler then rates each of these executable KSARs and proposes the one with highest rating for execution. To the best of the system's knowledge (which is knowledge of the strategy at that cycle) the highest-rated KSAR is the one with the most preferred action to take; so execution of the preferred action ends this cycle and the next cycle can start. The remaining question then is how the system actually rates KSARs.

ACTION	Explanation
EXECUTE	execute the action selected in the preceding cycle
TRIGGER	use the generated events to trigger KSs in the system; generate KSARs
CHECK	check pre- and obviationconditions of each KSAR
RATE, SELECT, and CONFIRM	rate all executable KSARs; select the one with highest rating for execution and ask the system user to confirm this selection

Figure 4-21: Structure of an Execution Cycle in BB1

4.3.2. Rating Mechanism

At any cycle there can be multiple executable KSARs. This is a desirable situation, as the central design feature of the blackboard architecture is that the scheduler has the capability to decide which of the competing proposed actions to execute. The scheduler makes this decision by assessing how well the action of each of the executable KSARs matches the actions prescribed by the current control strategy. This match is expressed by a numeric rating value ranging (by convention) from 0 to 100, interpretable as ranging from no match to perfect match. Based on this rating the scheduler orders the KSARs and proposes the one with the highest rating for execution; the absolute value of the rating is, therefore, not important. In case of a tie, when the rating values do not allow the scheduler to discriminate between KSARs, the default choice is LIFO: the KSAR that became executable most recently is proposed for execution.

The numeric rating value for a given KSAR is established as follows: the scheduler finds all foci that are active in the current cycle; it looks up the heuristics that implement them and applies each heuristic rating function (the value of an attribute of a heuristic) to the KSAR. These functions return numeric values, which are then integrated with the weights on the heuristics or on the foci into one final rating value: the rating of the KSAR under the given control plan. This heuristic integration function is just another heuristic on the control plan.

When a focus employs an application language such as ACCORD, a rating is established somewhat differently. In that case no heuristic implements the focus, but, instead, a matching scheme is implied by the language. For example, Figure 4-22 illustrates how a focus sentence that uses an ACCORD template matches a KSAR's action sentence. When a noun and/or a verb in the KSAR's action sentence is of the same type as those of the focus sentence, a rating value of 100 is assigned; otherwise a value of 0 is assigned. If modifiers precede a noun or verb in the focus, then the modifying function (the value of an attribute of a modifier) is applied to the matching noun or verb in the KSAR's action, and a value between 0 and 100 is returned.

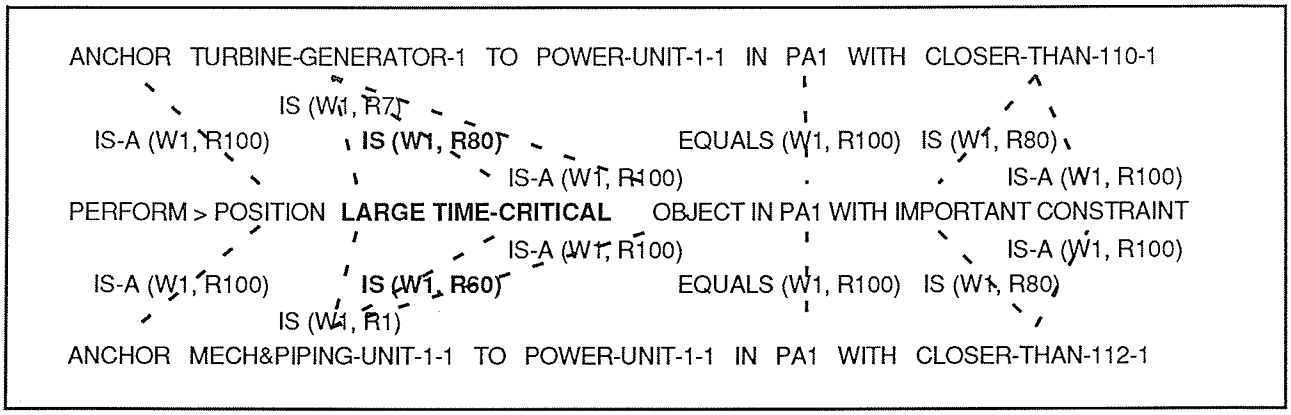


Figure 4-22: Matching of a Domain Action against a Focus

The rating mechanism explained here was based on the match between a focus and the action sentence of a KSAR. A logical extension to this mechanism is to match a focus and triggerconditions, or a focus and preconditions, to establish the desirability of promoting a KS or a KSAR. For more information on this, see for instance the work of [Johnson 87] on goal-directed reasoning. SightPlan only makes use of rating based on action sentences.

4.4. The GS2D Constraint Engine

SightPlan reasons about which actions to execute at each cycle. Actions involve specific objects and constraints. When an action executes, SightPlan passes the constraint and the object pair with their current essential area to a constraint engine, called GS2D, for the computation of constraint satisfaction. For each object, GS2D then returns the subset of essential areas which meet the constraint. Note that only constraint satisfaction between object pairs is handled by GS2D, acting as a procedure called by SightPlan. Issues such as constraint propagation or truth maintenance are dealt with and reasoned about by the SightPlan system itself.

GS2D is a procedural system, implemented in common LISP, that solves geometric constraints between rectangular objects in a two-dimensional world, using a bounded interval representation. It is a re-implementation of the 3-dimensional GSD constraint engine that was used for PROTEAN, adapted for a computationally less demanding 2-dimensional application such as SightPlan.

The fundamental decision in the design of a geometric constraint engine is its representation of space. Given the implementation of objects within SightPlan—objects are rectangles—an orthogonal bounded interval representation was selected to model the possible positions of an object. Objects' axes must be aligned on a single global Cartesian coordinate system, thus they are allowed a 0 and a 90 degree orientation. Within GS2D each bounded interval pair describes a rectangular set of legal locations for the center point of an object, which we call the essential area. A fixed object is therefore described by a single point. An object that is not fixed, has dimensions and a set of orthogonal bounded intervals for each of its two possible orientations.

Given this representation of space and objects, we decided to limit constraints to be rectilinear. That is, the distance between two objects is, by our choice, the minimum of the horizontal or the vertical absolute distances between their edges. Thus the implementation of constraints is simplified to a series of expansion and intersection operations on sets of rectangles. This greatly speeds the operation of the constraint engine at the cost of reduced precision in computing distance.

Figure 4-23 shows how GS2D reduces the sets of possible positions of two objects to allow only for positions where the objects are CLOSER-THAN a given distance to each other. People unfamiliar with GS2D will need to get used to the fact that GS2D computes on rectangles representing sets of possible locations for the centerpoint of each entity, rather than on the entities themselves. Also, the definition of distance used in GS2D may be counter-intuitive.

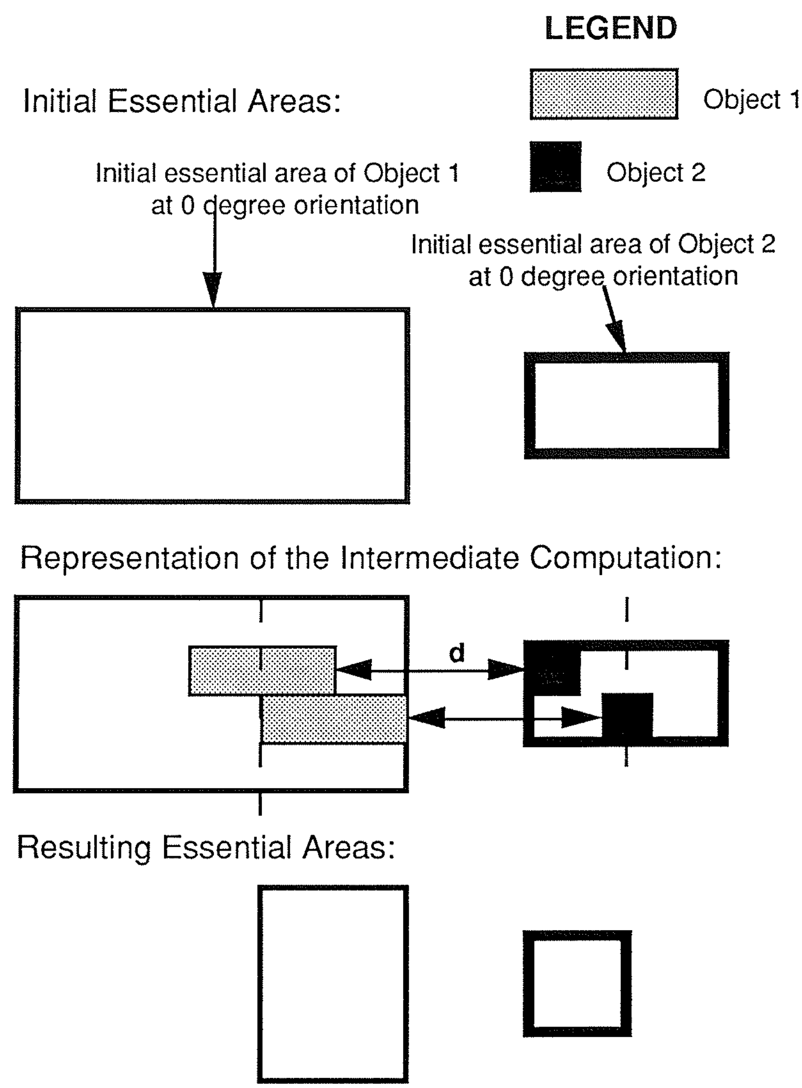


Figure 4.23: Application of the CLOSER-THAN Constraint to Two Objects

GS2D was designed to be extensible. The system provides a set of low-level primitives that operate on bounded intervals and a number of high-level constraints defined in terms of these primitives. Constraints not provided by the system can be built using a combination of the two. The high-level constraints fall into two types: binary and unary acting which affect both of, or only one of, the objects respectively. Binary and unary constraints used by SightPlan, such as (Further-than obj1 obj2 distance) or (North-of obj1 obj2), are defined in terms of one or more primitive GS2D constraints. The constraints that SightPlan uses are obviously all supported by GS2D, but the SightPlan and GS2D programs are implemented as totally independent modules. GS2D is available for use by other application systems, and SightPlan could easily make use of another constraint engine if it needed to. See [Confrey 88] for a more detailed account of GS2D.

4.5. SightView: An Interactive Graphical Interface for SightPlan

The initial version of SightPlan used the TI Explorer's™ graphic LISP primitives to provide a graphical display of the site, with permanent facilities, and sets of possible locations for temporary facilities currently included in a solution. SightPlan had KSs to decide when to display objects and what to display. This monochromatic display occupied roughly the lower half of the TI Explorer™ screen. Consequently, although of considerable value in debugging and explanation, it had only limited resolution, and was of little value in distinguishing between the essential areas of different temporary facilities once any significant number of them were being displayed. Moreover, SightPlan spent a significant number of cycles and processing time on reasoning about display.

We, therefore, developed a colored graphical display for SightPlan to run on a second processor. We chose an Apple Macintosh II™ computer. Its high resolution color display would permit a user to discern far more detail about the state of a solution under development. The display system running on the Macintosh II™, called SightView, is connected via an ethernet connection and a Communication Interface [Hewett 88b] to SightPlan running on a TI Explorer™ (see Figure 4-24). Each time SightPlan updates an object's essential area, that information is passed within the same action to SightView for display. In this way, SightPlan only spends a minimum amount of time on handling display information. The representation of objects and their positions is consistent throughout the SightPlan system: the object data structure used by GS2D (essential areas) and by SightPlan (state families) is also used for data transfer to the Macintosh™ display.

The SightView user can open multiple windows and can set each individually to view any portion of any partial arrangement at any scale, and to display a selected subset of the facilities. By selecting the temporary facilities of interest in different windows and choosing colors to provide contrast between them, it is possible to obtain a vivid and intuitive understanding of the state of a solution (see Fig. 4-25

Window#1 in Figure 4-25 shows the whole site, #3 a close up, #2 and #4 different facilities of interest. In #4 the existing legal locations of the Air Quality Control System (AQCS-2) is highlighted with the grey-shade line. The open dialogue box permits the user to decide whether or not to display the object in the window, to change the color of the object's essential area display, to reduce the object's essential area or to fix the object's location.

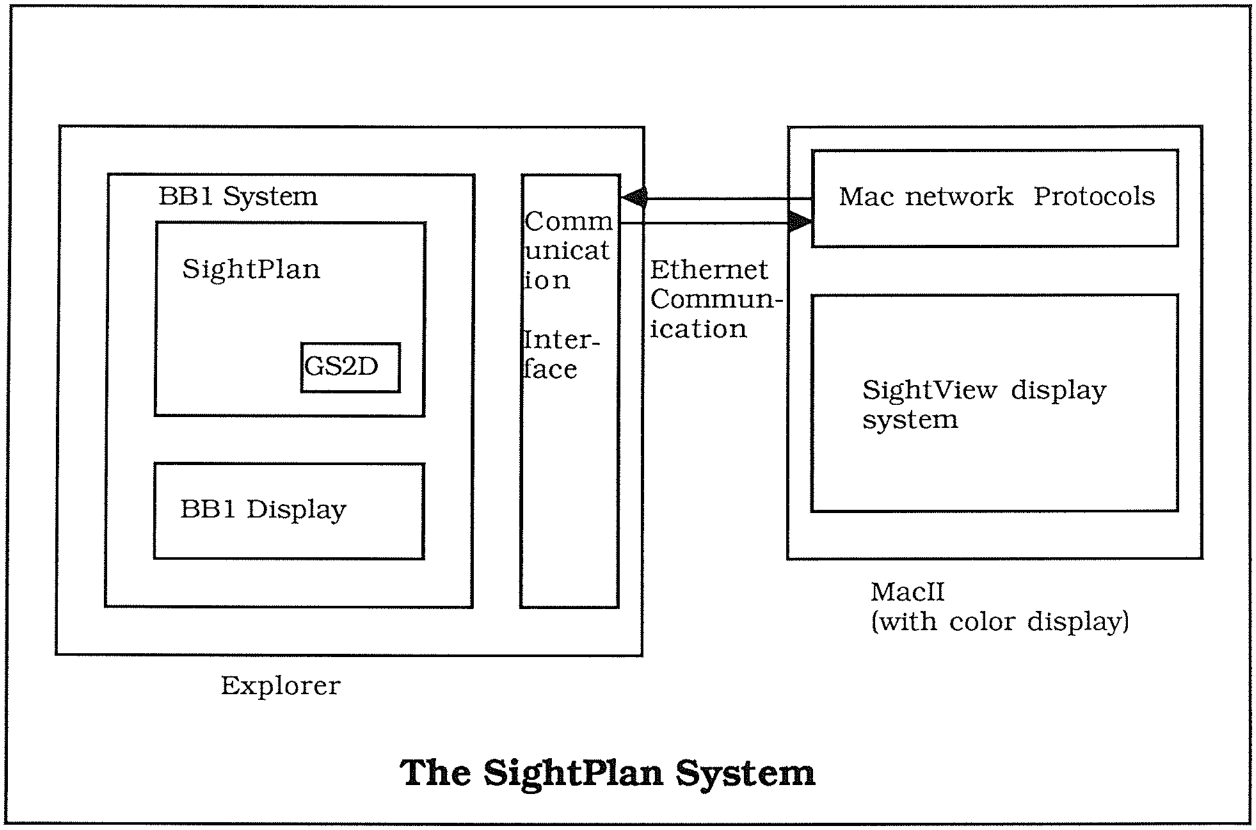


Figure 4-24: The Configuration of the Integrated Sightplan System

SightView has been designed to interact with SightPlan. A user can select a temporary facility and give it a precise location somewhere within the facility's set of currently legal locations. Alternately, a facility's set of legal locations can be arbitrarily reduced within existing legal limits to satisfy some user preferences about the region of its ultimate location. It this way, the user is involved as one more knowledge source in the SightPlan system.

