SightPlan

AN EXPERT SYSTEM THAT MODELS AND AUGMENTS HUMAN DECISION-MAKING FOR DESIGNING CONSTRUCTION SITE LAYOUTS

A DISSERTATION

SUBMITTED TO THE DEPARTMENT OF CIVIL ENGINEERING

AND THE COMMITTEE ON GRADUATE STUDIES

OF STANFORD UNIVERSITY

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS

FOR THE DEGREE OF

DOCTOR OF PHILOSOPHY

By

Iris Denise Tommelein

August 1989

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 dissertation 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 on-going 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.

Acknowledgements

It has been exciting to compile this thesis over several years in the fascinating and cosmopolitan Stanford environment, working with people full of ideas and enthusiasm.

I owe most thanks, recognition, and respect to my two main advisors, Ray Levitt and Barbara Hayes-Roth. Ray's unconditional support and Barbara's brilliant ideas, together with their intellectual challenges and encouragements provided the cornerstones of my work. Boyd Paulson and C. Bob Tatum gave insightful feedback at opportune times, and I much appreciate the thorough review and editing of this dissertation by all four of them, as well as by Evelyn Gilbert and Karen Hansen.

I am most grateful to the following engineers, managers, and professors for their time and patience in challenging me 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 been the same. My exploratory discussions were with Ed Ritchie, Bob Franklin, Ken Reinschmidt, Gavin Finn, Ron Wagner, Hank Pagliarulo, Bob Clay, Hank Parker, and Clark Oglesby. 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. Appendix A lists the titles of the people from industry and the companies with which they are associated.

The Stanford Construction Institute, the National Science Foundation (Grant #MSM-86-13126), and the Center for Integrated Facilities Engineering at Stanford made this research possible by providing funding.

The department of Civil Engineering and the Knowledge Systems Laboratory both welcomed me as one of their members. I would like to thank their staff for providing a supportive environment to work in. Special thanks goes to Ms. Gelatt for exchanging ideas about things besides school.

In Civil Engineering I appreciated the assistance of Gail Tubbs, who compiled an initial literature review, and Pat Steele, who helped me develop realistic layout examples.

In Computer Science, M. Vaughan Johnson enthusiastically helped me build my 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. Tony Confrey increased the value of SightPlan tremendously by implementing the SightView user interface and, together with François Daube, programming GS2D. Diana Forsythe made me aware of the human interaction issues during knowledge acquisition.

My friends made Stanford a fun place to live; they also make it hard to leave. Among others, Rob van der Wijngaart, Ashish Karamchandani, Raimund Lammersdorf, Maria Koundoura, Alain Prelis, Karen Hansen, Glenn Katz, Nabil Kartam, Andy Gans, and Adnan Darwiche spent many hours with me talking about AI, philosophy, and life in general. Of course, my Stanford "home," Breakers Eating Club, is the best place of all for food, laughter, and intellectual discussion; and sources of support, well beyond Stanford, are my family, and my friend, Jim Lovekin.

To Marijke Mollaert,

who piqued my interest in research.

Table of Contents

Figures

Tables

Chapter 1

Introduction

1.1 Research Motivation

The SightPlan project addresses the question of whether or not *artificial intelligence* (*AI*) programming techniques can contribute to layout modeling, specifically, of temporary facilities on construction sites. A review of literature on layout design and discussions with construction managers showed that a large discrepancy exists between the procedures field practitioners use to lay out construction sites and the solution procedures represented in mathematical models, which are seldom used for this purpose. Presumably, a tool which contains the site data accessed by managers, and which models step-by-step the way that experienced managers lay out a site, would have a better chance of being used in the field. Thus, I embarked on "knowledge engineering" SightPlan, a blackboard expert system, to capture site data needed for determining temporary facilities and to model the layout design methods of construction field managers.

Our interest in SightPlan was also sparked by the cognitive-science viewpoint of researchers in AI, who are less concerned with the solution of the design problem per se than they are with understanding the design process itself. Their questions are: *How do people design?* and *Why do they design that way?* After completing the first SightPlan model, I studied its performance by gauging the system against several qualitative criteria, and I tried to identify features of the implementation that could be improved. The architecture of the implementation, which separates strategic problem-solving knowledge from knowledge about the domain, provided the opportunity to experiment with an alternate solution strategy and to incorporate graphics to make the system interactive. Thus, SightPlan became an expert system that models human decision-making for designing construction site layouts, and that augments human decision-making with computational and graphical support tools.

1.2 Objectives and Scope

The research had four major objectives; they are listed below in decreasing order of importance.

1. **INVESTIGATE WHAT KIND OF KNOWLEDGE MANAGERS USE FOR LAYING OUT CONSTRUCTION SITES IN ORDER TO *ARTICULATE THIS CONSTRUCTION FIELD PRACTICE.***

The concepts of *site layout* and *temporary facilities* are meaningful to construction site managers. Everyone has a reasonable understanding of what these words signify and can explain the basic concepts. Yet, when asked how one lays out temporary facilities on a site, managers found it difficult to articulate the process. Several questions came to mind. Is site layout so involved and complex that one does not know where to begin explaining? Is it too simple to talk about? Is knowing how to lay out sites a trade or company secret? Are rules-of-thumb used and guesses made to determine how a site should be laid out? Whatever the method or knowledge is that field managers use to lay out sites, it is important that it be articulated. In this way, people can communicate, learn from, improve on, and possibly formalize field practice, and they can develop an understanding that may facilitate the layout task.

2. ***BUILD A MODEL* THAT WOULD MIMIC THE ACTIONS PEOPLE TAKE FOR DESIGNING SITE LAYOUTS.**

What could be learned from written information and from field data? How does one generate a *good* layout? How does one model the layout process? We were interested in modeling the strategies of people experienced in laying out sites, and we intended to develop a computer model that would generate layouts similar to the ones these people obtain. This model would then be available for inspection and for experimentation [Buchanan 87].

Our analysis would constitute a comparison between the model and human practice. This would allow us to speculate on the reasons why our model might perform better or worse than people. By developing an articulated representation that we could reproduce, test, and validate, we hoped to acquire insight into people's practices as well as to improve the model itself.

3 *TEST BB1*, A GENERIC BLACKBOARD ARCHITECTURE, ON A REALISTIC DESIGN PROBLEM.

The blackboard architecture is an expert system technology for representing multiple, modular knowledge sources and using them in a coordinated way to reason about action. Moreover, this architecture promises to be a foundation for integrated engineering environments. Implementing SightPlan and assessing the representation power provided by BB1 would show the usefulness of the system for modeling this and other design tasks.

4 FIND OUT WHAT *TYPE OF INTERACTION* MIGHT EXIST BETWEEN A COMPUTER MODEL LIKE SIGHTPLAN AND A PERSON LAYING OUT SITES.

How good could a model like SightPlan possibly be? Would it behave like an implementation of a black-box algorithm to which data is fed and from which results are returned? Could SightPlan be made "intelligent" enough so that it would operate alone, or would user interaction be needed to obtain desired performance? Would SightPlan cooperate with other systems to solve its site layout problem? Articulating a model and testing how field managers perceive it would be a worthwhile investigation.

To meet the above objectives, we established the following scope for SightPlan:

1 BECAUSE LAYING OUT SITES IS COMPLEX, SIGHTPLAN RESTRICTS ITSELF TO LAYING OUT ONLY TEMPORARY FACILITIES THAT ARE ON SITE FOR A RELATIVELY LONG TIME.

These include, for example, long-term laydown areas and construction support buildings; they exclude short-term work areas and construction equipment.

2 WE DID NOT ASPIRE TO BUILD SIGHTPLAN TO OUTPERFORM HUMAN EXPERTS, SO IT IS ACCEPTABLE TO REPRESENT ONLY A SIMPLE MODEL.

We wished to understand how people design layouts and represent the type of knowledge they use for that task. SightPlan models field practice, learned from two case studies. SightPlan's main model is tailored after the first case and is validated through the second case.

We preferred real-world case studies over hand-crafted examples because we wanted to learn from field practitioners and they often introduce unanticipated factors.

A disadvantage of case studies is that project identification and field data collection take time. We limited the number for our study to two, so we cannot claim generality of the knowledge represented in SightPlan. SightPlan is an early attempt to formalize the layout process and to deliver a proof of concept showing that the representation scheme is expressive and useful; in time, more knowledge can flesh out the concepts presented in the model.

3 THE ARCHITECTURE SUCCEEDS FOR OUR PURPOSES IF IT IMPLEMENTS SIGHTPLAN'S INPUT DATA AND REASONING IN A COMPREHENSIVE WAY.

We neither engaged in a formal evaluation of the BB1 architecture, nor assessed its utility compared to that of other blackboard architectures or other knowledge representation schemes. We used BB1 because of its potential for representing SightPlan's knowledge and because it was readily available. Since SightPlan is a research tool, we are not concerned with execution speed, but rather with the ease of using the implementation environment.

4 SIGHTPLAN'S MERITS AND SHORTCOMINGS WOULD BE QUALITATIVELY ASSESSED.

No formal assessment would be made of how the model is perceived by people using it. Rather, we would speculate on possible modes of operation of the model, and suggest improvements to be made for realizing them.

1.3 Methodology

This dissertation describes an experimental study of how field practitioners lay out construction sites. It used the following methodological steps:

1 DEFINE AND REDEFINE PROBLEM

Defining what site layout entails is not easy. I started out with a definition for site layout, but revised that definition to include or remove factors as I learned more about the complexity of the problem itself and about the modeling capabilities available to me.

2 REVIEW EXISTING MODELS

I reviewed the literature related to theoretical models for site layout and made several visits to construction companies in order to understand how the problem was viewed from the two perspectives of theory and practice.