When the user graphically restricts the possible locations of a temporary facility via the Macintosh™ interface, that information is sent back to SightPlan. SightPlan then checks to see which constraints need to be reapplied with other objects, so that it can further restrict their legal positions. The new essential areas for all objects are then re-displayed and the user can make additional positioning decisions. Alternatively, the user can decide not to intervene and allow SightPlan to proceed with the position actions it selects using its stored knowledge of site layout.

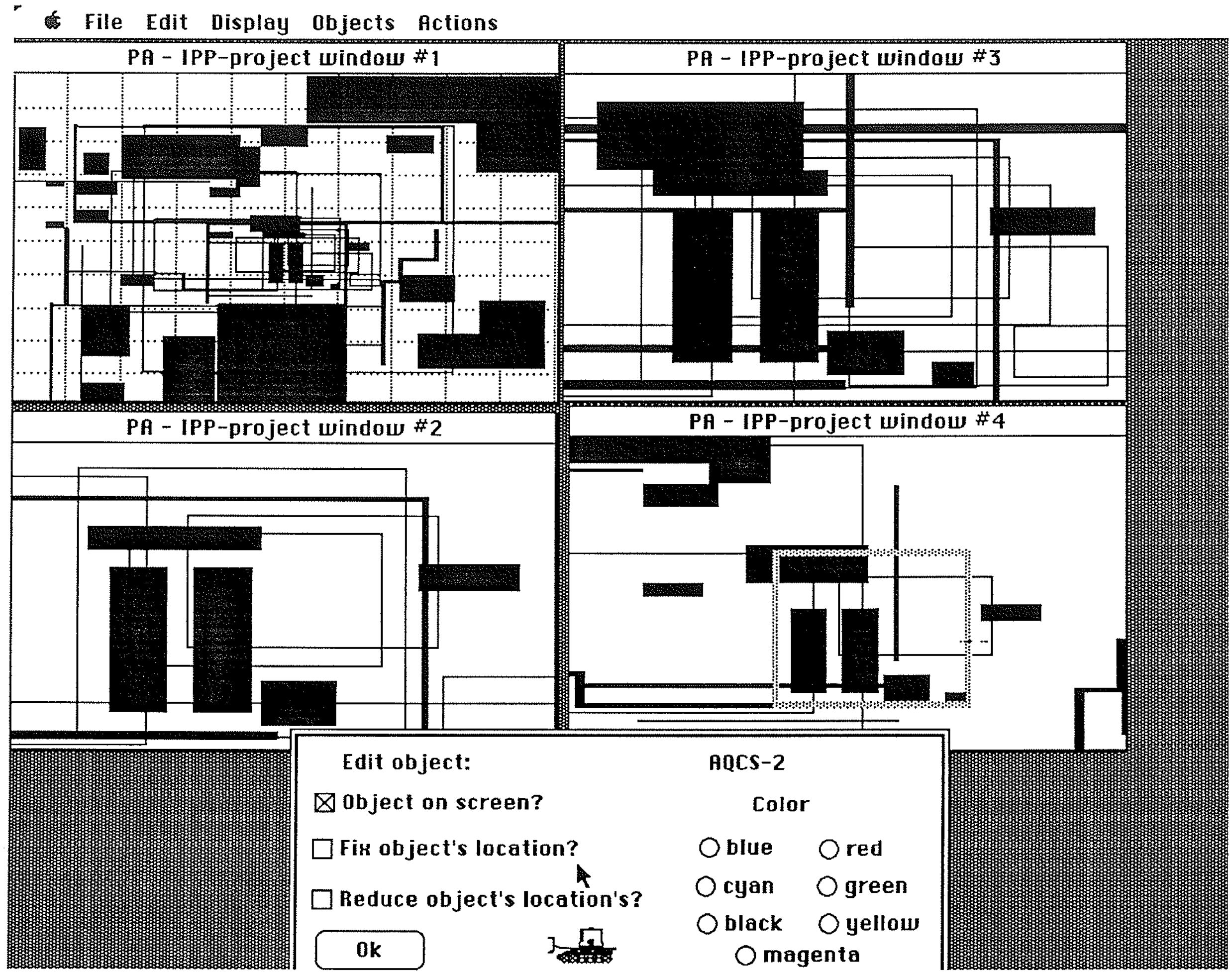


Figure 4-25: Four Views of the same Site in the SightView Graphical Display System of SightPlan.

We decided not to permit the user to choose a location for a temporary facility that lies outside the boundaries of its current essential area. Permitting this would imply relaxing a previously met constraint and the system would need a dependency-directed backtracking mechanism, or some other form of truth maintenance capability to determine which constraints needed to be reapplied or relaxed to permit the desired positioning of the facility. Implementing this capability is a potential extension to the SightPlan system that is feasible (since BB1 keeps records of all of its actions including the application of constraints), and might be attempted by the second author in the future.

5. Knowledge Acquisition for the SightPlan System

SightPlan's first prototype system was implemented so that the civil engineering researchers on our team could gain familiarity with the BB1 architecture. Thanks to the close collaboration between a BB1 group member and a SightPlan group member this initial model was developed in just a few days. The prototype was a simplistic system, consisting of about five objects to be positioned on a site with one fixed facility. Its knowledge sources were taken directly from the PROTEAN application and needed only minor adjustments for their application in SightPlan. This working model allowed us to firm up our initial ideas for a site layout model so that these new ideas could support a proposal to request external funding.

The subsequent implementations of SightPlan were to address full scale site layout problems. We chose to do two case studies for SightPlan in order to acquire realistic knowledge for the various blackboards, and to challenge the architecture with a problem of substantial complexity.

The first case study was the Intermountain Power Project, a coal-fired power plant of two 750 MW units, located in Delta, Utah. We interviewed representatives of the project manager, the architect/engineer, and the construction manager on the project and learned about their different viewpoints on and involvements with the layout task (see acknowledgements for names of persons interviewed). The project manager played the role of global coordinator. The architect/engineer was responsible for the layout of the long-term facilities, i.e., the facilities that are on site for most of the duration of construction of the project, and that sometimes remain on site to function as maintenance facilities upon project completion. Several of these facilities were buildings that needed to be designed like permanent facilities are. Thus, the architect/engineer designed and laid out these facilities before passing on the site arrangement drawing to the construction manager. The construction manager was responsible for the long-term laydown areas and short-term assembly yards. Long-term laydown areas were allocated upon award of the subcontracts so their layout was done as a pre-planning task upon commencement of construction. Although these areas met the needs for space and access (e.g., by railroad or by road) by the contractors, they were not laid out in the prime area of the site. This prime area, immediately surrounding the power units under construction, was left open so that short-term assembly yards could be located there at later times.

SightPlan was provided with a description of the Intermountain Power Project in terms of approximately 50 permanent facilities, roads, and railroads (all with a given location on the site). The system is charged with the task of laying out about 20 temporary buildings and another 25 long-term laydown areas, and it makes use of the strategy that we learned from the architect/engineer and the construction manager to do this. This strategy implements the division of tasks in the power plant engineering industry, and reflects the early commitment approach that people use.

To assess the generality of the site layout knowledge acquired for our fist case study, we decided to validate the strategy that we had learned there on a second site. The second case study was the American 1 Power Project, a co-generation plant of 120 MW, burning oil or gas, and located in King City, California. We interviewed the project manager and several superintendents on the site, and discussed the layout of the temporary facilities and the material laydown areas.

SightPlan was provided with a description of the American 1 Power Project in terms of approximately 20 permanent facilities and roads, 5 facilities, and 10 laydown areas. Because this second project was so much smaller in scope, the managers spontaneously raised the issue of layout changes over time while explaining the layout of long-term facilities to us. This additional knowledge about site layout temporal strategy allowed us to extend the Expert Strategy to reason explicitly about layouts that change over time.

6. SightPlan Experiments and Performance of the System

The information contained in the SightPlan system is stored on the various blackboards described in the previous section. The objects that need to be positioned on a specific sites and the constraints to be satisfied between them are written on the PROBLEM BB, e.g., the INTERMOUNTAIN BB. Moreover, the SightPlan system contains an explicit representation of the strategy it follows to construct a solution. Such a strategy is important in that it determines what solution will be generated.

For example, the strategy determines the order in which primitive objects meet their constraints with respect to other objects, what objects need to be grouped together in one arrangement before the entire arrangement is located with respect to other objects, and so on. Besides reasoning about positions of objects, SightPlan can also reason about when and how to display layout information or to acknowledge user changes.

The declarative knowledge environment that constitutes SightPlan, with its individual object frames, loosely-tied blackboards, and independent knowledge sources, allowed us to experiment with alternate strategies and input data from different sites. It formed an excellent test-bed for experimentation. The SightPlan experiments are described in this section.

6.1. Expert Strategy

The first model, which we term the Expert Strategy, emulates the strategic decisions and steps taken by experienced designers and field managers while laying out the Intermountain Power Project (IPP) site. This model reflects the division of tasks that exists in the facility engineering process as it is practiced today. In addition to designing and laying out the permanent facilities, the Architect-Engineer (AE) on IPP also laid out what we call "Long-term Temporary Facilities." After the AE completed its work, the Construction Manager (CM) laid out the "Long-term Laydown Areas" for the major contractors on site. Short-term temporary facilities are left out of this model. Including them would have substantially increased the complexity of the computations, without contributing to our research goals.

This divided process was represented in SightPlan: SightPlan generates layouts at two instants in the design-construct lifetime of the project. The first layout is generated in the design phase. The second layout is generated after the civil contractor has commenced work on foundations, but before other major contractors arrive at the site. SightPlan treats the layout it generates at each of these snapshots in time as a static layout.

6.1.1. Implementation of the Strategy

Table 6-1 charts how the Expert Strategy functions over the duration of a problem-solving session. The left column lists the cycle numbers. To the right of it are the executed domain actions.

CYCLE	ACTION	
14	create pa1	**ARCHITECT-ENGINEERS**
19	include context in pa1	
24	include fixed objects in pa1	
29-30	include and identify occupied-space in pa1	
34-37	include areas in pa1	
43	include first aggregate in pa1	
44	size aggregate context	
45	shape aggregate context	
47	include second aggregate in pa1	
48	select aggregate layout plan	
55	create pa2	
60-72	include object in pa2	
77	orient pa2	
82-136	anchor object in pa2	
142	shape context pa2	
145	transfer size from aggregate context in pa2 to aggregate in pa1	
152	orient pa1	
158-165	position first aggregate in pa1	
167-175	position second aggregate in pa1	
182-194	refine pa2	
14	create pa	**CONSTRUCTION MANAGERS**
19	include context	
24	include fixed objects	
29	include and identify occupied-space	
33-36	include areas	
40	include laydowns	
44	orient	
50-136	position objects in zone or outside of zone	
138-188	position objects so that they don't overlap with permanent facilities	
190-293	position large objects first, with as close as possible constraints, then update occupied space and proceed with following object	

Table 6-1: Some Cycles from SightPlan's Expert Strategy Applied to IPP

The first half of the IPP's layout is generated by SightPlan mimicking the actions of the AE. SightPlan starts by creating a first partial arrangement (PA1) (Cycle 14) on the SOLUTION-BB. *(This is as if a person took a blank sheet of paper.)* It includes the context, that is, it picks what site it is going to lay out (Cycle 19). *Including* consists of picking an example on the PROBLEM-BB and creating an instance for it on the SOLUTION-BB. This model lays out the IPP site in Delta, Utah (DELTA-SITE), whose site boundaries are predefined. These boundaries further define limits on the location of objects that will be positioned in that context.

Following this, all objects that have a predefined and fixed location on the site are included (Cycle 24). These are the permanent facilities, the roads, and the railroads. *(So far, the person laying out the site has identified what the site and the permanent facilities look like.)* SightPlan draws them at their known location on site.

The system recognizes that there are two major aggregations of objects to be included, the IPP-CONSTRUCTION-ENTRANCE (*ENTRANCE*) and the IPP-CONSTRUCTION-FACILITIES (*FACILITIES*), and chooses to include those. Because they do not have a given location on site, they will need to be positioned. An aggregation of objects is represented by a so-called *aggregate object* on the PROBLEM-BB, that is, an object that *includes* other objects. As of yet, these two aggregate objects are shapeless and undimensioned. Figure 6-1 shows the "includes" link between the ENTRANCE aggregate and its parts. In this example, each part has a prespecified geometry represented by the shaded rectangles drawn to scale to the right of the part's label.