3 SELECT MODEL

For pragmatic reasons, I decided to use the BB1 blackboard architecture as a development tool.

4 COLLECT DATA FROM LITERATURE

Once the type of model was formulated and the environment in which to implement it was selected, I collected information from published sources.

5 DEVELOP PROTOTYPE

I then implemented a simple problem to explore the capabilities of the model.

6 SELECT PROJECT

Having decided to obtain data from a particular site, I investigated what projects were available that met our criteria, and selected one for detailed study.

7 ACQUIRE FIELD KNOWLEDGE

I made a field trip to the selected project site and visited several home offices to acquire project-specific data.

8 DEVELOP SYSTEM

Using the field data, I designed and implemented the SightPlan Expert Strategy.

9 VALIDATE MODEL INTERNALLY

I evaluated the model informally by testing a few alternate strategies and constraints on objects. Then I asked one of the field managers on the project to inspect and comment on the model.

10 SELECT A SECOND PROJECT

To validate the first strategy, I selected a second project to be modeled.

11 VALIDATE MODEL AND EXPERIMENT

I tested and extended the Expert Strategy, and subsequently developed an alternate strategy for a comparative study.

12 REPORT ON CONCLUSIONS, RATIONALIZE, AND PROPOSE FURTHER EXTENSIONS

Many interesting conclusions were drawn from the development of the SightPlan model and from the experiments conducted with it.

1.4 Reader's Guide

This dissertation is organized to enable the reader interested only in construction site layout (the *layout reader*), or only in AI and design (the *AI reader*), to focus on the relevant sections.

Chapter 2 first defines "site layout," then sketches how field practitioners lay out construction sites. Section 2.3 on models and methods reviews visual aids and computational tools that can help conceptualize and solve a layout problem. The literature referenced in this chapter is arranged by degree of generality and by degree of formality in Section 2.4. The gap in layout research, identified in this arrangement of the literature, is proposed as the target of work on SightPlan. Section 2.5 provides evidence that the knowledge SightPlan is to rely on can indeed be articulated. The chapter concludes by pointing at the research direction SightPlan will take. The layout reader will enjoy the line of reasoning followed in Chapter 2, which supports the usefulness of applying AI techniques to site layout modeling. Section 2.3.2 on computational models is a more theoretical discussion, and is not essential to the remainder of the dissertation, so it can be skipped. The AI reader can just skim this chapter to learn about the layout problem.

Chapter 3 reviews the AI literature on design and spatial reasoning, and may be ignored by the reader interested in construction site layout only.

Chapter 4 describes in careful detail all the elements comprising the SightPlan system. Unless the reader wants to know the implementation details, he or she may thumb through this chapter on a first reading, and return to it later.

Chapter 5 introduces the two power-plant layouts, Intermountain and American 1, that SightPlan models.

Chapter 6 constitutes the main report on the SightPlan research. It elaborates on alternate SightPlan strategies and their application to the data provided by the case studies. Section 6.1 describes the Expert Strategy applied to Intermountain. Section 6.2 validates that strategy by applying it to American 1. Section 6.3 describes the Computational Strategy that was crafted to take advantage of the power provided by the computer and compares it with the Expert Strategy. Every section examines one experiment and ends with a detailed discussion of the findings.

Chapter 7 concludes the dissertation, first by stating the contributions to knowledge of the SightPlan work; second by contrasting the successes and shortcomings of SightPlan; third by drawing the implications from this work; and last by suggesting directions for future research.

Chapter 2

Modeling Site Layout

Site layout needs to be addressed routinely by managers on construction sites. Though it is generally acknowledged that an efficient overall layout plays a key role in the operational efficiency, timeliness, cost, and quality of construction, project managers usually learn to accomplish this task only by trial and error in the course of years of field work. There is not a single well-defined method that can guarantee solving the problem and that can be taught. As Neil states in his guidelines and recommendations for approaching a site layout problem:

> "There is no quick and easy solution because of the many variables that make each project unique. However, there are several basic principles, many considerations, and some criteria which, if applied with good judgement, can direct planners towards a site layout solution which will be ultimately far superior to one that is allowed to evolve with the construction activity." [Neil 82]

This chapter begins by defining what the task of laying out a construction site entails. What are temporary facilities? What factors influence a site layout, and who is responsible for the layout? What objectives are to be met? What difficulties are encountered in defining and solving the problem? The chapter then describes the field practices of managers who lay out sites, examines the visual aids and computational support tools they have at their disposal, and reviews the small body of literature that pertains specifically to site layout. A chart depicts the existing approaches to site layout in two dimensions—generality and degree of automation—and points out a gap in the spectrum of modeling practice. The new model, named SightPlan, is an attempt to fill that gap. The kind of information used by field practitioners that SightPlan can apply to generate site layouts is described next. The chapter concludes with the state of the art in construction site layout and with a discussion of the research direction pursued by SightPlan.

2.1 Problem Formulation

Site layout consists of 1) identifying the facilities that are temporarily needed to support construction operations on a project but that do not form a part of the finished structure, 2) determining the size and shape of these facilities, and 3) positioning them within the boundaries of the available on-site or remote areas. These so-called *temporary facilities* usually remain on site for a period ranging from a few days to several months or even years, a time period that ranges from the duration of a construction activity to the duration of a major phase or for the entire construction period. In some instances, these facilities are not dismantled after project completion and are, instead, used for operations and maintenance when the project is in use. In other instances, some permanent facilities may be temporarily designated as construction support facilities.

Temporary facilities consist of, among other things:

roads and railroads,
staging areas,
parking lots,
warehouses,
material storage areas,
fabrication and assembly yards,
construction utilities,
and office trailers or buildings.

A detailed list of facilities commonly encountered on construction sites, compiled from talking to field practitioners and from publications such as [Popescu 80a, 81; Neil 82; Rad 83; Handa 87, 88, 89], is provided in Appendix B. One could add to this list lifting and material handling equipment such as cranes and trucks, but I have not done this here. Although the location of such equipment can have major impact on the location of other temporary facilities, representing where these dynamic entities are to be located and how they move about on site would likely involve some type of simulation-based modeling; this is beyond the scope of the SightPlan model.

Appendix B can be used by field managers as a checklist to identify the types of facilities needed. What facilities will be located on site, and how they are sized and shaped depends on many factors, which have to do with the *construction type and scale*, *work organization*, and *project location* (an excellent reference is [Neil 82]). A project can be of industrial, heavy civil, residential, or building **construction type**. For example, an industrial project such as a power plant is characterized by its number of units and their size, and these, in turn, determine what quantities of materials will be delivered and installed. Since such a project has much piping and ductwork, its construction will involve many welders and thus require welder-testing and training facilities as well as weld-testing laboratories. The space available on site will vary depending on whether or not the units are to be constructed on a new site, or added on to an existing plant, among many factors.

The **work organization**, dictated by the nature, type, and scope of construction contracts and labor agreements, affects the construction schedule and methods. The available resources such as space have to be allocated to meet the needs of contractors. The maximum and average site populations determine the sizes of change houses (buildings or trailers where workers can have lunch, change clothes, and possibly shower), sanitary facilities and so on. Regulatory and safety requirements impose explicit constraints on the layout. Whether the job is union or open shop dictates whether or not multiple entrances to the site should be provided.

Given the **project location**, the site geography and topography as well as climatic conditions are known. The location determines site accessibility and must be accounted for in the decision of whether or not to provide on-site or off-site services. The climate influences requirements for change houses, weather protection, or climatized storage areas for materials.

Of the above-mentioned requirements, several are one-time only. That is, they determine the need for certain facilities and their size, and can 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 problems interact with one another: if not enough space is available in an area on site, the size of a facility can be reduced to make it fit while still meeting its functional requirements; conversely, if space is abundant, facilities tend to expand in area although they may be restricted in number.

What is it that one tries to achieve by laying out a site? Certainly, there are high-level objectives, such as getting the project built on time and within budget, promoting safety, maintaining good employee morale, and achieving efficient operations. Yet, a commitment to those objectives does not in any way prescribe how site layout should be performed. Alternatively, there are low-level objectives that prescribe in more detail. For example, one should minimize travel distance and time for movement of personnel and equipment between parking and work positions, between field offices and work areas, and between supply sources and work areas [Neil 80]; it is desirable to have a low ratio of material handling time to production time; and one should avoid creating obstacles for material flow [Chandler 78; Parsons 80; Popescu 81]. These low-level objectives closely tie to particular concerns of the person laying out a site. They may help focus on specific aspects of a layout, but because they are "localized" they may not provide global criteria to judge the overall site layout.

Because so many issues are at stake in laying out a site, it is not surprising that the individual objectives and preferences of the person responsible for the layout's design will affect his or her decisions, and that the resulting layout will be different from one generated by someone else. First, *the role played by the person doing the layout affects the layout.* For example, Handa found that sites laid out by construction managers had better access provisions (such as haul roads, turn-around places, routing of internal roads) than those laid out by general contractors or developers [Handa 87]. In his survey, general contractors had a better safety record than either of the others. Their performance was even higher when operating with their own equipment. Handa suggests these results show general contractors' desire to keep the job accident free, to get the best results from their equipment, as well as to keep the insurance and compensation cost low. He rationalizes that developers were the best ones of the three regarding field materials handling and housekeeping, apparently because they better appreciate the importance of materials handling and its impact on profits. Second, *a person's level of involvement with the project affects the layout that person generates.* Individual contractors view the layout of the area allocated to them from a different perspective than the construction manager. Whereas a contractor will be concerned with the detailed layout and internal organization of the area, the construction manager will focus more on whether or not proper access roads were provided to that area as a whole, and how that area fits globally with those of other contractors on site. Finally, *even when a single individual designs a layout, that person may choose one of several alternative strategies or may go back and forth between them.* Rad writes that constructors tend to commit themselves to one of two strategies: 1) locating labor-oriented facilities (such as sanitary facilities, craft change houses, job offices) closest to the work site, followed by material-oriented facilities; and 2) placing material-oriented structures (such as warehouses, storage areas, and laydown areas) closest to the work site [Rad 83]. That choice of strategy, in turn, may depend on the type of project. For example, he suggests that power companies working on nuclear power plants opt for the second strategy because quality assurance requirements impose strict inventory control. He leaves unspecified how constructors might integrate both concerns into one strategy.