SightPlan includes and identifies the occupied-space (Cycles 29-30). *(The layout designer looks at the problem and recapitulates what space is no longer available.)* The program then sub-divides the site in the way that is specified on the PROBLEM-BB (Cycles 34-37). The site's sub-areas are the WORK-AREA, the CONSTRUCTION-AREA, the COAL-HANDLING-AREA, and the OPERATIONS-AREA.

Before the aggregates are positioned on site, the system estimates their size and initializes their shape. As a first guess, the area of the FACILITIES is estimated to be 1.25 times the sum of the areas needed by its parts (Cycle 44). As soon as the FACILITIES are dimensioned, they are molded in a rectangular shape with a length-to-width ratio of 3 to 1 (Cycle 45).

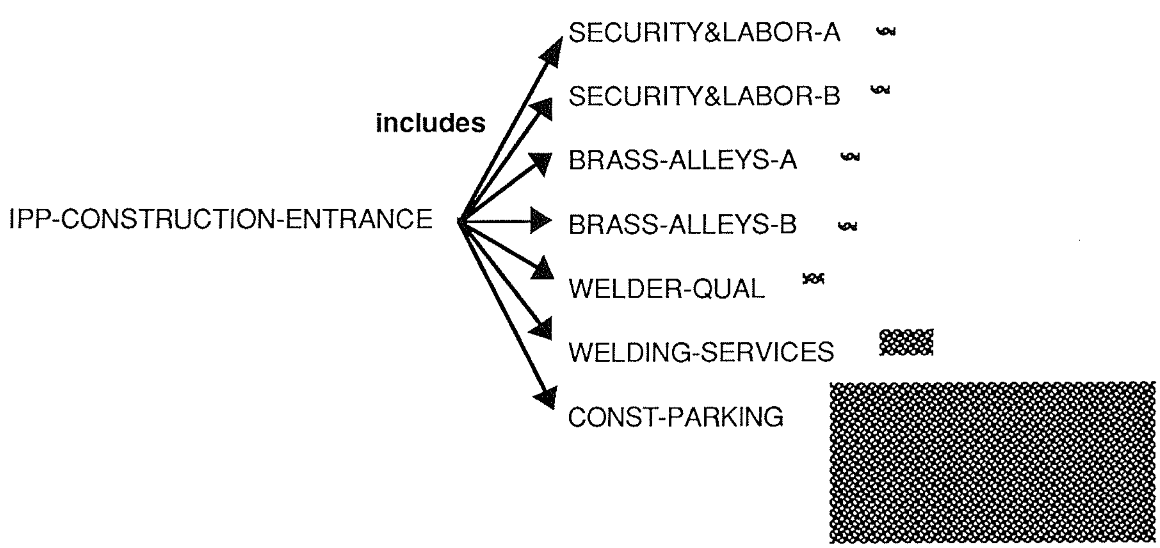


Figure 6-1: Example of the IPP-CONSTRUCTION-ENTRANCE Aggregate Object The figure shows the includes links to its parts in the graph, together with the geometry of each part.

SightPlan could shape the ENTRANCE in a similar manner, but from looking at the types and the number of constraints that the ENTRANCE's parts have with one another, it determines that the ENTRANCE is a *tightly constrained* aggregate. For such aggregates, SightPlan knows the following alternate strategy: SightPlan builds a second partial arrangement (Cycle 55).

No specific context is known, but all the parts of the ENTRANCE need to be included (Cycles 60-72). (Note that some intermediate steps are for display actions.) SightPlan picks the largest one as anchor of the arrangement (Cycle 77), with respect to which the other parts will be positioned.

In order of decreasing size, each of the other objects is positioned in this arrangement so that, one at a time, SightPlan meets its "non-overlap" constraint, "North-of" and "between-short-sides" constraints, "closer-than" constraint, and/or "adjacent-to" constraint with the anchor (Cycles 82-136).

Other constraints between the parts remain, but SightPlan decides that, after meeting the above constraints, it can reasonably estimate the area needed to fit this arrangement (Cycle 142).

With this estimate, the system focuses again on the first arrangement PA1, and transfers the size of the context of the ENTRANCE aggregate in PA2 to be the dimensions of the ENTRANCE in PA1 (Cycle 145). SightPlan chooses the first constructed power unit (unit 1) as anchor in PA1 (Cycle 152).

Then, it locates the FACILITIES by finding the set of positions that zone these FACILITIES inside the construction area (Cycle 158), outside the work area (Cycle 160), and where they do not overlap with any occupied space (Cycle 162).

Finally, in order to pick one single position from those that remain possible, SightPlan selects the one as close as possible to unit 1 (Cycle 164) and updates the occupied space to account for this positioning (Cycle 166).

This process repeats for positioning the ENTRANCE (Cycles 167-175), except that two more constraints are applied. These express that the ENTRANCE must be above the main access road to the site and below unit 1.

Now that overall positions for each of the aggregates have been found, SightPlan can further detail the parts of these aggregates. Attention focuses on the second partial arrangement again. Of the ENTRANCE, the only part that has constraints with objects outside of the aggregate is the WELDER-TESTING. Its constraint, to be as close as possible to unit 1, is met. The constraints on the other parts are also met, object by object, and this results in a single location for each. In a similar way, SightPlan can then proceed to create a third partial arrangement and layout out the parts of the FACILITIES aggregate. This concludes the task of the AE.

The second half of the IPP's layout is generated by SightPlan mimicking the actions of the CM. Since the CM's layout was generated by a person different from the person who did the AE's layout, and there is a real separation between the two tasks, we implemented SightPlan in two separate systems.

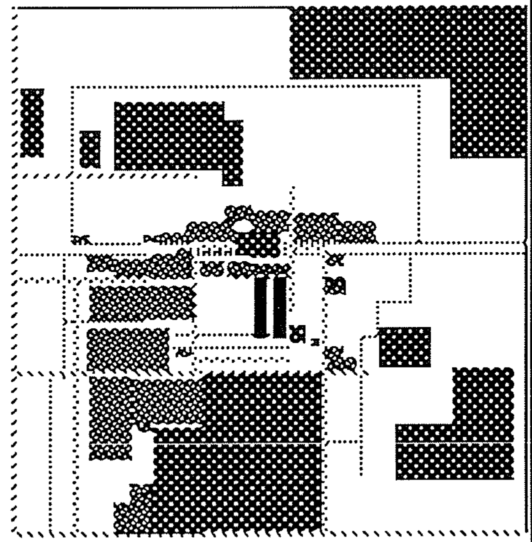
SightPlan starts anew by creating a first partial arrangement (Cycle 14), including the context (Cycle 19) and the objects with predefined and fixed location on the site (Cycle 24). This includes the two aggregate objects since these now have a fixed location. (The starting layout looks the same as Figure 6.19).

SightPlan proceeds as it did in the AE's solution method, by identifying the occupied space (Cycle 29), and by including the sub-areas of the site (Cycle 33-36). All long-term laydown areas (LAYDOWNs) for the contractors need to be laid out by the CM, so they are included at this time (Cycle 40). Again, unit 1 is chosen as anchor (Cycle 44).

Each of the LAYDOWNs is located according to its zoning constraint (Cycles 50-136).

SightPlan positions the largest LAYDOWN in its area by meeting the LAYDOWN's constraints with the fixed facilities on site in the following order (numbers in parentheses are numeric weights for the matching constraints): zoned-in (0.98), zoned-outside-of (0.95), non-overlap-set (0.93), non-overlap (0.9), at-long-side (0.85), betw-short-sides (0.83), adjacent-to (0.8), north-of (0.7), south-of (0.7), west-of (0.7), east-of (0.7), closer-than (0.6), further-than (0.5), parallel (0.4), perpendicular (0.4), discrete-sample (0.3), as-close-as-possible (0.2), pick-one (0.1). When all constraints are met, the last one to be applied is the as-close-as-possible constraint, which picks one single position from those that remain possible. Then the occupied space is updated.

This process repeats for the second largest object, and continues for the other LAYDOWNs in order of decreasing size (Cycles 103-152). When all laydown areas thus have a single position, the solution layout has been achieved. Figure 6-2 depicts SightPlan's CM solution layout.



SOLUTION.PARTIAL-ARRANGEMENT.PA1

Figure 6-2: Solution Layout Generated by the Expert Strategy on IPP

6.1.2. Discussion of the Expert Strategy

The Expert Strategy of SightPlan demonstrates that the constructive assembly method provides an excellent basis for a model that mimics the actions of field managers laying out sites. We discuss the results of modeling the expert strategy in this section.

6.1.2.1. Layout Method and Strategy

SightPlan makes use of each of the domain knowledge sources that implement this method: SightPlan 1) positions objects one at a time, 2) abstracts or specializes arrangements at different levels of detail, and 3) works on partial arrangements. In addition to this, SightPlan reasons about the size and shape of (aggregate) objects. Moreover, the declarative knowledge representation environment and the opportunistic reasoning mechanism of BB1 are found to be suitable for articulating layout knowledge and manager's choices at each step in the problem-solving process. SightPlan's knowledge is represented in 1) the definition of objects in the layout, 2) the constraints between the objects, and 3) the heuristics of the strategy that generates the arrangement. These heuristics together with the foci, both constituting the strategy, are implemented by SightPlan's control knowledge sources.

The power of SightPlan lies in the flexibility with which it decides what to do. For example, SightPlan can choose to lay out aggregate objects first, or can choose to lay out the larger objects first. SightPlan can alternate between working on the partial arrangement of the ENTRANCE aggregate, working on that of the FACILITIES aggregate, and working on the partial arrangement of the overall DELTA-SITE.

The Expert Strategy of SightPlan is the product of modeling how people lay out sites. Therefore, it is also reflects the fact that people adjust their strategy of searching for a solution to cope with human cognitive limitations. A first manifestation of human cognitive limitations affecting problem-solving occurred at the moment of definition of the site layout problem. The problem scope was restricted, because including all the variables that possibly play a role in layout generation would have led to an overly complex definition, impossible for people to handle. Similarly, site layouts are defined at a snapshot in time, yet cover facilities on site over ranges of time.

A second manifestation of human cognitive limitations affecting problem-solving occurs during problem-solving. People can only focus attention on one or a few objects and constraints at any one time, and as a result of this they resort to an early commitment strategy for positioning objects in a layout. For example, the CM follows an early commitment strategy when choosing one single object, meeting its constraints one at a time, finally picking one position from the set of allowable positions, and placing the object there, before the cycle repeats with the following object. Once an object is positioned, it remains in that location, and no other objects can be positioned in that location thereafter.

The Expert Strategy exhibits several other ways in which the AE and the CM used early commitment to reduce the complexity of the layout process at IPP. These include: partitioning over time, partitioning over space, aggregating in time, and aggregating in space.

6.1.3.2. System Performance

System performance in terms of absolute speed has not been a concern in this research. For those readers who wish to know the order of magnitude of system run-time, the AE and the CM Model take from two to three hours on an Explorer II™ to construct a solution. Readers should realize, however, that the system has not been optimized for speed. The BB1 architecture keeps information about each cycle's actions, events, and scheduler decisions around for bookkeeping purposes and to facilitate learning from looking at a system's run.

Altman investigated how one might partially compile knowledge when a sequence of actions is well-known ahead of time, in order to improve system run-time, at the cost of losing some flexibility [Altman 87]. Some knowledge sources in SightPlan use this idea. For example, the domain KS include-all-fixed-objects-loop executes in one cycle and includes all objects with fixed location on site at once, rather than including one object per cycle. Another way in which SightPlan's execution speed was increased is by creating an object named occupied-space to keep track of the space occupied by facilities with fixed locations on site. This reduced expressing that no two facilities could overlap (factorially many constraints) to a linear number of single constraints between each facility and occupied-space.

6.1.3.3. Quality of solutions

The early commitment approach is pervasive in the Expert Strategy, and it appears to work well on IPP, as SightPlan succeeds in generating a solution. In general, however, the construction method that is used by SightPlan is a weak method, that is, it cannot guarantee that a solution will be reached, even if a solution is known to exist. A solution can be constructed for IPP because the problem is stated in such a way that it is underconstrained, i.e., there exist multiple arrangements in which all the constraints on the objects are met.

A problem is truly underconstrained when there are inherently few constraints that need to be satisfied; but any problem can be made to be underconstrained by phrasing the knowledge about the problem, more specifically, about the objects and the constraints, in this way. The managers at IPP perceived the site as a spacious one, so that there would be no lack of space overall. Of course, project sites always lack prime space, and allocating prime space is what site layout is about. Therefore, it is not necessarily clear whether or not the IPP problem is inherently under– or overconstrained. A few factors that stem from the knowledge acquisition and representation, however, make the IPP problem more underconstrained.

1) It is known ahead of time that all objects will fit on site, and additionally, that a solution layout exists because a layout arrangement for temporary facilities was available (hindsight is 20/20!). This bias in the phrasing of the layout problem is attributed to the fact that the knowledge was acquired only after the site had been laid out and implemented. Moreover, because we defined object's shapes by taking their shape from the solution arrangement, it is logical that objects will fit at least in that solution position on site when SightPlan positions them. Reasoning about size and shape in more flexible ways in SightPlan requires additional research.

2) It is known ahead of time that all constraints can be met. As the knowledge engineers who defined the constraints, we made sure that all constraints specified could be met, and we defined constraints loosely so that there would be some alternative positions for objects. Also, the SightPlan model can, at this time, only "understand" a limited vocabulary of spatial constraints, so constraints were limited in expressiveness. Another shortcoming of the implementation is that all constraints in SightPlan are currently specified before the program starts its layout generation. We feel that people add additional constraints during problem-solving in order to be able to select among alternative solutions. These constraints may not be known ahead of time; rather, they are defined opportunistically as problem solving proceeds.

AI researchers have identified these problems before, and have proposed several ways of avoiding them. For example, one may assess the quality of the outcome of an action before taking the action, one may undo the results of actions that were taken earlier, or one may postpone commitment. We will return to postponed commitment and investigate how that strategy affects problem-solving in Section 6.3.

6.1.3.4. Validation

It did not come as a surprise that SightPlan could construct layouts very similar to the ones that had been designed by the people involved on them. This is because of the way in which objects and constraints had been defined in the project-KB of Intermountain and of American 1, and therefore this test of the validity of the model—relying on a comparison of the resulting layouts—is a biased one.

SightPlan was validated by one of the project managers who had been involved on the Intermountain Project. He followed the program as it pursued the Expert Strategy and commented on its actions, mainly by pointing at the graphical layout display.

His main criticisms involved constraints that the prototype SightPlan lacked, i.e., SightPlan accepted positions for objects that it should not accept. He found that a second shortcoming of the system was its inability to reason about the shape of laydown areas. This latter deficiency is a shortcoming in the completeness of the knowledge acquisition rather than of the architecture. In two instances (the construction-entrance and construction-facilities aggregate objects) SightPlan does reason about sizing and shaping of objects. Extensions to the knowledge base of the example project might be worthwhile to demonstrate the full strength of the system.

We were pleased to find the Expert Strategy that we formalized on the Intermountain site proved to be largely transferrable to the American 1 site. This observation leads us to claim that the Expert Strategy may well be a good initial description of a more generally applicable site layout strategy; however, additional validation and strategy refinement are needed to substantiate this claim.

6.2. Temporal Reasoning Extension to the Expert Strategy

By developing a temporal strategy and applying it to the American 1 problem we demonstrated how the Expert Strategy might be extended to accommodate more variables in the model, and how such an extension can lead to even better layout solutions. That fact that the BB1 architecture and the constructive assembly method could easily incorporate this extension confirms some of the potential strengths that we had counted on in selecting this environment.

The second model consists of applying the same Expert Strategy to another construction site, that of American 1, with the intent of validating and assessing the generality of the Expert Strategy. The scope of American 1 was substantially smaller than that of IPP; this made the implementation far more manageable. The part of the expert strategy that prescribed separate partial arrangements for aggregate objects could be left out of this smaller system. Based upon application of this strategy to the American 1 site, we were pleased to find that SightPlan could successfully lay out this site as well. Upon this success we extended the Expert Strategy to do reasoning about changes in the layout over time. The revised strategy is called the Temporal Strategy.

6.2.1. Implementation of the Temporal Strategy

The implementation of the Temporal Strategy consisted of attaching an additional attribute to objects on the AMERICAN1-BB to show the time interval during which they were needed on site, modifying KSs to take such time intervals explicitly into account, and adding a few domain and control KSs to the SightPlan system.

In SightPlan's Expert Model, KSs referred to sets of objects of an imposed type. Objects in such a set typically appear on the PROBLEM-BB and are grouped under one level. For example, the goal of INCLUDE-CONSTR-FACILITIES is true when all the objects at the level AMERICAN1.CONSTRUCTION-FACILITIES have an instance on the SOLUTION-BB. In the implementation of the Temporal Strategy, such references to objects at a level are restated as conditions on the time interval that they are on site, and all objects in the problem are tested for these conditions. Thus, instead of including fixed facilities first, construction facilities next, followed by laydowns, SightPlan now includes objects in order of decreasing total length of time on site.

The domain KS, SITUATE-OBJECT, is new to the system. This KS adds an object to the attribute **situated-objects** of the PA that matches the object's time frame, and creates additional PAs to cover the preceding and following time periods if needed. The domain KS INCLUDE-OBJECT now has as a precondition that an object must appear in the attribute situated-objects of a PA before it can be included in that PA.

Table 6-2 gives the main actions of SightPlan's Temporal Strategy applied to AM1. SightPlan creates a partial arrangement PA1 (Cycle 14), includes the context (Cycle 19), includes all *permanent* objects (Cycles 24-28), determines the occupied space (Cycle 34), and divides the context up into sub-areas (Cycles 38-40).

From the remaining objects that can be included in PA1, SightPlan selects the one that is on site for the longest time period; in case of a tie, it selects the largest of those. It *situates* that object with respect to PA1 (currently there is only one PA on the SOLUTION-BB). As it turns out, this first object is on site for the entire project duration, so SightPlan lists it in the attribute **situated-objects** of PA1 (Cycle 47) and it subsequently includes that object in PA1 (Cycle 48). SightPlan includes the two other long-term facilities in PA1 in a similar manner (Cycles 50-51 and 53-54), and selects and anchor for the arrangement (Cycle 60).

CYCLE	ACTION
14	create pa1
19	include context
24-28	include permanent objects
34	include and identify occupied-space
38-40	include areas
47	create situating pa
48	include object in pa1 because time intervals match
50	create situating pa
51	include object in pa1 because time intervals match
53	create situating pa
54	include object in pa1 because time intervals match
60	orient pa1
66-72	position long-term facilities in areas
79-80	position largest object with important constraints
83-88	position second largest object
95-105	position last object
113	create situating pa for short-term facility (pa2, pa3, pa4)
117	include facility in pa that matches time frame (pa3)
119-123	similar to 113-117, include in pa6
125-129	similar to 113-117, include in pa9
136-140	position each object in its pa within the requested area
147-151	position object in its pa so that it does not overlap with the fixed facilities
153-159	position object with its preference constraint

Table 6-2: Some Cycles from SightPlan's Temporal Strategy Applied to AM1

The system positions each long-term facility in PA1 by meeting the zoning constraints (Cycles 66-72). Subsequently it selects the largest facility and satisfies its remaining constraints in decreasing order of importance (Cycles 79-80). When a fixed position is determined SightPlan updates the occupied space. Then, it repeats this process for all facilities that are on site for the entire duration of construction of the project (Cycles 83-88 and 95-101 respectively).

When SightPlan selects the first object that is not on site for the entire project duration, it determines that it needs to create three new PAs (PA2, PA3, and PA4) to detail PA1, and that the object must be included in PA3 (Cycle 113). Following this, SightPlan includes the object in PA3 (Cycle 117). Similarly, SightPlan creates PA5, PA6, and PA7 for the second object and includes the object in PA6 (Cycles 119-123), and creates PA8, PA9, PA10 for the third object and includes the object in PA9 (Cycles 125-129).

The system positions all three objects in their respective PAs by meeting the zoning constraints (Cycles 136-140), followed by the remaining constraints in strategic order, and updates the occupied space. This results in a total of 9 PAs, which are shown in Figures 6-3a through 6-3g These PAs show different site layout stages over the duration of construction of the AM1 project. Note that one area is used for three separate purposes over the duration of the project. Figure 6-4 outlines the time frame of each layout with respect to the entire construction duration.

Figure 6-3: Construction Time Sequence of Solution Layouts Generated by the Temporal Strategy Applied to AM1.

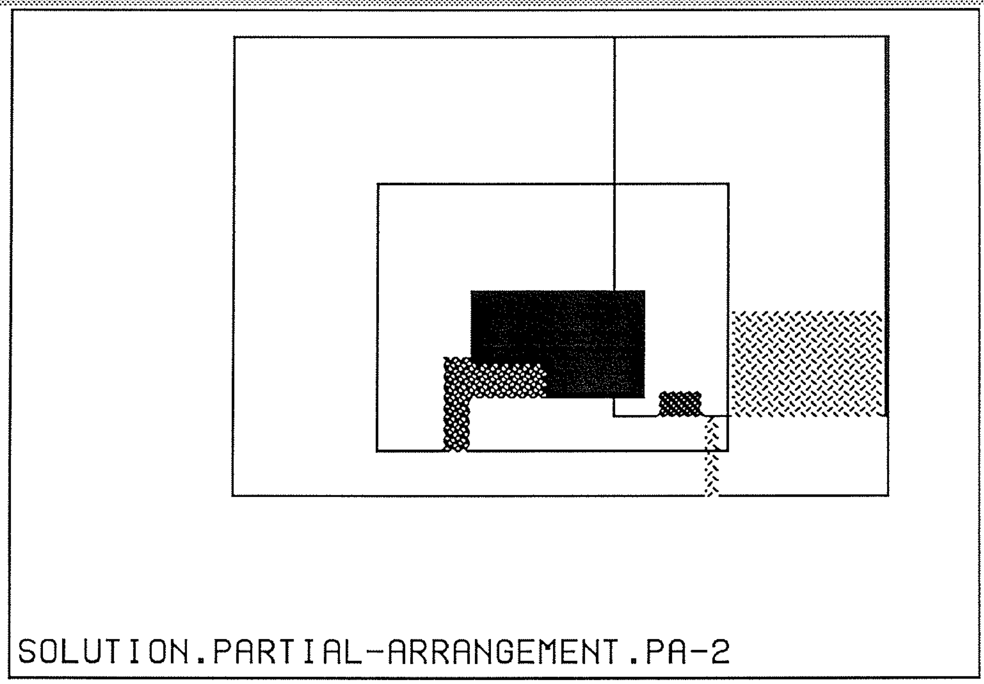


Figure 6-3a: PA2, Time Frame 870901–871201. No short-term laydown areas are on site.

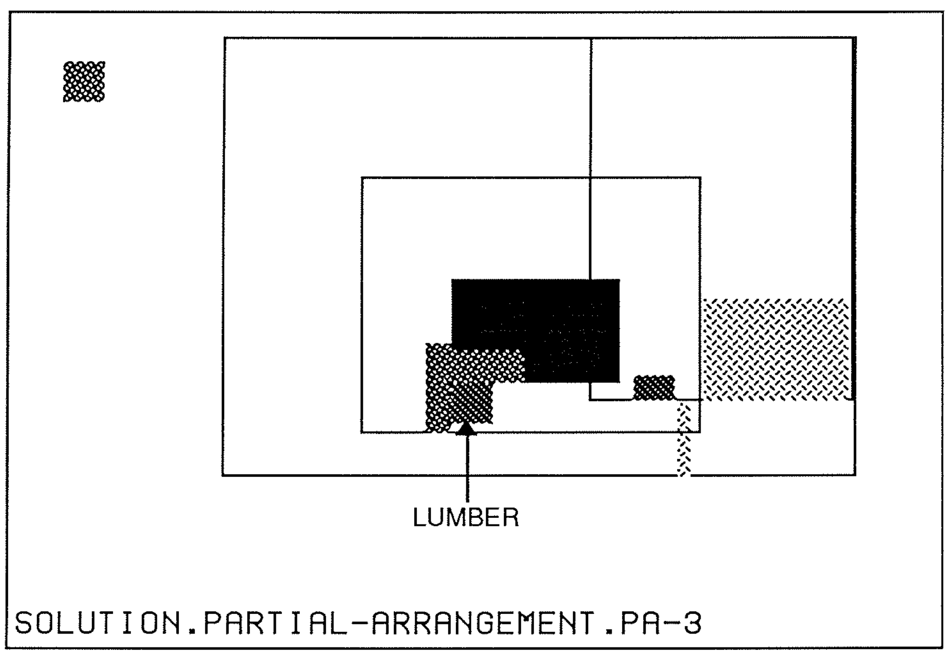


Figure 6-3b: PA3, Time Frame 871201–880430 including the LUMBER short-term laydown area.

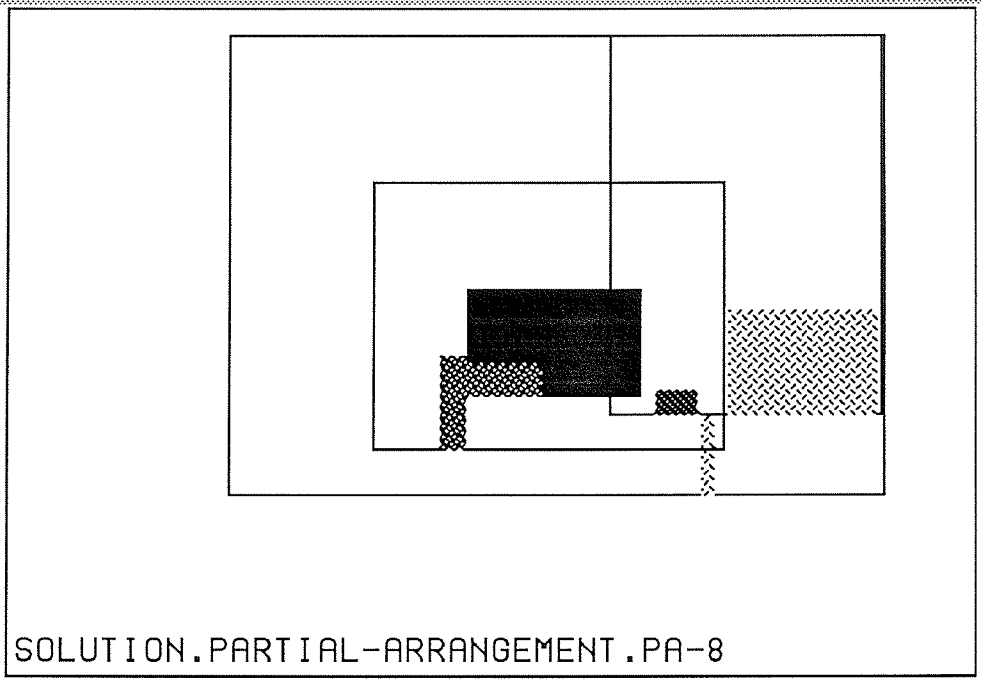


Figure 6-3c: PA8 with Time Frame 880430–880501. No short-term laydown areas are on site.