The same factors that make it difficult to choose decision variables to generate a layout make it difficult to express criteria by which to judge the layout. Dressel and Handa both propose a scheme for criticizing an existing layout [Dressel 63; Handa 88]. They suggest that managers make a list of the facilities and planning elements to look for before they visit a site. Each facility is assigned a maximum score for its ideal organization. Facilities are grouped by type, and each type is given a weighting factor. During their site visit, managers should assign minus points to each facility based on its deviation from the ideal in terms of practicality, operability, and organization of the layout. Minus points combined with the weights result in a measure of quality of the layout. Sadly enough, this approach is curative, not preventive. A bad layout is often recognized only when it is too late to avoid it, and when the only remedy is to replan during construction to compensate for unanticipated problems.

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. Handa mentions that, "A major purchaser of construction has established a financial return of 4 to 8 dollars for each dollar spent on preplanning" [Handa 88]. (Layout is, of course, only one cost item of preplanning.) Popescu points out that, because site layout costs are typically charged to project overhead, no one will eagerly pay for them [Popescu 81]. He further mentions that temporary facilities and utilities vary in average cost—and thus in relative importance based on the type of project. For example, it is from 3.5% of direct project cost on high-rise office buildings, 5% on chemical plants, refineries, dams, and offshore platforms, and 10-12% on power plants. Clearly, the impact of site layout becomes more obvious on the larger and more remote projects, such as power plants. In any case, the cost incurred for temporary facilities amounts to substantial sums of money, and this should warrant managers' concern for layouts that can bring major improvements to the overall construction process. A well-organized site facilitates inventory control (for example, no materials are lost in piles of dirt, theft is limited), cuts travel times, reduces noise and dust, prevents obstructions and interferences, increases safety and security, and improves site access.

So far, site layout has been presented as if it were a particular subtask of the construction process. In fact, spatial layout is closely tied to other resource allocation tasks, such as scheduling, labor assignment, equipment selection, and financial analysis. Given so many unknowns and such incomplete information at the moment the layout is generated, default assumptions need to be made. When an initial layout is generated, it may need to be updated as soon as other information gets revised, or when other people get involved in it. In this way, layout design becomes an iterative process, tightly intertwined with other construction management processes. This is illustrated in Figure 2.1.

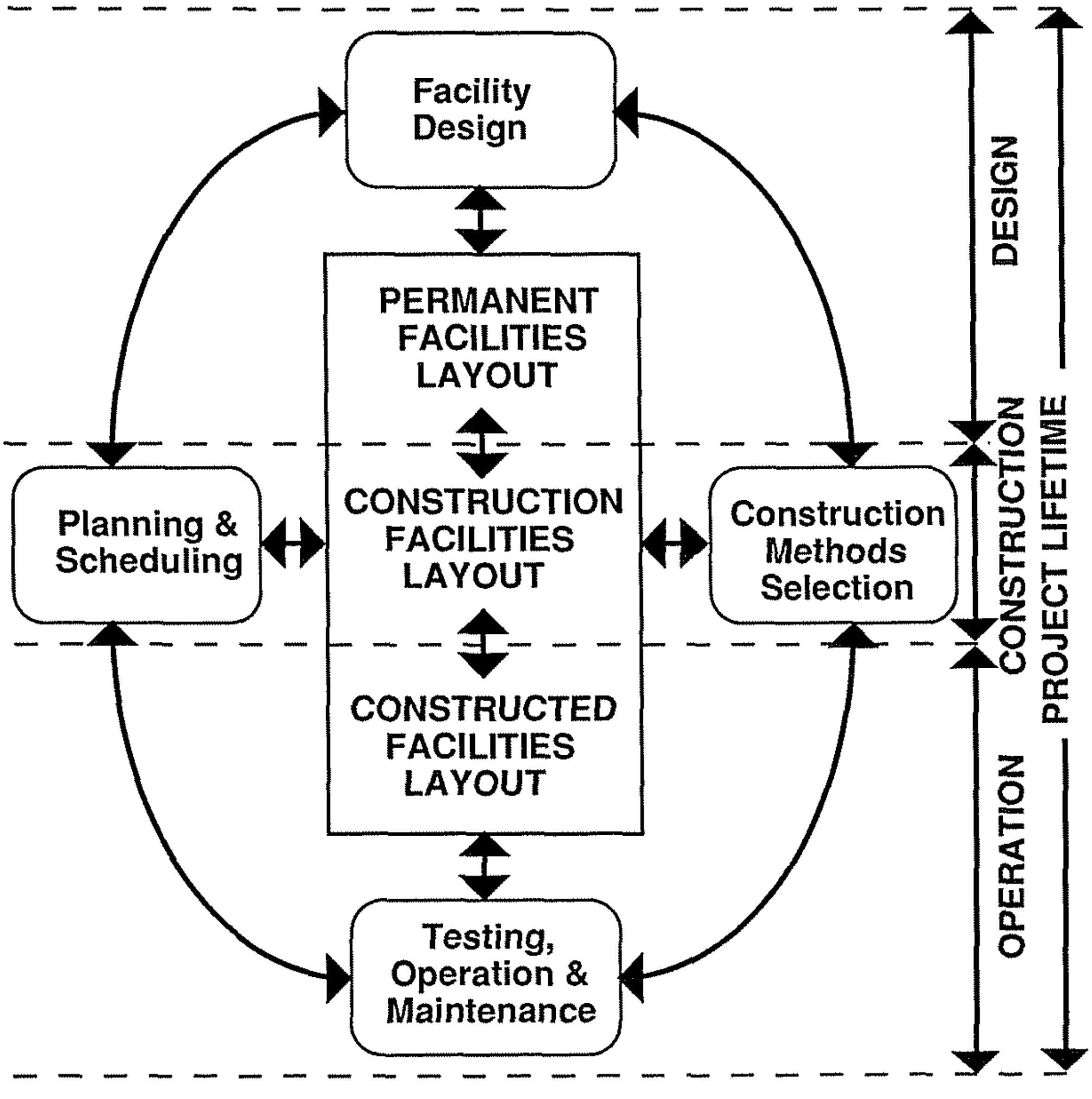


Figure 2.1: The Project Context of Site Layout

Furthermore, temporary facilities can be redesignated as support facilities for project operation, and some permanent facilities could be redesignated temporarily to support construction. The interaction between temporary and permanent facilities can indeed be tightly knit, because decisions made for the permanent design affect the layout of temporary facilities, and because decisions made for the layout of temporary facilities may affect the permanent plant design. Therefore, models for temporary layout should be designed for use for permanent layout as well. This is so because of the many concerns that relate to permanent and temporary layout design, and that could be addressed by considering these designs simultaneously.

Having defined the site layout problem, several factors that affect it are shown in Figure 2.2. With all its complexity, the problem may appear virtually impossible to solve. Yet, field practitioners routinely lay out construction sites. To examine their process, I will sketch field layout practice: who does this task, what information is available, what methods are used, when is the problem solved, and what do the result looks like? I will then describe different physical and computer models that can assist people with this task.

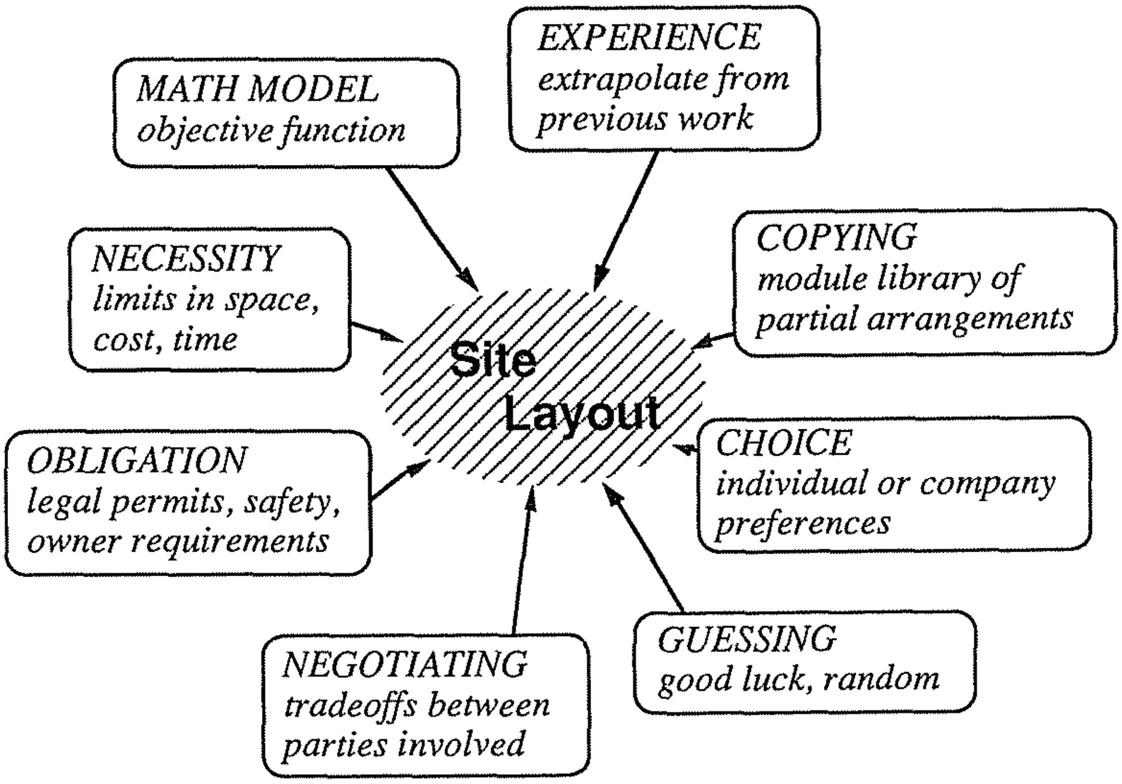


Figure 2.2: Principles, Considerations, and Criteria that Affect Site Layout

2.2 Site Layout Field Practice

In current practice, a project engineer or a superintendent plans the layout of temporary facilities on a project at the beginning of construction. A field manager is chosen because such a person can take full responsibility for implementation and execution of the layout plan. Site layout is often a preplanning task, following substantial completion of design drawings, after civil works (such as clearing and grading of the site, or excavating and installing foundations for the project) have already commenced, but before construction of the project gets too far along. 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 this information helps in generating the site layout.