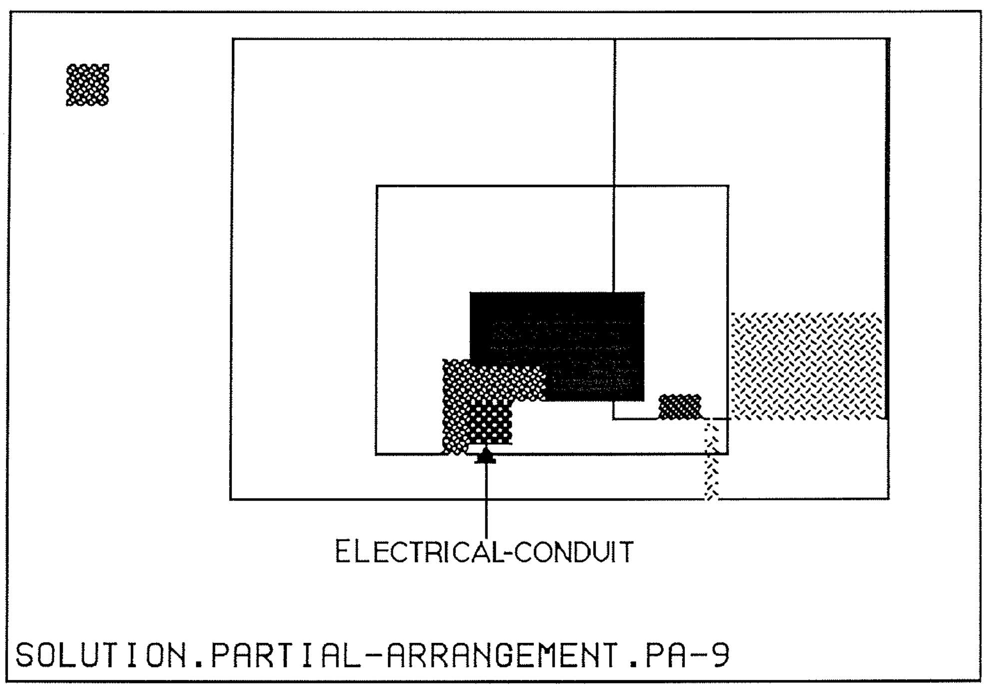


Figure 6-3d: PA9 with Time Frame 880501–880630 including the ELECTRICAL-CONDUIT short-term laydown area.

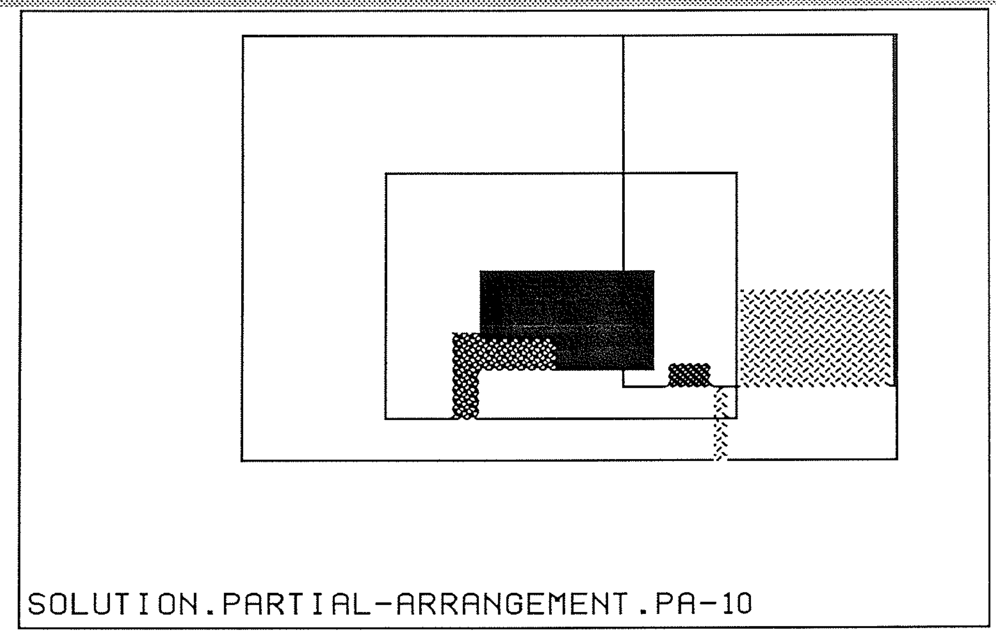


Figure 6-3e: PA10 with Time Frame 880630–880701. No short-term laydown areas are on site.

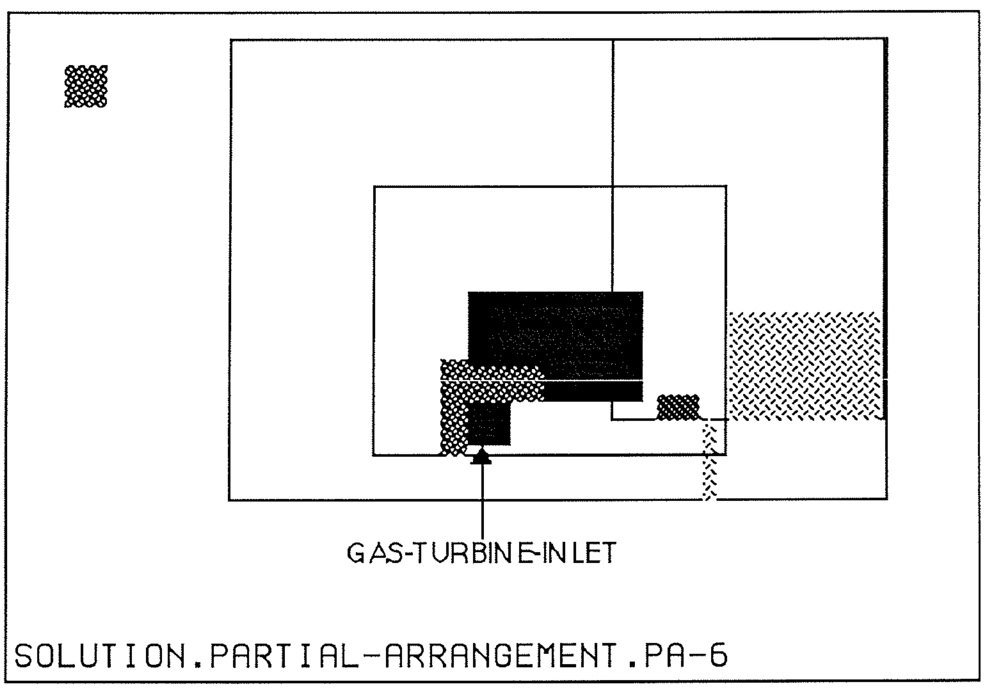


Figure 6-3f: PA6 with Time Frame 880701–881231, including the GAS-TURBINE-INLET short-term laydown area.

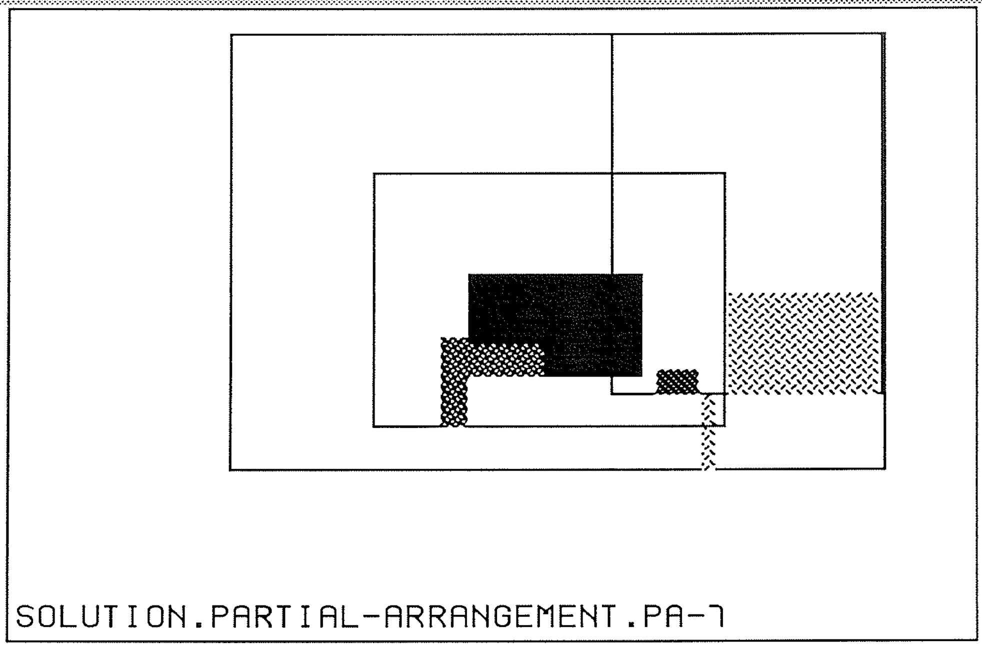


Figure 6-3g: PA7 with Time Frame 881231–890531. No short-term laydown areas are on site.

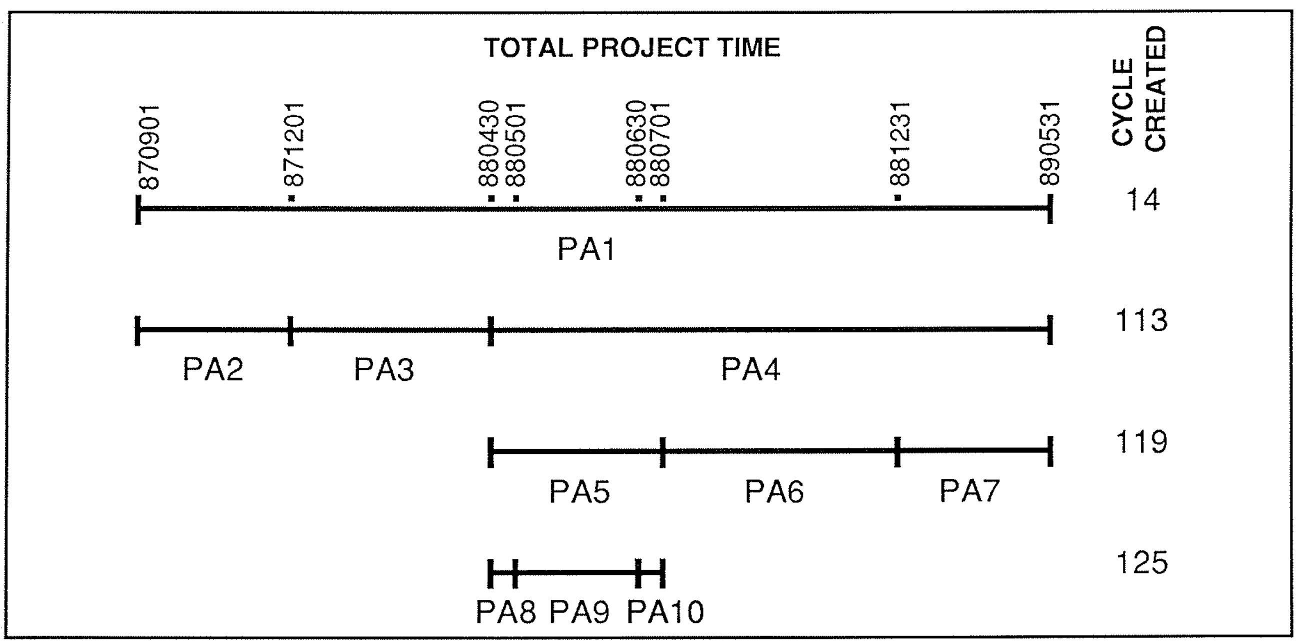


Figure 6-4: Partial Arrangements Laying Out the Site, Dividing Total Project Time into Time Intervals.

6.2.2. Discussion of the Temporal Strategy

SightPlan's Temporal Strategy demonstrates that the BB1 architecture and the constructive assembly method are suitable to accommodate not only spatial reasoning, but also spatial reasoning combined with temporal reasoning as needed for construction site layout. That is, when additional variables (such as time) are made explicit in the system, the system can be adapted to reason explicitly about them. SightPlan's model is therefore also promising for future work on the allocation of resources besides space and time. In the current system we have barely touched upon the temporal reasoning. Our future research will probe these issues.

6.2.3. Extensions to the Temporal Strategy

SightPlan's approach for temporal reasoning is a very simple one. There are many possible alternatives to it, many of which are explored in work on planning and scheduling. A few possible extensions of SightPlan's reasoning with respect to time and space are the following. One could allow objects to change positions from one PA to another by taking into account transition constraints. In a similar way, objects could be allowed to change area and shape over time. As permanent facilities only gradually materialize on site as their construction progresses (i.e., permanent facilities are not really on site at the moment construction commences), the space they occupy could be gradually included in the PAs. Finally, the total length of time that a facility is on site and the facility's size may not be the sole determinants of its criticality. When other factors are more important than time and space, they should be introduced in the model. We consider these challenging yet tractable areas for future research.

6.3. Computational Strategy

The third model of SightPlan is the basis for a comparative study between the Expert Strategy and the Computational Strategy. In crafting a strategy that is not restricted by human cognitive limitations, and in comparing it with a strategy that is restricted by such limitations, we hoped to gain insight about the type of cognitive extensions that computers can bring to their users for such problems.

This experiment uses the BB1 architecture and the application knowledge of IPP as a test bed. The only variables that differentiate the two strategies are the control knowledge sources of each of the models. Both models make use of the constructive assembly method to generate solutions. We modified the Computational Model to include constraints on objects and domain KSs, in addition to those that were used in the Expert Strategy model. The new constraints and KSs, however, could have been present in the Expert Model as well, assuming that the Expert Strategy had been specialized to derate the action they propose or are involved in, compared to the actions it chose before these additions were made. We were not concerned with the system's absolute execution time; thus, the additional cost incurred for having these constraints and KSs in the system is irrelevant. Also, in this comparative study, we do not account for the cost of rating, executing, and scheduling actions in this comparison, because these actions are part of the native BB1 architecture, and optimizing their efficiency should not be the task of an application developer.

Consider how one strategy could be better than another. First, one strategy could generate the same results as the other one, but in a more efficient manner, such as, in a shorter computation time. For example, SightPlan might improve its efficiency by performing control reasoning about constraints. The factors that could possibly improve efficiency, and that are under control of SightPlan, are when and how to call the constraint engine. The constraint engine's efficiency depends on the complexity of the input, and on the types of function calls that are needed to compute constraint satisfaction. The constraint engine is characterized in Section 6.3.1.1.

Second, one strategy could generate alternative solutions that were excluded by the other strategy. For example, SightPlan might deviate from the early commitment strategy, which was used in the Expert Model, and pursue a postponed commitment or a least commitment strategy. One advantage of this is that early commitment is not always capable of producing a solution in cases where postponed or least commitment might succeed; but even when early commitment succeeds, following this strategy results in only one solution to the problem whereas the other strategies may propose several alternatives. How SightPlan's Expert Strategy is reformulated to perform postponed commitment is described in Section 6.3.1, which gives the detailed formulation of the Computational Strategy.