Because the layout problem is so intertwined with other site management issues, and because of human cognitive limitations (see [Miller 56]), humans cannot possibly keep track of all the factors that could affect the selection, dimensioning, and location of temporary facilities. Field managers have adapted their site layout approach to deal with this situation. First, they extract the main variables from those considered less relevant, so that the problem can be formulated in a concise manner. Second, they divide the global problem into more cognitively tractable subproblems. Possible interactions between variables and/or subproblems initially can be omitted to reduce complexity further. Third, they reason about the decisions to make; once made, they are not changed during further problem-solving. This approach is called *early commitment.* Finally, tools like pencil and paper, icons and templates, scale models, and sometimes mathematical models help conceptualize the problem, and they allow people to inspect the layout visually while it is being developed.

Figure 2.3 abstracts the site layout problem to a comprehensive formulation: it draws a schema of the layout design procedure. Based on the specifics of the project and its surroundings, the manager assigned to plan the site layout can select needed facilities from a checklist (1). From estimates of the labor-force requirements (2), facilities can then be sized (3). After identifying relative closeness requirements (4), layout relationships (5), a desired level of quality of facilities (6), and miscellaneous other considerations (7), the site layout can be generated. Although this schema identifies the steps to be taken for layout generation, the real layout process is not as linear as it is depicted here.

The following examples show how site managers may simplify their task. Managers divide available space up into smaller sub-areas, and delegate the task of laying out contractor areas to the contractors themselves. Once they commit major facilities to a location, they do not modify or even question these locations when positioning less important facilities. Managers adapt layouts from other sites to fit the new project conditions. They group and arrange facilities separately, before positioning the group into the overall arrangement. They sketch in the trial-and-error process of generating the arrangement, before drawing the final plan, and move around templates to test whether or not particular arrangements will work.

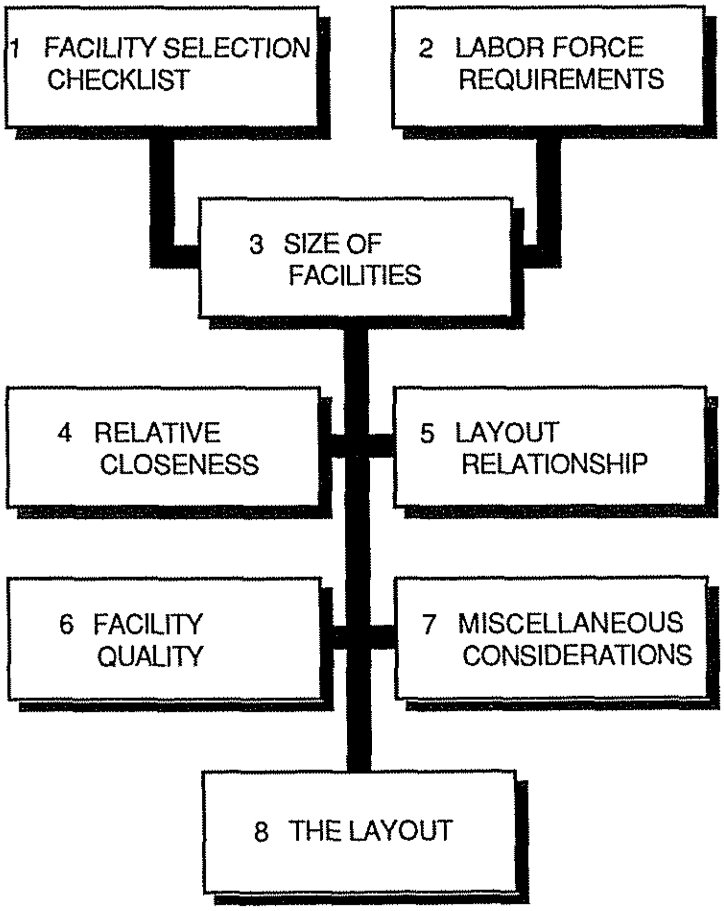


Figure 2.3: Schematic Drawing of Layout Design Procedure
(Figure from [Rad 83]. Note that the "labor force requirements" in my schema are called "manpower requirements" in Rad's.)

Field managers routinely summarize the result of preplanning the site layout of temporary facilities on a *single* site arrangement drawing which is rarely updated as construction progresses. They use this drawing throughout the project. This single drawing, however, contains information regarding facilities on site at successive time intervals during project construction. It really consists of an *overlay* of several drawings. For example, the drawing may be a marked-up site-arrangement blueprint that shows the permanent facilities at project completion. The marks may show major excavations needed for installation of building or equipment foundations. They may demarcate how areas, excavated at one time and backfilled later, are used for material laydown areas during another phase of construction. In addition to this single drawing, special detailed studies may be performed. For example, a separate rigging scheme may locate the huge cranes needed for major lifts.

Because so many changes take place over time, updating to keep track of all facilities, especially those that move around on site, constitutes too much manual work when compared to the potential benefits of performing this work. Any person who is to interpret such a drawing needs to imagine how the layout will evolve over time, and needs good spatial visualization as well. Visual aids and more formal layout methods can facilitate the layout task.

2.3 Models and Methods for Site Layout

wisdom is knowing what to ignore

Models help a problem-solver abstract those aspects or features of a problem considered to be more relevant than others within a specific context of problem-solving. Consider an example in construction site layout, in which facilities occupy physical space: a model may encode facility dimensions by representing them to scale, and show that facilities consist of matter by making them out of wood. Physical or graphical models also make it easier for the viewer to see what is being represented. Models thus remove some of the cognitive barriers encountered by the problem-solver. Because they impose a chosen level of abstraction, models also provide a common representation to better communicate ideas among different spectators and to permit multiple participants to integrate their input. Any model, however, can reflect only a partial and selective representation of reality.

The following subsections review several types of models that assist field managers with laying out construction sites. There are physical models to display a layout and its parts (such as drawings, templates, and three-dimensional scale models), and there are computational methods to generate a layout (such as heuristic and optimization methods).

2.3.1 Physical Models for Layout Display

In this context, **physical models** represent properties of the entity to be represented (reference) in such a way that the entity and its representation have the *same visual appearance*; that is, to a human viewer they *look alike*. Typically, this means that the reference and its representation have scaled dimensions, although some form of accentuation may take place in the transformation of representations. Physical models *represent components of the problem input* to the problem-solver. The method for reaching a solution that is subsequently used by the problem solver is not articulated by these models at all.

Usually, physical models correctly display the spatial relations between the parts of the objects they represent. This most important feature makes them very useful for modeling a construction process as a kind of assembly task. A thorough guide to the many types of models used in construction management was written by [Henderson 76]; I will refer to this work on many occasions.

2.3.1.1 Icons, Templates, and Two-Dimensional Scale Models

Icons, **templates**, and **two-dimensional (2-d) scale models**, three types of physical models, have 2-d representation in common. The two dimensions could be those of a horizontal or a vertical cross-section, or any other cut or projection of the reference. Icons, for example, can be drawings, sketches, or symbols representing a reference on paper or on a computer screen. Templates and 2-d scale models can be simple cut-outs made from cardboard or styrofoam, or from more complex combinations of multiple materials, representing a reference by a tangible reproduction. The thickness of such material is typically ignored, or is not representative of the thickness of the reference.

As common visual aids in construction, sketches and drawings provide a basic means for communicating ideas among the parties involved throughout the life time of the project. They are easy to create with minimal tools. While duplicating drawings is relatively difficult to do manually, photo copiers and computers helped overcome this problem. Most drawings are difficult to change. For that reason, people sometimes draw things over other things, while literally overlooking lines already drawn. The result is a drawing that is difficult to interpret. Here, also, computers have reduced that difficulty. People can easily modify computer drawings, they can superimpose drawing layers, and they can "cut-and-paste" drawings to combine them with one another.

Templates can be used by field managers to help visualize object dimensions and shapes. They are also easy to generate at low cost, but, whereas drawings are part of the project records, templates are usually not valuable enough to be retained after serving the purpose for which they were made.

Templates and sketches are mostly used in trial-and-error processes. Sketches may show how different pieces fit together, but templates also lend themselves to be moved around; for instance, one can use them to check for interferences, or rearrange them until a satisfying layout is achieved. This approach to layout generation is very popular, probably because the person who is moving templates around has the feeling of being literally "in close contact" with the objects to be located. Although these visual aids help a person explore possible alternate solutions, they do not give any guidance towards which alternatives to pursue, nor do they help a user remember which arrangements were previously generated. Henderson and Rad discuss some of the uses for these models [Henderson 76; Rad 82].

2.3.1.2 Three-Dimensional Scale Models

Three-dimensional (3-d) scale models have the advantage over the preceding physical models that they represent to scale all of the spatial relations between the modeled parts of the reference. Hybrid models, in which some parts are shown in 3-d and others in 2-d, are also used. The materials from which 3-d models are commonly made are wood, styrofoam, or plastic, which is often color-coded to simplify interpretation. Models are built to a standardized scale, based on their primary purpose. To fix scales to a few conventional ratios is logical if one wants to construct the model from a generic set of standardized modeling blocks. Depending on the model's use, more or less detail will be represented by it. Despite the fact that scaling is not an issue when a computer generates the model, conventions for display types and visual clarity may demand that a computer model not display all parts in all detail either. Henderson classifies his findings under nine types of models with their typical scale and primary use [Henderson 76]. According to his survey, apparently none of the models used in industry is built primarily to help locate temporary facilities on construction sites, but both the Layout Model (at scale 1/8" or 3/8" per 1') and the Excavation Model (at scale 1" = 40' to 100') have that task as a secondary purpose. However, other models do exist primarily to help locate major equipment.

3-d models are always impressive when put on display. For example, 3-d models made for complex industrial projects, where a lot of interference checking is needed for piping, are studied by designers and construction managers who conceptualize what the final constructed product will look like and how its components fit together. Tangible 3-d models are often so expensive that only one of each kind is made. Besides, if more than a single physical model is developed, consistency management becomes a serious problem when each of them is modified independently over time. These limitations obviously imply that only a limited audience is allowed to access and modify the model(s). Moreover, because such models are quite fragile, they must be protected (for example from being taken apart and reassembled in construction sequencing studies) and thus cannot be exploited to their full potential. Computerized 3-d modeling techniques have overcome these hurdles and are nowadays widely used in place of, or in conjunction with, 3-d physical models in major construction and engineering companies (for example Bechtel's WalkThru [Bechtel 88], Black & Veatch's POWRTRAK, and Stone & Webster's integrated system [Zabilski 89] tie together 3-d graphics packages and engineering databases. Some general design systems are Intergraph, AutoCAD, Catia, and Computervision).

Figure 2.4 and the accompanying text which follows this paragraph, both from [Henderson 76], clearly illustrate the tangible 3-d model as a medium for communicating ideas among different parties involved in the design-construct (and -operate) sequence. (Time periods in this figure are referred to by numbers in parentheses in the text.) The reader can imagine this (somewhat out-dated) tangible model as if it were replaced by a computerized project model. Chapter 6 will point out, for each of the projects studied, at which time the generation of the layout drawings that show the temporary facilities took place. I will then be able to tie my work on layout modeling in with other work on computerized modeling for construction management.

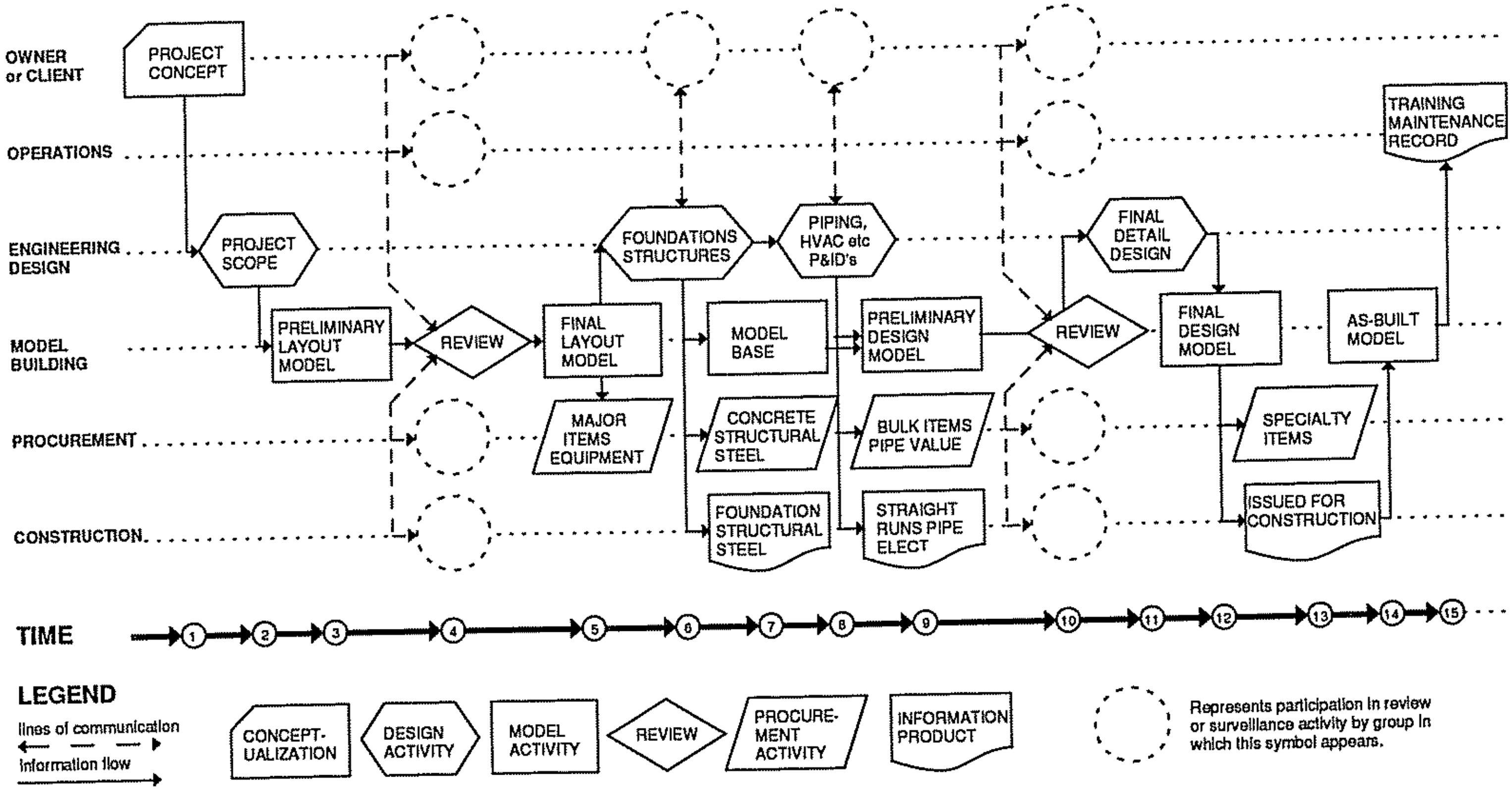


Figure 2.4: Model Information Flow Chart
(Figure from [Henderson 76])

Henderson describes the use of scale models as follows [Henderson 76]:

"The project starts (Time 1) when the owner or client conceives the need for a new facility. Selecting a design engineering firm to design the plant, the client expects the designer to define the scope (Time 2) of the project. That is, the client expects the design engineer to determine the type of equipment and the size of the plant and equipment required to meet the client's needs. When a proposed plant site and the major equipment have been selected, the preliminary equipment arrangement or layout model (Time 3) is brought into use. After the design engineers have decided on one or more possible arrangements for the plant, it is time for a review (Time 4) with all interested groups participating. The owner insures that the basic project concept is being met. The design engineers explain the project scope and the assumptions upon which the proposed layout is based. Construction, operations, and procurement review and study the model to provide input from their perspectives. The model may be rearranged at this review in order to test proposals by any of these parties. All changes and rearrangements should be order to test proposals by any of these parties. All changes and rearrangements should be photographed to provide a record. When all parties agree upon the final layout (Time 5), the information contained on the layout model is used by procurement to order major items of equipment and by the civil and structural design engineers to design the foundations and structural systems to support and contain this equipment (Time 6). The client normally reviews the design and procurement activities.

When the design of the foundations and structures is completed and approved by the client, this information in drawing form is released to the model shop for the production model bases (Time 7). It is also released to procurement for purchase of concrete, reinforcing steel, embedded items, and structural steel, and it is released to construction to start work on the foundations.

As the foundation and structural design is completed, the design engineers start producing P&ID's [*piping and instrumentation drawings*] for piping systems, one-line diagrams for the electrical systems, and specifications for these and other systems (Time 8). Procurement is kept informed of design progress. As information becomes available, purchase orders for bulk items such as stock lengths of pipe, reels of cable, and typical valve sizes are issued. If the model is not complete, drawings are sometimes released at this point to construction for the installation of straight runs of pipe and cable trays (Time 9). By this time the model bases have been delivered from the model shop to the project model designers.

The engineering information is fed to the model designers as it is developed on the P&ID's (Time 9). The model designers use this information to route piping, cable trays, and duct banks, and to locate and orient valves on the model base. Guided by frequent review by the design engineer, this is the critical point in the model design process where interferences are located, identified, and resolved.

When the model is about 85% complete, it undergoes a formal review (Time 10) by engineering, the client, operations, procurement, and construction. At this final review, each system is examined in detail and all information is finalized on the model. If any changes are required at this time, engineering is called upon to redesign (Time 11) before the model review is completed.