6.3.1. Formulation of the Computational Strategy on IPP

We considered two separate ways to illustrate the potential of computers to overcome human information handling limitations for site layout: a least commitment computational strategy, and a postponed commitment computational strategy. They are described in this section.

6.3.1.1. Least Commitment

We initially crafted the Computational Strategy so that it would follow a least commitment strategy. A least commitment strategy is one in which commitments are postponed for as long as possible. The commitments that can be postponed are those related to satisfying preference constraints, so we removed the AS-CLOSE-AS-POSSIBLE constraints from the Expert Strategy. Upon further inspection of the Expert Strategy, it became obvious that the ZONING constraints express a kind of preference as well, so we removed them from the Expert Strategy.

We added in domain knowledge sources to perform the sampling of instances from sets of possible locations and to generate coherent instances (SAMPLE-POINTS, GENERATE-COHI). The SAMPLE-POINTS domain KS proved to be necessary because SightPlan might otherwise compute infinitely many combination layouts. This KS calls the application of a unary constraint that picks points out of the set of possible positions in the essential area of the object it applies to. For example, one way of picking is selecting each of the corners of the rectangles in the essential area. Coherent instances are layouts in which each object has a unique position, and in which no two objects overlap. They are similar to a solution layout, but they are generated from a layout with multiple positions for objects, by picking a unique position for each object.

Furthermore, we changed the Expert Strategy to reorder the ranking of constraints. This ranking was obtained based on the characteristics of SightPlan's constraint engine.

In this way, the problem becomes so underconstrained that the Computational Strategy limits itself to being a rough, brute-force approach, that almost blindly generates combinatorially many layouts. The execution of the least commitment strategy consists of the following steps:

1 Create a partial arrangement.
2 Include the context.
3 Include facilities with a fixed position.
4 Include objects to be positioned in the context.
5 Check for non-overlap between the objects to be positioned and the fixed facilities.
6 Sample positions from the very large sets of possible positions of objects.
7 Generate coherent instances.

Upon inspection of some of the solutions obtained, we recognized that least commitment with random sampling creates chaotic layouts. This shows the importance of using preference constraints to select among alternatives. Even though preference constraints do not necessarily have to be met (that is, they are not hard constraints, in the same way that physical or safety constraints are hard constraints), they help to restrict underconstrained layout problems. As a result, we maintain, people are tempted to add in constraints to the problem formulation which can be applied before or after solution generation, so that the resulting layouts will fewer and better organized.

6.3.1.2. Postponed commitment

In this way we arrived at a second idea: to craft the Computational Strategy so that it would follow a postponed commitment strategy. A postponed commitment strategy is one in which commitments are postponed until it is opportune to make them. That is, a postponed commitment strategy does not make decisions as early as an early commitment would, and not as late as a least commitment would; it strikes a balance between these two strategies.

The AS-CLOSE-AS-POSSIBLE preference constraints were again removed from the Expert Strategy, but the ZONING constraints were left in this time. From a computational standpoint, zoning constraints are very effective in that they drastically reduce the total area of an essential area, and in that they partition the layout space in smaller parts for which independent sets of coherent instances can be computed. The domain KSs and heuristics introduced for the Least Commitment remained. Following this new strategy, SightPlan is able to narrow the set of satisfying layouts, but it needs to resort to generating coherent instances to obtain a solution. These coherent instances are then shown to SightPlan's user, who can select the most preferred one.

This Computational Strategy was applied to the input provided by the AE on their site arrangement drawing; so it is an alternative to the Expert Strategy discussed above.

6.3.2. Implementation of the Computational Strategy

Table 6-3 charts the execution of the Computational Strategy. SightPlan creates a partial arrangement (Cycle 14), includes the context (Cycle 19), and includes the objects with fixed location in that context (Cycle 24). The system then identifies the occupied-space (Cycle 29), includes the sub-areas of the arrangement (Cycles 33-36), includes all the laydown areas (LAYDOWNs) in the arrangement (Cycle 40), and orients the partial arrangement (Cycle 44).

CYCLE	ACTION
14	create pa
19	include context
24	include fixed objects
29	include and identify occupied-space
33-36	include areas
40	include laydowns
44	orient pa
50-101	position laydowns non-overlapping with fixed objects
102-119	position within coal area
120-145	position within construction area
146-153	position within operations area
154-178	position outside of work area
182	add restriction constraints (run-time constraints)
183-234	restrict objects
239	compute coherent instances

Table 6-3: Some Cycles out of SightPlan's Computational Strategy Applied to IPP

Following these initial steps, the strategy prefers large objects and meets their constraints in the following order (numbers in parentheses are numeric weights for the matching constraints): non-overlap-set (0.98), non-overlap (0.98), zoned-in (0.90), zoned-outside-of (0.88), at-long-side

(0.85), adjacent-to (0.8), parallel (0.7), perpendicular (0.7), betw-short-sides (0.65), north-of (0.6), south-of (0.6), west-of (0.6), east-of (0.6), closer-than (0.5), further-than (0.5), discrete-sample (0.1), as-close--as-possible (0.1), pick-one (0.1). This order reflects that some constraints (such as non-overlap between temporary and permanent facilities) are hard, while others (such as zoning different areas) express only preferences. To some degree, this order contradicts the order suggested by computational efficiency of the constraint engine, because hard constraints are not always the most efficient ones to compute.

When all constraints on all objects are met, and objects remain with sets of possible locations, SightPlan introduces additional constraints into the problem to reduce those sets heuristically. Each set of possible locations will be restricted by sampling the corner points of the rectangles in their essential area by examples of the SAMPLE-FOUR-CORNERS constraint (Cycle 182).

Subsequently, SightPlan uses these limited locations for generating coherent instances (Cycles 183-234). SightPlan returns all coherent instances to the user; it considers them to be alternative solution arrangements (Cycle 239, Figures 65, 65 and 67).

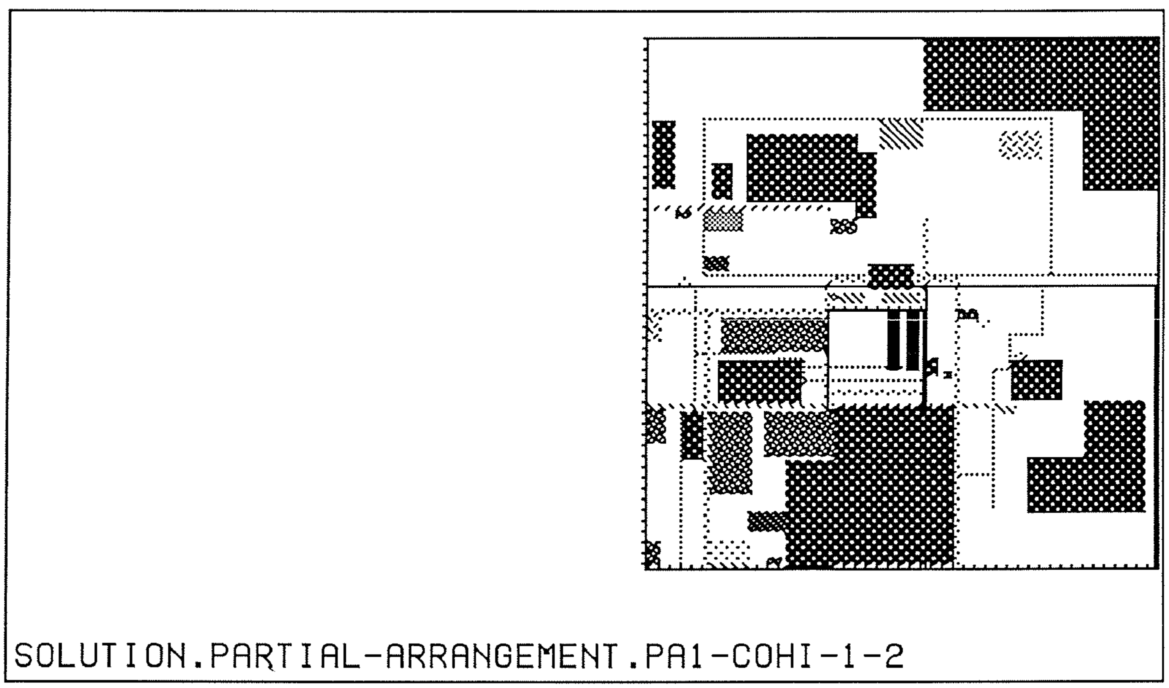


Figure 6-5: One Coherent Instance of a Solution Layout Generated by the Computational Model Applied to IPP

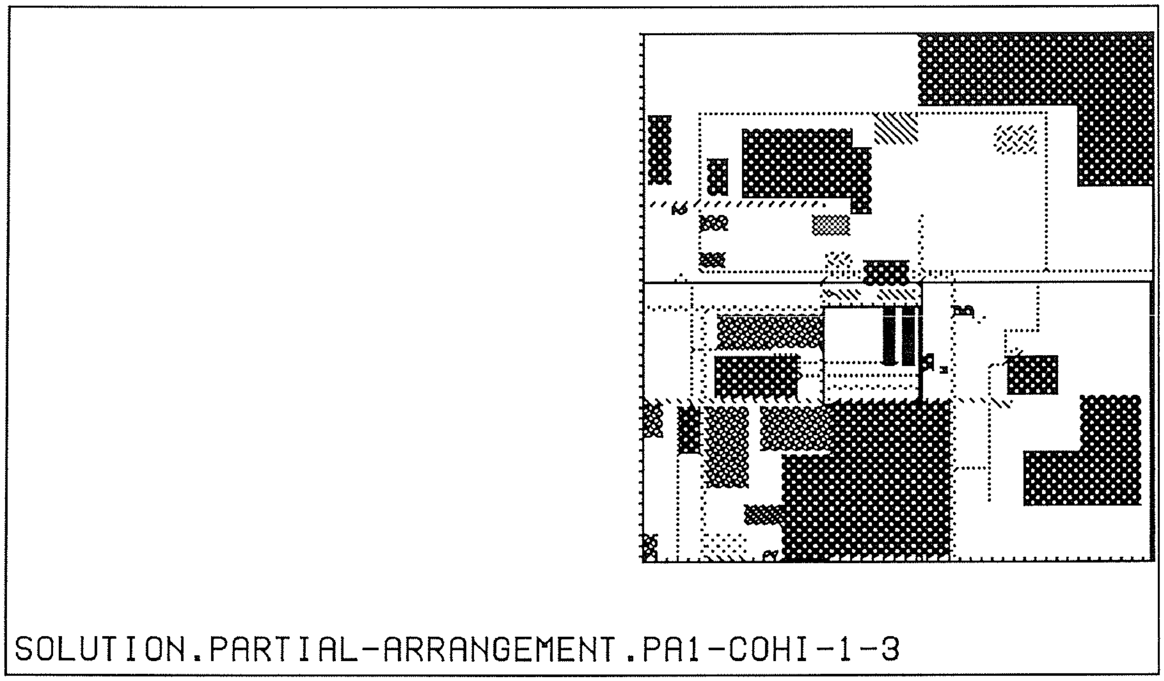


Figure 6-6: A Second Coherent Instance of a Solution Layout Generated by the Computational Model Applied to IPP

The system user, or SightPlan, can then evaluate those solutions and select the best one. For example, to evaluate the alternatives, SightPlan computes the sum of the distances between the centerpoints of each LAYDOWN and POWER-UNIT-1. COHI-1-2 rates 72,055, COHI-1-3 72,055, and COHI-1-4 74,687. These values provide a simplistic comparative measure of the relative quality of alternate layouts but fall far short of measuring their quality in any meaningful sense. As we stated at the outset, the difficulty of specifying a multi-attribute quality function for site layouts makes the problem hard to solve using conventional optimization techniques and motivated the AI approach to the problem used in our research. We discuss this again in section 6.4.2 below.

6.4. Comparison between the Expert and Computational Strategies

SightPlan applied different strategies to the IPP site: one of early commitment (implemented as the Expert Strategy and described in Section 6.1), one of least commitment (explored as an option in Section 6.3.1), and one of postponed commitment (implemented as the Computational Strategy and described in Sections 6.3.1 and 6.3.2). In the following paragraphs we discuss the advantages and disadvantages of each.

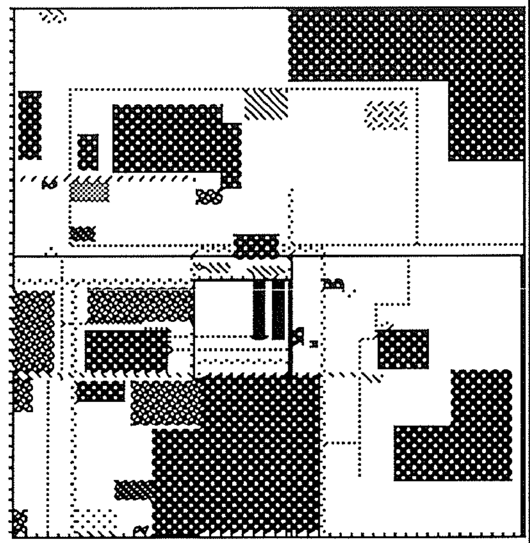


Figure 6-7: A Third Coherent Instance of a Solution Layout Generated by the Computational Model Applied to IPP

6.4.1. Early Commitment

The Expert Model demonstrated that an early commitment strategy can succeed for laying out construction sites. Yet, the discussion of this approach pointed out that such a strategy may not always result in a solution, even if one exists. We also pointed out that many alternative solutions are excluded by following this strategy, and we found that by providing an expert with alternatives, the expert could differentiate between good and better satisfying solutions providing us with additional knowledge for generating solutions.