After this review, the model is completed and prepared for shipment (Time 12). Procurement issues purchase orders for the remaining items needed to complete the plant based on the final design shown on the model (Time 13). The model is shipped to the job site for use as a construction planning tool and for guidance in installation (Time 13). After the construction has been completed, the model should be revised to reflect the as-built configuration of the plant (Time 14), after which it can be turned over to the plant operations personnel (Time 15) for use as a training tool and as a record."

Construction field practitioners are accustomed to using physical models for layout generation; they seldom use formal computational models. For the sake of completeness, and as a prelude to my research, I nevertheless will review computational models without elaborating on them in detail.

2.3.2 Computational Methods for Layout Generation

Computational models, by my definition, *represent a method* that is applied to the input of a problem in order to generate a solution. They use some representation for the input data, but this representation need not be a picture of what is represented. Computational models have no physical appearance besides that of the code that implements them; the method they implement can be followed by a person but is typically designed to be executed by a computer.

Depending upon which factors the model takes into account, one can distinguish **layout** problems from **location** problems. *Layout* problems represent the entities to be positioned by their area molded in a (usually standardized) shape. For example, a layout problem may consist of locating and fitting rectangular entities close or adjacent to each other, or tightly packing them within a contour. *Location* problems ignore the dimensions of the positioned entities and abstracts them to points. These dimensions are considered irrelevant to the problem because they are negligible compared to the distances at which objects are located. Insufficient space is not an issue in location problems, because these kinds of arrangements are very loosely packed. SightPlan tackles the problem of positioning temporary facilities on construction sites as a layout problem.

The traditional methods for dealing with layout problems are reviewed next, and, for completeness, a few location methods are mentioned as well. A standard work on layout and location methods is [Francis 74]. From my review and an industry survey by Driscoll of software for computer-aided facility layout, no major changes seem to have taken place in the approach to layout problems since the early seventies [Driscoll 86].

2.3.2.1 Heuristic Methods for Facility Layout

The problem of laying out facilities is NP-complete; that is, no algorithm exists to date that can guarantee to solve any kind of layout problem in polynomial time [Garey 79]. Therefore, many specialized models were developed to solve a particular subclass of the general layout problem, but, in spite of this, there are only a few algorithms that promise to optimize, and they do so only for fairly unusual layouts. Most models introduce heuristic rules to arrive at a solution layout. That is, their method distinguishes good from better solutions, but does not guarantee that the best solution will ever be obtained. Often, heuristic methods relax constraints during problem-solving, or suggest solutions that do not meet all of the constraints.

Layout models commonly use a 2-d representation for the space and entities to be laid out, although there are a few exceptions. Typically, models mold the area of such entities to be a composition of unit areas (for example, areas are composed of several elements in a matrix, each of which represents a unit area), or in shapes with rectangular or variable outlines (such as rectangles or polygons). Some models allow for the change of entities' shapes during problem-solving.

Two well-known heuristic methods for solving layout problems are: **improvement** and **construction** [Moore 80]. A third method involves mathematical transformations and is related to graph theory.

Improvement methods follow an iterative process consisting of the following steps:

- create an initial layout,
- generate an alternate layout by modifying the current one,
- evaluate the alternate layout, and
- terminate the iteration, or return to the generation step if desirable.

Depending on the approach, initial layouts can be given by the layout designer, or they can be methodically or randomly generated by computer. Alternate layouts can be generated by swapping two or several adjacent or equal-sized objects with one another. The resulting layout is then evaluated. Many evaluation functions are based on the *AEIOUX* closeness desirability rating developed by [Muther 61] in his work on *Systematic Layout Planning*, which antedated computerized layout planning. Muther suggested relating activities to one another by determining whether or not their proximity is:

A	Absolutely necessary
E	Especially important
I	Important
O	Ordinary closeness OK
U	Unimportant
X	Undesirable

Muther left it up to the layout planner to represent graphically the closeness rating, and to develop alternate arrangements for the layout creatively. Others used his qualitative scale to assess the global quality of layouts. Quite a few systems, however, are interactive and let the user decide when to terminate the process. Note that although this method is called an *improvement* method, the alternate layout is not necessarily an improvement over the preceding one. If continuous increase of the evaluation function value is requested (an approach which is called "hill-climbing" [Nilsson 80]), the iteration may end at a local optimum. This optimum will depend on what the initial layout was and how the alternates were generated. A possible way to overcome local optima is to allow for an occasional or randomized decrease in value of the evaluation function, which is done, for instance, in "simulated annealing" [Kirkpatrick 83].

Probably the best-known implementation of a improvement procedure is CRAFT (Computerized Relative Allocation of Facilities Technique) [Armour 63; Buffa 64]. The CRAFT examples apply to manufacturing plant layout, but the program's method is generic and could tackle problems in any layout domain. CRAFT starts with an initial user-provided layout. It computes a layout's total transportation cost based on sum of transportation costs between department pairs. This cost is the product of the distance between the centers of the departments, the number of unit loads moving between them, and the cost to move a unit load. CRAFT computes the change in transportation cost for pairwise exchanges in department locations. It selects as a new layout the one that achieves the highest gain, and thus uses hill-climbing to reach its near-optimal solution.

One of the few references applying this method to construction site layout is [Rodriguez-Ramos 82]. Rodriguez-Ramos starts with a layout that a contractor generates intuitively, then uses rectilinear distances between facility centroids to compute transportation costs. His method is similar to CRAFT's, but with one modification. Before interchanging facilities pair-wise and hill-climbing to a solution, he identifies the dominating facility in the layout. The dominating facility is that with the highest transportation cost between it and all other facilities. He uses a single-facility algorithm to find the best location for it in the layout, and swaps locations between the dominating facility and the facility at that best location.

Construction methods also constitute an iterative procedure. Without needing an initial layout, this method iterates through the following steps:

- select a candidate for placement,
- place the candidate according to a placement criterion,
- repeat the two preceding steps until all objects are positioned, and
- modify the result if needed.

Again, depending on the approach, a candidate for placement can be designer-selected, or chosen by computer from ranked or randomly listed objects that need to be placed. The candidate is then placed according to some criterion such as desired closeness, minimum material movement or cost, and so on. In this way, objects are placed one at a time, until the final layout is achieved. Several implementations of this method allow the user to intervene in the selection and positioning process, or the user can modify the final layout to make it fit the requirements.

Probably the best-known *traditional* implementations of construction procedures are CORELAP [Lee 67] and ALDEP [Seehof 67]. These programs fall within the class of operations-research (*OR*) methods. Other implementations for layout fall within the class of artificial intelligence (*AI*) methods that are often termed *knowledge-based systems*. This approach was investigated in the early 70s by Eastman [Eastman 72]. Benefitting from advances in AI and expert systems technology, it continues to be of high interest today. Work on knowledge-based facilities layout by Yoshida, and on construction site layout by Hamiani and the present work on SightPlan, attest to this continued interest [Yoshida 86; Hamiani 87, 88; Tommelein 87a, 87b].

The AI approach to layout planning brings flexibility to how the layout objects and the layout process itself are modeled. Chapter 3 will describe in more detail how AI techniques can be applied and what AI issues have been addressed in models for and about design; for now, I will suggest only how this increased flexibility may be exploited. First, *contextual information* can be made available to an AI system so that its capacity to decide how to size and shape objects increases. This information also increases the system's capacity to select which objects to include in a partial or global layout, and to select which object to position at a particular time. Second, with *constraints between objects* articulated and specified individually, the system can reason about which ones to apply, which to ignore, and when to take applied constraints into account. Moreover, if the system were developed to do so, it could even generate its own constraints in order to proceed with problem-solving. Third, when *strategic problem-solving information* is built in, the system might construct the solution to a layout problem entirely on its own. Such strategies can prescribe how the system can back out of dead-end partial solutions, and what intermediate solutions (possibly all) to conserve before proceeding further. Last, provided that *multiple criteria for evaluation* are supplied, the system could display arrangements based on how they meet the various requirements.

A third method is based on graph theory, and starts by representing adjacency relations in a "bubble diagram." It then applies heuristics to transform this graph into a 2-dimensional layout [Whitehead 65; Foulds 85; Hashimshony 88]. The main difficulties in this approach are obtaining the planarity of the graph so that the existence of a layout can be guaranteed, and expressing more than only adjacency in the graph. This approach is often taught to architecture students, but, to my knowledge, has not been used for construction site layout.

This section referenced only a few papers that describe methods to solve layout problems. Chapter 3 will return to the classification of layout methods from the more general perspective of design problem solving. I list additional references there.

2.3.2.2 Optimal Generation Methods for Facility Location

Optimal generation methods attempt to optimize the location of facilities in terms of one or multiple "objectives" while satisfying a given set of constraints. For example, they might begin with a problem definition that enumerates the facilities to be located, possible location points, supply capacity, sets of demand points, demand capacity, and transportation costs. The material flow between facilities expresses their interrelations. They require an objective function that takes the above variables into account, and subjects the variables to constraints that need to be met by a solution to the problem—a solution that will optimize the objective. (In this approach, if a constraint is articulated, it is necessarily imposed.) There are many variations on the formulation of location problems (see [Brandeau 87] for an overview). Depending on the problem definition, mathematical techniques such as linear programming, integer programming, branch-and-bound, or quadratic assignment could derive a solution (see [Francis 74] or [Hillier 80] for more detail). The advantage of these generation methods is that they can guarantee that they will produce an optimal solution. That is, *optimal* in terms of meeting all constraints and obtaining an extreme value for the objective function. In certain instances, however, even when a problem can be formulated so that a solution is known to exist, it may be computationally impractical to try to follow the exact method.