6.4.2. Least Commitment

The least commitment strategy, which SightPlan applied as its preliminary Computational Strategy, models a brute force approach. The brute force approach has the advantage that it will produce a solution if one exists. In the case of the underconstrained IPP problem, SightPlan's strategy resulted in an almost infinite number of alternative arrangements. When faced with all these alternative arrangements, a person not satisfied with picking one at random, must apply some criterion of evaluation or discrimination in order to differentiate between them. As a result of this, not only is the cost of (almost exhaustively) generating all possible combinations of objects in a layout very high, the cost of differentiating between the results may be prohibitive as well. So, a system should balance the costs incurred for generating alternatives with the benefits of finding a better arrangement.

When we rationalized the Computational Strategy, we suggested that problems are often stated in highly underconstrained terms. People opportunistically add constraints before or during problem-solving in order tighten the problem specification and to narrow the set of potential solutions. To explore this concept, we crafted a postponed commitment strategy for SightPlan, that generates a reasonable subset of all possible solution layouts using additional preference constraints.

6.4.3. Postponed Commitment

The postponed commitment strategy, implemented in the final Computational Strategy, strikes a balance between heuristically pruning the solution space, and flexibly generating alternatives. SightPlan applies some user preference constraints, heuristically, samples sets of possible locations of objects, and generates a set of coherent instances. When constraining and sampling succeed in cutting out extraneous locations, then generating instances is fast, and a small number of solution layouts can be returned to the user for evaluation. This strategy may not find a solution, even if one exists, but its probability of success is higher than that of the early commitment strategy.

Further research on postponed commitment strategies may focus on:

- heuristic sampling of positions
- opportunistic generation of coherent instances
- qualitatively differentiating between coherent instances
- evaluating arrangements

6.5. User-Interactive Model

Our study in modeling how people lay out construction sites, and our suggestions on how to improve upon their layout strategy demonstrated to us that symbiotic human-machine systems have a good chance of providing better solutions than either one working alone.

The SightPlan study went through the following stages:

- Model field managers' practice
- Critique model
- Improve model

We could now experiment with a user-interactive model and explore the implications of such an improved model on managers' practice. SightPlan's strength is that it provides cognitive support to its users by making available memory capacity, computation power, display, and representation capabilities. Memory capacity makes it conceivable to build large knowledge bases, encompassing information about entire projects and covering projects' lifetimes. Computation power permits postponed commitment strategies. Display and representation capabilities allow the system to exchange information with other systems so that all data and strategic decisions are always readily available.

Implicit in our approach was the desire to explore whether it might be possible to build an interactive decision-support system that could permit a human expert using a computer to exploit the strengths of humans (in spatial visualization) and of machines (in data storage and retrieval, and in constraint bookkeeping) to plan for better designs than either of these could do alone. Therefore we named our system SightPlan: Sight -> to see and Plan -> to plan. Because SightPlan makes explicit every step it takes and can be interrupted, the user can intervene in the system's operation at any time; this is an essential prerequisite for effective collaboration.

6.5.1. Knowledge-Based Interactive Graphics

A computer screen with multiple colors is a natural representation of possible object locations on a 2-D site, thus graphics might be an especially powerful medium for human-machine communication in this domain. We concentrated on the more fundamental representation and reasoning issues in SightPlan and we exploited graphics for explanation and input in SightView.

6.5.2. Real-Time Interaction

The interactive graphical interface developed for this project combines three modalities of computing that provide significant leverage to one another:

- Blackboard expert system reasoning to choose which temporary facilities to position on the site and which constraints to apply to them; and
- A procedural constraint engine for processing constraints efficiently, and for updating the state of the solution, both on the SightPlan solution blackboard, and on the graphical display.
- Graphical display of solution status and input of user choices. A further level of user control over SightPlan implemented by SightView is to allow the user to choose between different possible layouts for aggregate objects displayed in competing partial arrangements.

Based upon our experiments with the system to date, and on Professor Levitt's experience with "GanttAlive", an interface of this type for interactive project scheduling in a knowledge processing environment [Levitt 87], the authors are confident that this type of knowledge-based interactive graphics has the potential to provide powerful new kinds of decision support for a range of engineering applications. We envision that a fleshed-out system like SightPlan could take over the role of the physical layout model that used to be the focus of attention for all parties involved in the design, construction, and operation of a facility.

7. Conclusions

7.1. Project Results

Our main objective for the SightPlan project was to explore AI architectures for solving the class of spatial arrangement problems to which site layout belongs. We found that by using carefully selected domain knowledge and a flexible reasoning mechanism we could gain significant advantages over generic and more rigid heuristic construction methods for layout design. The control strategy that emerged from modeling our selected site might be a first formalization of a layout strategy that could be taught to novices.

We conclude that the representation and reasoning schemes used in SightPlan are appropriate ones and are powerful enough—we can model the site layout process—but SightPlan needs a lot more knowledge, and significant efficiency enhancements if we want it to become a field-usable system.

ACCORD is domain-independent and potentially applicable to arrangement problems in a wide range of domains. The present work on SightPlan provides an excellent test of both of these hypotheses. We succeeded in utilizing both BB1 and ACCORD as a foundation for SightPlan. Indeed, the development of SightPlan was substantially facilitated by the availability of BB1/ACCORD and the layering of progressively more specialized software tools and knowledge bases.

7.2. Documentation and Dissemination of Project Results

This paper summarized how the BB1 representation and control architecture and the ACCORD positioning language were applied and adapted to the SightPlan problem. Future publications will describe the implementation of SightPlan in more detail for readers interested in replicating or extending the system.

We plan to publish a series of more detailed technical papers describing the knowledge representation and reasoning, and the comparison between the human and machine problem-solving strategies, that were implemented in SightPlan.

In addition to journal and conference papers, we have produced a videotape of the SightPlan system running on a networked TI Explorer™ and Macintosh II™ so that academic and professional colleagues can have the opportunity to see the system running and to develop a better sense of the opportunistic problem solving approach used in SightPlan. We are also planing to create and document a modular set of program code that can be run on a TI Explorer™ or microExplorer™ computer for distribution. Interested readers may contact the Center for Integrated Facility Engineering in Stanford's School of Engineering to obtain a copy of the videotape or program code.

7.3. Potential Extensions to the SightPlan System

Extensions to the SightPlan system itself might include extending the site and facility representation to three dimensions, increasing the flexibility of constraints beyond simple rectilinear measurements, and explicitly introducing temporal reasoning with transition states between project phases.

We have found this layered software architecture of BB1, ACCORD and SightPlan to be very effective, both cognitively and computationally, for the development, maintenance, and operation of SightPlan, and we plan to develop extensions in a similar layered fashion.

We have already implemented and validated a prototype system for project planning within the BB1 architecture. This system, which we call the Object-Action-Resource Planning System

(OARPLAN), develops construction project plans from CAD descriptions of permanent facilities to be constructed [Darwiche 89].

At the level of SightPlan, we are initiating the development of a system to perform architectural space layout of rooms within buildings, and are considering the development of a (yet unnamed) system for designing the layout of the permanent facilities on a power plant construction site. Reasoning in the power plant layout system could be integrated with SightPlan to optimize site layout more globally from both permanent operations and construction points of view—i.e., stored knowledge of site layout constraints would permit the permanent facilities to be relocated on the site during the design stage to permit a more optimal layout of facilities needed during construction of the plant.

The latter kind of project points the way to a kind of integration across project phases which is at the core of the mission of Stanford's Center for Integrated Facility Engineering (CIFE), a cooperative research center involving civil engineering and computer science faculty, in partnership with buyers, designers, builders and hardware/software tool developers in the architecture-engineering-construction (AEC) industry

8. Acknowledgements

Financial support from the *Stanford Construction Institute*, the *National Science Foundation* (Grant # MSM-86-13126), the *Center for Integrated Facilities Engineering at Stanford*, and the *U.S. Army Construction Engineering Research Laboratory* made this research possible. Although each of these organizations has endorsed the goals of the SightPlan research by their support, all of the opinions expressed in this report are those of the authors.

We are most grateful to an articulate and enthusisatic group of engineers and managers from Los Angeles Water and Power, Bechtel, Black & Veatch and Stone & Webster for their time and patience in challenging us with issues pertaining to site layout, in explaining layout approaches, and in granting permission to reproduce company documents and blueprints. Without their anecdotes and cases histories, SightPlan would not have achieved its goals. Our exploratory discussions were with Ed Ritchie, Bob Franklin, Ken Reinschmidt, Gavin Finn, Ron Wagner, Hank Pagliarulo and Bob Clay. Information on the **Intermountain Power Project** came from Walt Smalley, Rick Scott, Reed Nielsen, John Knight, Gary Rose, Burr McCorkle, and Jim Richter. Information on the **American 1** project came from George Spindle, Dean Bursheim, and Ray Sedey.

Profesors Bob Tatum, Boyd Paulson, Henry Parker, John Fondahl, and Clarkson Oglesby provided valuable input to the early prototype systems and ongoing critique of the work. Two MS students in Stanford's Construction Engineering and Management Program assisted in this research: Gail Tubbs, compiled an initial literature review of construction site layout methods, and Pat Steele helped to develop realistic layout examples.

In Computer Science, M. Vaughan Johnson enthusiastically helped Dr. Tommelein build the first SightPlan prototype; Micheal Hewett made many thoughtful suggestions for implementing "features and hacks"; former PROTEAN members Russ Altman, Jim Brinkley, Craig Cornelius, and Alan Garvey willingly discussed their representation and reasoning about arrangement problems, and Francois Daube assisted in designing and programming GS2D. And Diana Forsythe, an anthropologist associated with the Knowledge Systems Laboratory, provided assistance with some of the human interaction issues that arose during knowledge acquisition.

A final word of acknowledgement is in order to the chairs of the departments of civil engineering (Haresh Shah) and computer scence (Nils Nilsson), to the directors of CIFE (Paul Teicholz) and KSL (Edward Feigenbaum), and to Dean James Gibbons of the School of Engineering at Stanford. Their enthusistic support of multidisciplinary team approaches to engineering research problems created an environment in which the pursuit of this project was not only possible–it was honorable!

9. Bibliography

[Altman 87] Altman, R.B., Buchanan, B.G.: Partial Compilation of Strategic Knowledge, *Proceedings of AAAI 87*, pp. 399-404, 1987

[Andre 86] Andre, J-M.: CADOO, a Knowledge-Based System for Space Allocation, 2nd International Expert Systems Conference, London, pp. 399-409, 9 Sept-2 Oct 1986

[Andre 87] Andre, J-M.: CADOO, a Knowledge-Based System for Spatial Accommodation, Proceedings of ORIA'87: Artificial Intelligence and Sea, Marseille, France, 18-19 June 1987

[Armour 63] Armour, G.C., Buffa, E.S.: A Heuristic Algorithm and Simulation Approach to Relative Location of Facilities, *Management Science*, Vol. 9, pp. 294-309, 1963

[Baykan 87] Baykan, C.A., Fox, M.S.: An Investigation of Opportunistic Constraint Satisfaction in Space Planning, Proceedings of the Tenth International Joint Conference on Artificial Intelligence, IJCAI 87, Milan, Italy, pp. 1035-1038, 23-28 August 1987

[Bechtel 88] Bechtel: *3D Design System Users Manual, Overview of Bechtel 3D Design System* (Version 1.1), Bechtel Power Corp., Gaithersburg, MA, 1988

[Black&Veatch 80] Black & Veatch Consulting Engineers: *Intermountain Power Project, Intermountain Generating Station*, Kansas City, Missouri, 1980

System Analysis File 9255.42.2200, Construction Facilities and Services

System Design Specification File 9255.43.5400 (Roads and Parking, Railroads, Fencing and Security, and more)

[Brandeau 87] Brandeau, M.L., Chiu, S.S.: *An Overview of Representative Problems in Location Research*, Departments of Industrial Engineering and Engineering Management, and Engineering-Economic Systems, Stanford University, Stanford, CA, 64 pages, November 1987

[Brinkley 86] Brinkley, J., Cornelius, C., Altman, R., Hayes-Roth, B., Lichtarge, O., Duncan, B., Buchanan, B., Jardetzky, O.: *Application of Constraint Satisfaction Techniques to the Determination of Protein Tertiary Structure*, Working Paper No. KSL-86-28, 10 pages, March 1986

[Buchanan 86] Buchanan, B., et al: The Heuristic Refinement Method for Deriving Solution Structures of Proteins, Report No. KSL 85-41, 22 pages, March 1986

[Buchanan 87] Buchanan, B.G.: Artificial Intelligence as an Experimental Science, Stanford University, Report No. KSL-87-03, 41 pages

[Buffa 64] Buffa, E.S., Armour, G.C., Vollman, T.E.: Allocating Facilities with Craft, Harvard Business Review, Vol. 42, pp. 136-158, 1964

[Chandler 78] Chandler, I.E.: The Planning and Storage of Materials on Site, *Building Technology and Management*, pp. 14-16, October 1978

[Confrey 88] Confrey, T., Daube, F.: GS2D: A 2-D Geometry System, *Report No. KSL-88-15,* Knowledge Systems Laboratory, Computer Science Department, Stanford University, March 1988

[Darwiche 89] Darwiche, Adnan, Raymond E. Levitt and Barbara Hayes-Roth, "OARPLAN: Generating Project Plans by Reasoning about Objects, Actions and Resources," *Artificial Intelligence for Engineering Design, Analysis and Manufacturing,* Vol. 2, No. 3, pp 169-181, 1988,

[Dressel 63] Dressel, G.: *Arbeitstechnische Merkblätter für den Baubetrieb*, Forschungsgemeinschaft Bauen und Wohnen, IFA - Verlag Stuttgart, West Germany, 3. Auflage, 1970 (1st Edition 1963)

[Driscoll 86] Driscoll, J., Sangi, N.A.: The Development of Computer Aided Facilities Layout (C.A.F.L.) Systems - International Survey 1985-86 - Survey Report and Results, The University of Liverpool, U.K., 119 pages, December 1986

[Eastman 72] Eastman, C.M.: Preliminary Report on a System for General Space Planning, *Communications of the ACM*, Vol. 15, No. 2, pp. 76-87, February 1972

[Eastman 75] Eastman, C. M. (editor): *Spatial Synthesis in Computer-Aided Building Design*, Halsted Press, New York, 333 pages, 1975