Examples of the application of an optimization method to site layout are given by [Warszawski 73a, 73b]. Warszawski locates a concrete mixing plant, a building blocks manufacturing plant, and a center for cutting, bending, and storing reinforced steel on a site with located dwellings. His objective is to minimize the total facilities' location cost, and his objective function constitutes of three parts: transportation cost, maintenance cost, and installation cost. Warszawski does not solve the generally stated problem. Rather, he considers more simple cases of the general problem with increasing degrees of complexity. These cases are still of such complexity that obtaining their optimal solution is computationally expensive, so he recommends that suboptimal procedures may be preferred over the exact procedure if a realistic number of locations is to be explored. This philosophy is similar to the one taken in heuristic methods in which users agree that a solution—if not optimal—can still be good enough.

While not really suited to deal with layout problems, optimal generation methods were found appropriate to deal with single facility location problems. For example, in construction site layout the location of a concrete batch plant, a single crane, or a haul road may be computed by such a method [Gates 78; Mayer 81; Rodriguez-Ramos 82, 83; Stark 83].

2.4 Conclusions from Literature on Site Layout

The previous two sections have referred to many papers on construction site layout. They made it clear that managers and researchers approach the problem from different angles. Figure 2.5 charts this literature by focusing on two dimensions:

- The level of specialization for a given domain.
- The degree of automated computation recommended to execute the method.

The level of specialization in this diagram shows how domain-specific the topics of the publications are. Papers that describe a particular power plant are shown at the top level of the chart; those that reflect company practice are shown in the second level; those that explain more general approaches applicable to any kind of construction site layout problem are shown in the third level; and those that describe work pertaining to any kind of layout are at the bottom level. The spectrum includes papers describing case studies [Tatum 81; Weidemier 86], manuals for field construction operations [Neil 82], teaching guidelines [Neil 80; Popescu 80], articles recommending layout practice and pointing out issues that need to be addressed [Popescu 78, 80a, 80b, 81, 86], and papers and books that describe generic layout methods [Eastman 72; Francis 74].

The degree of automated computation recommended to execute the method reflects my qualitative assessment of the extent to which an approach can be followed by a person, or whether or not it was designed for implementation on a computer. In the left column are the more abstract guidelines and heuristics or recommendations for people who lay out sites. In the center column are the papers that more specifically address what facilities may be required on a site and how one can manually evaluate existing layouts. In the right column are descriptions of methods to be implemented on computer. I distinguished AI work, which provides computational models mimicking human behavior, from OR work, in which models make use of numerical equations and mathematics to prescribe satisfying or optimal solutions. However, the two types of work cannot unequivocally be separated; for example, each contains so-called heuristic methods that could be classified either way. This large spectrum includes project studies and field manuals describing concerns in site layout [Tatum 81; Neil 82], checklists to help select temporary facilities [Handa 87; Rad 83], criteria to assist field managers in evaluating site conditions [Dressel 63; Handa 88], approaches to step-wise construct layouts [Eastman 72; Hamiani 87, 88], and optimization routines [Warszawski 73a, 73b; Rodriguez-Ramos 82, 83].

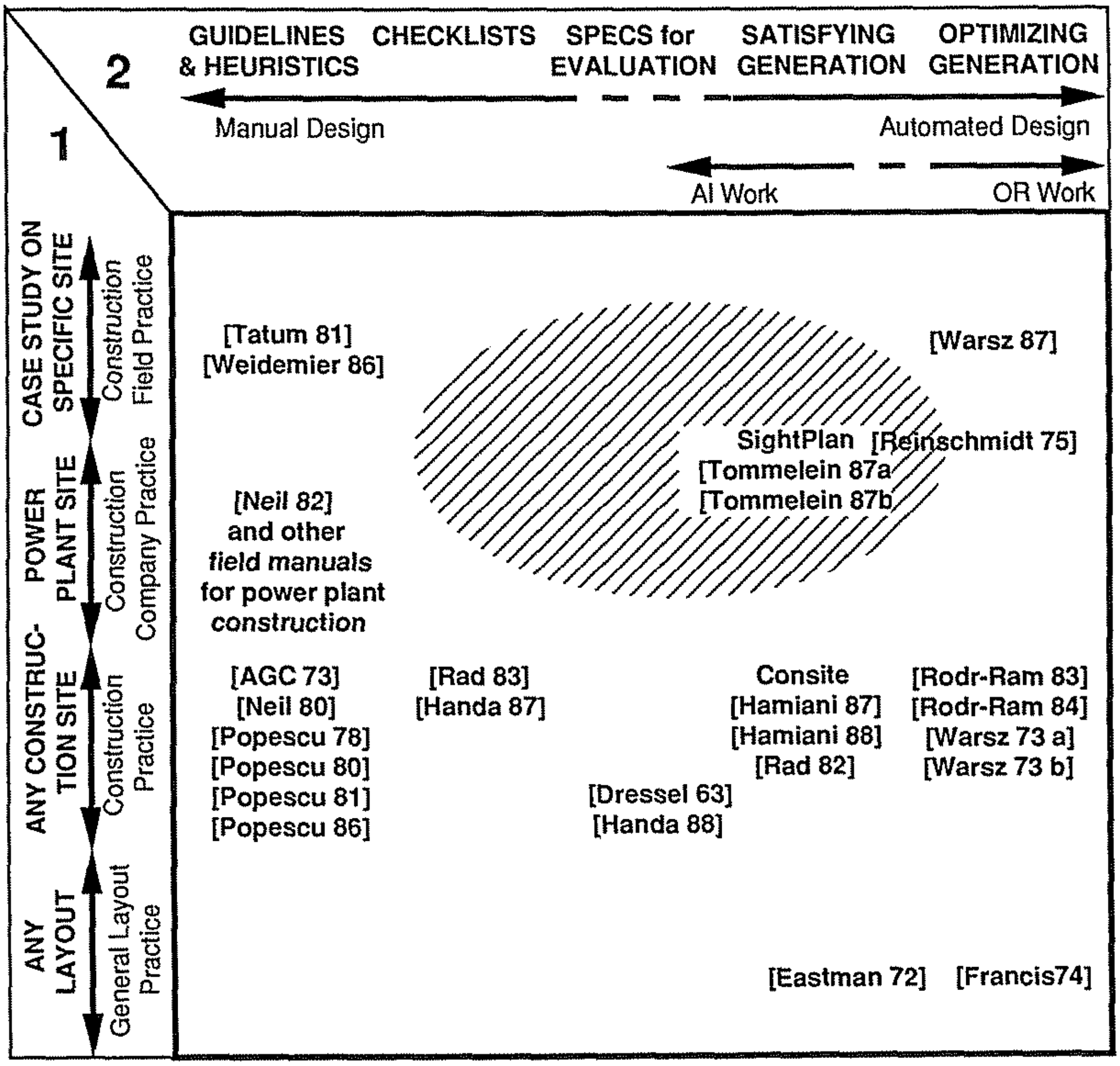


Figure 2.5: Literature on Site Layout Charted by
1) How General or Domain-Specific the Described Work is, versus
2) Whether the Described Method is Applied in
Manual or Computerized Layout Generation

As can be seen from the large cross-hatched oval on the chart, there is a large gap between field practice, shown in the leftmost columns, and formal optimization methods, shown in the rightmost columns. Possible reasons why mathematical layout models may not have gained much recognition and use in construction field practice include:

- Some amount of expertise is required for selection of an appropriate model and for problem formulation, and this expertise is quite different from field practitioners' know-how. [Fisher 84] clearly identified this need for expertise when he built a knowledge-based tool to assist layout designers with these very tasks.

- A substantial amount of data is often needed as input to these models and that information may not be readily available to field practitioners.

- Most mathematical models are implemented as *black boxes*. The procedure that is followed may be counter-intuitive, incomprehensible, or questionable to the person who uses the system. More important, the user cannot easily alter the model when the results are different from what is expected, and thus has to resort to a superficial tweaking of data, that may have been questionable to start with, to achieve the desired outcome.

- Is a field practitioner to be held responsible for the model's results and their implementation? The black-box model does not provide the practitioner with any means to get insight into the process. Nor does it allow the practitioner's intervention to make intuitive changes in order to lead to an acceptable solution. Thus, on the one hand, the field practitioner is expected to blindly accept the program's solution and implement it on the site, while, on the other hand, she or he is held responsible for the program's results. This situation clearly leads to resentment. Woods explains such resentment by the so-called *responsibility/ authority double-bind*: when people refer to a human specialist, they generally pass on *both* authority and responsibility together ([MillerP 83] referenced in [Woods 86b]).

- If the user introduced many simplifications to apply a model, it will take substantial effort to place and interpret the model's results in its broader context.

Many of these issues are well-known shortcomings of formal optimization models and computer implementations [Vollman 66; Francis 74; Hollnagel 86; Woods 86b].

They make it understandable that field managers prefer to use their own models to lay out sites over more abstract models.

The obstacle created by the demand for large amounts of data may be hard to overcome in any model that seeks to represent a complex problem. Notwithstanding this hurdle, models can be developed to reflect more closely the steps a person might take during problem solving, and this is the goal of SightPlan. My aim is to develop a prototype tool that people can relate to intimately, that will encourage them to experiment, and that will assist and support them during problem solving; in short, a tool that will prove the concept of an actual user-machine integrated environment. That this is a desirable model has been argued by others who pointed out limitations of computer models. See, for instance, [Paulson 72] for early ideas on the need for user-machine interactive systems in construction management. SightPlan is meant to be a tool that assists construction managers in doing site layout. Therefore, SightPlan should be knowledgeable about field practice in construction. The kinds of data and guidelines the system can rely on are described next.