[Engelmore 88] Engelmore, R., and Morgan, T. (editors): Blackboard Systems, Addison-Wesley, 1988

[Erman 80] Erman, L., Hayes-Roth, F., Lesser, V.R., Reddy, D.R.: The Hearsay-II speech understanding system: Integrating knowledge to solve uncertainty, ACM Computing Surveys, 12(2), pp. 213-253, 1980

[Flemming 89] Flemming, U., Coyne, R.F., Glavin, T., Hsi, H., Rychener, M.D.: A Generative Expert System for the Design of Building Layouts (Final Report), EDRC - Carnegie Mellon University, Pittsburgh, PA, 104 pages, 1989

[Francis 74] Francis, R.L., White, J.A.: Facility Layout and Location, Prentice-Hall, 468 pages, 1974

[FVB 84] Fonds voor Vakopleiding in de Bouwnijverheid: *Cursus Ploegbaas: Bouwplaatsorganizatie, 3. Bouwplaatsinplanting*, Brussels, Belgium, D/1984/1698/25, 31 pages, 1984

[Garvey 87a] Garvey, A., Hewett, M., Johnson, M.V.Jr., Schulman, R., Hayes-Roth, B.: *BB1 user Manual – Common Lisp version 2.0* , KSL 86-61, 110 pages, 7 August 1987

[Gates 78] Gates, M., Scarpa, A.: Optimum Location of Construction Haul Roads, ASCE, Journal of the Construction Division, Vol. 104, No. CO4, pp. 395-407, December 1978

[Grant 82] Grant, D.P.: Space Planning Methods, Design Methods and Theories, Vol. 17, No. 1, pp. 4-36, 1982

[Hamiani 87] Hamiani, A.: CONSITE: A Knowledge-Based Expert System Framework for Construction Site Layout, PhD Dissertation, Department of Civil Engineering, The University of Texas at Austin, 180 pages, December 1987

[Hamiani 88] Hamiani, A.: CONSITE: A Knowledge-Based Expert System for Site Layout, ASCE Proceedings of the 5th Conference, Computing in Civil Engineering: Microcomputers to Supercomputers, Alexandria, Virginia , March 29-31, 1988

[Handa 87] Handa, V., Lang, B.: Site Planning Elements, University of Waterloo, Construction Management Group and Construction Safety Association of Ontario, Canada, 72 pages, 1987

[Handa 88] Handa, V., Lang, B.: Construction Site Planning, Construction Canada 88 05, pp. 43-49, 1988

[Handa 89] Handa, V., Lang, B.: Construction Site Efficiency, Construction Canada 89 01, pp. 40-48, 1989

[Hayes-Roth 79] Hayes-Roth, B., Hayes-Roth, F., Rosenschein, S., and Cammarate, S. Modeling planning as an incremental, opportunistic process, Proceedings of the Sixth International Joint Conference on Artificial Intelligence, 1979

[Hayes-RothB 85a] Hayes-Roth, B.: A Blackboard Architecture for Control, *Artificial Intelligence*, Vol. No. 26, pp. 251-321, 1985

[Hayes-RothB 85b] Hayes-Roth, B. and PROTEAN Project Members: *Elucidating Protein Structure from Constraints in PROTEAN*, Technical Report KSL-85-35, 1985

[Hayes-RothB 85c] Hayes-Roth, B.: *Assembling Arrangements of Objects under Constraints*, draft, 44 pages, October 1985

[Hayes-RothB 86] Hayes-Roth, B., et al.: PROTEAN: Deriving Protein Structures from Constraints, *Proceedings of AAAI 86*, 1986

[Hayes-RothB 87] Hayes-Roth, B., Garvey, A., Johnson, M.V.Jr., Hewett, M.: *A Modular and Layered Environment for Reasoning about Action*, Report No. KSL 86-38, 63 pages, April 1987

[Hayes-RothB 88a] Hayes-Roth, B., Hewett, M.: BB1: An Implementation of the Blackboard Control Architecture, in Engelmore, R., Morgan, T.: *Blackboard Systems*, Addison-Wesley, pp. 297-313, 1988

[Henderson 76] Henderson, E.M.: The Use of Scale Models in Construction Management, Technical Report No. 213, distributed by The Construction Institute, Department of Civil Engineering, Stanford University, 105 pages, November 1976

[Hewett 88a] Hewett, M.: *BB1 User Manual – v2.1 Update (Common LISP)*, Report KSL 86-61a, Stanford, CA, 17 pages, 25 February 1988

[Hewett 88b] Hewett, M.: *Communication Interface User Manual and Using the Communication Interface with BB1*, Report No. KSL 88-xx, two times 9 pages, August 1988

[Hewett 88c] Hewett, M.: *Appendix to BB1 v2-1 Manual*, Report KSL 86-61a Appendix, 5 pages, November 1988

[Hollnagel 86] Hollnagel, E., Mancini, G., Woods, D.D. (editors): Intelligent Decision Support in Process Environments, Springer-Verlag, NATO ASI Series, 521 pages, 1986

[JohnsonMV 87] Johnson, M.V.Jr., Hayes-Roth, B.: Integrating Diverse Reasoning Methods in the BB1 Blackboard Control Architecture, *Proceedings AAAI 87*, pp. 30-35

[Lee 67] Lee, R.C., Moore, J.M.: CORELAP, Journal of Industrial Engineering, Vol. 18, pp. 195-200, 1967

[LevittR 85] Levitt, R.E., Kunz, J.C.: Using Knowledge of Construction and Project Management for Automated Schedule Updating, *Project Management Journal*, Vol. 16, No. 5, December 1985

[LevittR 87] Levitt, R.E., Kunz, J.C.: Using Artificial Intelligence Techniques to Support Project Management, *Journal of Artificial Intelligence in Engineering, Design, Analysis and Manufacturing* (AI EDAM), Vol. 1, No. 1, pp. 3-24, 1987

[MillerG 56] Miller, G.A.: The Magical Number Seven, Plus or Minus Two: Some Limits on our Capacity for Processing Information, *Psychological Review*, Vol. 63, No. 2, pp. 81-97, March 1956

[Moore 71] Moore, J.M.: Computer Program Evaluates Plant Layout Alternatives, *Industrial Engineering*, Vol. 3, No. 8, pp. 19-25, August 1971

[Moore 74] Moore, J.M.: Computer Aided Facilities Design: An International Survey, *International Journal Prod. Res.*, Vol. 12, No. 1, pp. 21-44, 1974

[Moore 80] Moore J.: Computer Methods in Facility Layout, *Industrial Engineering*, Vol. 19, pp. 82-93, September 1980

[Muther 61] Muther, R.: *Systematic Layout Planning*, Industrial Education Institute, Boston, Massachusetts, 1961

[Muther 70] Muther, R., McPherson, K.: Four Approaches to Computerized Layout Planning, *Industrial Engineering*, Vol. 2, pp. 39-42, February 1970

[Neil 80] Neil, J.M.: Teaching Site Layout for Construction, ASCE Meeting, Portland, OR, April 14-18, pp.1-11, 1980

[Neil 82] Neil, J.M.: Steam-Electric Generating Station Construction, M-K Power Group, section on Construction Site Layout, pp.7-11 to 7-29, June 1982

[Nii 82] Nii, H.P., Feigenbaum, E.A., Anton, J.J., and Rockmore, A.J. Signal-to-symbol transformation: HASP/SIAP case study. AI Magazine, 3, pp. 23-35, 1982

[Paulson 72] Paulson, B.C.Jr.: Man-Computer Concepts for Planning and Scheduling, *ASCE Journal of the Construction Division*, Vol. 98, No. CO2, pp. 275-286, September 1972

[Pfeffercorn 75] Pfeffercorn, C.E.: A Heuristic Problem-Solving Design System for Furniture Layouts, Communications of the ACM, Vol. 18, No. 5, pp. 286-297, 1975

[Popescu 78] Popescu, C.: Large Construction Site Design and Organization in Developing Countries, SAE paper No. 780533, Society of Automotive Engineers (SAE), 1978

[Popescu 80a] Popescu, C.: Temporary Facilities—Utilities Designing Steps, ASCE Convention and Exposition, Portland, OR, 20 pages, 14-18 April 1980

[Popescu 80b] Popescu, C.: Construction Site Population: Estimating the Size and Structure, Proceedings of the 8th CIB Congress, Oslo, Norway, 1980

[Popescu 81] Popescu, C.: Managing Temporary Facilities for Large Projects, Proceedings of the Project Management Institute and INTERNET Joint Symposium, Boston, MA, pp. 170-173, 28-30 September 1981

[Rad 82] Rad, P.F.: A Graphic Approach to Construction Job-Site Planning, Cost Engineering, Vol. 24, No. 4, pp. 211-217, August 1982

[Rad 83] Rad, P.F., James, B.M.: The Layout of Temporary Construction Facilities, Cost Engineering, Vol. 25, No. 2, pp. 19-27, April 1983

[Reinschmidt 75] personal communication during my visit at the Stone & Webster offices in Boston, Massachusetts, December 1986

[Rodriguez-Ramos 82] Rodriguez-Ramos, W.E.: Quantitative Techniques for Construction Site Layout Planning, Ph.D. Dissertation, University of Florida, Gainesville, FL, 1982

[Rodriguez-Ramos 83] Rodriguez-Ramos, W.E., Francis, R.L.: Single Crane Location Optimization, ASCE Journal of Construction Engineering and Management, Vol. 109, No. 4, pp. 387-397, December 1983

[Rodriguez-Ramos 84] Rodriguez-Ramos, W.E.: On Construction Layout Modeling and Microcomputer Heuristics, Proceedings of the 2nd National Conference on Microcomputers in Civil Engineering, Orlando, Florida, October 30-31, November 1, 1984

[Sowa 84] Sowa, J.F.: Conceptual Structures: Information Processing in Mind and Machine, Addison-Wesley, 481 pages, 1984

[Stark 83] Stark, R.M., Maher, R.H.: *Quantitative Construction Management: Uses of Linear Optimization*, John Wiley & Sons, 159 pages, 1983

[Stone&Webster 78] Stone & Webster Engineering Corporation: *Reference Fossil Power Plant Models*, 18 pages, Boston, Massachusetts, 1978

[Stone&Webster 79] Stone & Webster Engineering Corporation: *Construction Field Manual—Construction Facilities Guidelines, Volume I*, Construction Department, CFG 2.2, Rev. 0, 8 pages, Boston, Massachusetts, 23 March 1979

[Tatum 81] Tatum, C.B., Harris, J.A.: Construction Plant Requirements for Nuclear Sites, *ASCE Journal of the Construction Division*, Vol. 107, No, CO4, December 1981

[Tommelein 87a] Tommelein, I.D., Johnson, Hayes-Roth, B., M.V., Jr. Levitt, R.E.: SIGHTPLAN: A Blackboard Expert Systems for Construction Site Layout, in J.S. Gero (ed.), Expert Systems in Computer-Aided Design, North Holland, Amsterdam, pp. 153-167, February 1987

[Tommelein 87b] Tommelein, I.D., Levitt, R.E., Hayes-Roth, B.: Using Expert Systems for the Layout of Temporary Facilities on Construction Sites, in Lansley, P.R., Harlow, P.A. (eds.), Managing Construction Worldwide Volume One: Systems for Managing Construction, Spon, London, pp. 566-577, September 1987

[Tommelein 89a] Tommelein, I.D., Levitt, R.E., Hayes-Roth, B.: SightPlan: An Artificial Intelligence Tool to Assist Construction Managers with Site Layout, 6th International Symposium on Automation and Robotocs in Construction, San Francisco, pp. 340-347, 6-8 June 1989

[Tommelein 89b] Tommelein, I.D.: Comparing Design Strategies of Agents with Limited Resources, MSAI Thesis, Department of Computer Science, Stanford University, Stanford, CA, June 89

[Tommelein 89c] Tommelein, I.D.: SightPlan—An Expert System That Models and Augments Human Decision-Making for Desinging Construction Site Layouts, PhD Dissertation, Department of Civil Engineering, Stanford University, Stanford, CA, September 89

[Tubbs 85] Tubbs, G.: *Research Methodologies for Laying out Temporary Facilities in Construction*, Stanford University, Internal Research Report, Department of Civil Engineering, 5 pages, December 1985

[Van Hattum 72] Van Hattum en Blankevoort: *Werkterreinindelingen*, 10 pages, Beverwijk, the Netherlands, 1972

[Vollman 66] Vollman, T.E., Buffa, E.S.: The Facilities Layout Problem in Perspective, *Management Science*, Vol. 12, No. 10, January 1966

[Warszawski 73a] Warszawski, A., Peer, S.: Optimizing the Location of Facilities on a Building Site, *Operational Research Quarterly*, Vol. 24, No. 1, pp.35-44, 1973

[Warszawski 73b] Warszawski, A.: Multi-Dimensional Location Problems, *Operational Research Quarterly*, Vol. 24, No. 2, pp.165-179, 1973

[Warszawski 73c] Warszawski, A.: Analysis of Transportation Methods in Construction, *ASCE Journal of the Construction Division*, Vol. 99, No. CO1, pp. 191-202, July 1973

[Warszawski 87] Warszawski, A., Peled, N.: An Expert System for Crane Selection and Location, pp. 313-324, in Paciuk, M. (editor), Publications 1987 - The Building Research Station, Technion - Israel Institute of Technology, Faculty of Civil Engineering, Technion Research and Development Foundation, ltd., 397 pages, 1987

[Weidemier 86] Weidemier, J.: Layout of Power Station Construction Sites, ESAA Conference, Section 1–Generation, Port Augusta, Australia, pp. 6B.1 to 6B.9, September 1986

[Woods 86a] Woods, D.D.: Paradigms for Intelligent Decision Support, pp. 153-173, in [Hollnagel 86]

[Woods 86b] Woods, D.D.: Cognitive Technologies: The Design of Joint Human-Machine Cognitive Systems, AI Magazine, Vol. 6, No. 4, pp. 86-92, Winter 1986

[Yessios 71] Yessios, C.: FOSPLAN: A Formal Space Planning Language, Institute of Physical Planning, School of Urban and Public Affairs, Carnegie-Mellon University, Research Report No. 27, November 1971