2.5 Available Knowledge from Management Practice

Neil refers to "several basic principles, many considerations, and some criteria" that field managers have to apply with good judgement [Neil 82]. My contention is that many of these principles, considerations, and criteria are well-known facts in construction practice. They are often compiled from written construction documents or derived from field data obtained on previous projects. But because there are so many facts to consider, they are seldom assembled in one formal system, except possibly in a manager's mind. Many of the concerns to which Neil alludes can be categorized according to the degree to which the knowledge they contain is project-specific. I distinguish three of these categories:

1. Generic construction knowledge or "common sense."
2. Knowledge pertaining to a Specific construction type.
3. Project-specific knowledge.

The following sections describe these different layers of knowledge.

2.5.1 Generic Knowledge for Site Layout

After reviewing guidelines and construction field manuals in documents of different national origins (American [Stone&Webster 79; Neil 82], Australian [Weidemier 86], Belgian [FVB 84], Dutch [Van Hattum 72], and German [Dressel 63]), I found my intuition confirmed: there exists some sort of *standard practice* in construction management for doing site layout. One can only speculate on how such a standard practice may have developed. Part may be the result of cross-pollination within the industry: construction workers move from one project to another, across regional and national boundaries, and learn different companies' management practice; the same contractor works on many sites, each run by another manager; owners who build several projects get involved with construction teams that are composed in varying ways; and so on.

This standard practice—or what I will call *common sense*—in its manifestation at one level, can be traced to two sources.

1 THERE ARE THE *PHYSICAL RULES* THAT EVERYONE HAS TO COMPLY WITH.

For example, a road must be of a minimum width or a truck cannot drive over it, a warehouse must be large enough for it to store certain equipment, a railroad must extend to within crane reach if materials are to be lifted off a railroad car, a pile of sand or another bulk material naturally occupies a computable area on site. These physical constraints are well-understood and can be translated into simple mathematical rules to be applied systematically.

2 THERE ARE THE *LEGAL AND PRAGMATIC RULES* THAT ARE LARGELY OBEYED.

Examples of legal rules are OSHA regulations or other safety standards (such as [OSHA 85; US Army 87]) that need to be complied with. Cranes or other high equipment may not be obstructed by overhead power lines for reasons of safety. Table 2.1 shows requirements on clearances around storage areas for combustible materials. Examples of pragmatic rules are: a stack of materials may not exceed a certain height for easy reach; a parking space should be located as close as possible to the work to cut worker travel time; and it must be rough-graded and reasonably surfaced against mud or clay.

12.B.04. At least 10-foot clearance from buildings or structures shall be maintained for piles of lumber and other combustible materials to be used in the construction.

12.B.05. Driveways between and around open yard combustible storage piles shall be at least 15 feet wide and free from accumulation of rubbish, equipment, or other articles or materials. Driveways shall be so spaced that a maximum grid system unit of 50 ft by 150 ft is produced.

Table 2.1: Clearances for Combustible Materials
(Excerpt from [US Army 87])

On another level, the types of rules mentioned above have been combined with previous construction experience, resulting in directions to guide layout arrangement. Such directions might *rank* temporary facilities *by the priority* each facility has in the allocation of prime space in the layout. The manager who follows these directions to lay out a site first identifies prime space (such as fabrication yards, and short- and long-term laydown areas); then works down the list of temporary facilities to pick a facility, and finds a satisfactory location for it. Ideally, facilities are located in close proximity to the project under construction, because all permanent materials related to that facility need to wind up there ultimately. Thus, the area immediately surrounding the project is prime space. Table 2.2 shows two such rankings found in the literature.

Facility ranking based on closeness to work [Rad 83]:

1. Sanitary Facilities
2. Craft Change houses
3. Job Office
4. Warehouses
5. Storage Facilities
6. Laydown Areas
7. Warehouse Office
8. Brass Alleys
9. Fabrication Shops
10. Time Office
11. Parking Lot
12. Test Shops

Rules on premium space for premiumactivities set priorities on proximities [Neil 80]:

(1) tool rooms, common item issue points, sanitary facilities, hoisting and other constr equipment needed for work underway and materials currently being installed

(2) project offices, warehouses, fabrication facilities and batch plants

(3) parking lots, training facilities, equipment maintenance shops, and open laydown areas

Table 2.2: Two Examples of Ranking of
Facilities Based on Priority for "Premium Space"

Obviously, physical constraints or legal and pragmatic rules do not prescribe one and only one way for arranging layouts. Moreover, there are so many rules, and they could reach such a level of detail that it may be all too easy for a manager to overlook some.

This is precisely where the thoroughness with which a computer can impose or check each requirement is ideal. Before elaborating on how my model will use these kinds of rules, I will discuss the next layer, which examines the level of detail of knowledge related to a specific type of construction.

2.5.2 Layout Knowledge Specific to Power Plant Design

Dressel pointed out that field managers might want to devise *reference plans* for a given project size and scope for a specific type of construction [Dressel 63]. These plans could then be further refined for an individual project, to take into account location-specific data and project data that varies over time. This is, in fact, what has been done in industry. When a construction project is of a particular type of which many instances have been built, then its design concepts lend themselves to generalization.

For example, in the case of power plant construction, large engineering firms that had built several projects of a similar nature developed so-called *reference plants* based on well-established plant concepts and system configurations and did so for typical ranges of plant capacities (for example, Stone & Webster has 350–600 megawatts and 600–900 megawatts Reference Fossil Power Plant Models [Stone&Webster 78], see Figure 2.6).

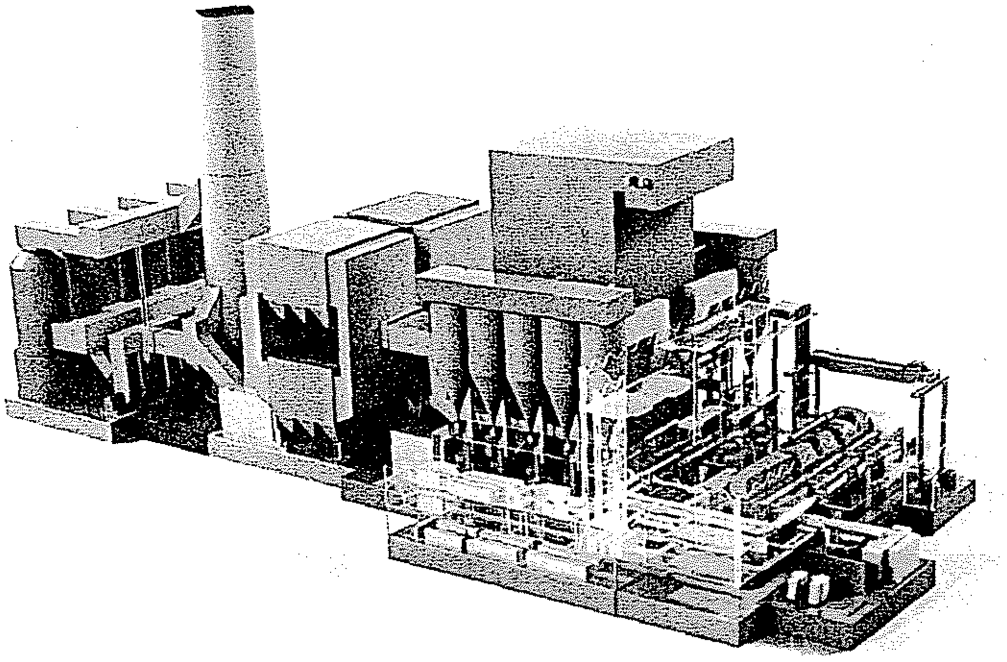


Figure 2.6: 600-900 megawatts Configuration of a Reference Fossil Power Plant Model (Figure from [Stone&Webster 78])

Descriptions of reference plants consist of sketches of the building volume, coarse layouts, and a plausible milestone schedule. Supplements to these documents are construction field manuals which summarize company practice. Such manuals usually have a section on temporary facilities: they provide rules to estimate the needs for temporary facilities based on any one or several of the features of the reference plant. Table 2.3, an excerpt from Stone & Webster's field construction manual, shows how one could obtain a *first-order estimate* on the size of warehouses needed for construction of non-nuclear power plants [Stone&Webster 79]. Note the very simple linear relationship between unit size and warehouse area estimates. Site weather conditions are crudely taken into account, but the time frame within which each project might be built is omitted entirely.

SINGLE UNIT	**NON-NUCLEAR**	
MegaWatt Rating	Min South	Max North
200	2,000 sq ft	4,000 sq ft
400	4,000 sq ft	8,000 sq ft
600	6,000 sq ft	12,000 sq ft
800	8,000 sq ft	16,000 sq ft
1,000	10,000 sq ft	20,000 sq ft
Two Units	Multiply Total sq ft x 1.5	
Three Units	Multiply Total sq ft x 1.75	
Four Units	Multiply Total sq ft x 2.5	

Table 2.3: Warehouse Sizing Estimates for Non-Nuclear Power Plants
(Table from [Stone&Webster 79])

In addition to these guidelines, estimators who have worked on many similar projects may have developed their own rules-of-thumb, so that they can perform "parametric estimating." For example, based on the boiler capacity or the volume included by the main power building, they can estimate material quantities such as those of the structural steel and project the area required for long-term laydown.

An industry survey performed by Rad is summarized in Table 2.4 [Rad 83]. This table gives orders of magnitude for the areas needed for temporary facilities, such as construction support buildings, parking areas, and laydown areas. Due to the fact that the data in the table were derived by *factoring out* many project-specific attributes, it may have only limited relevance to project managers who lay out a given site. Conversely, papers that describe individual projects [Tatum 81; Weidemier 86] and shed light on many *more* attributes than those pertaining to the reference plant, may provide data that lacks generality and that may not be directly applicable to other sites. It is thus up to the people who develop reference plants to do *case-based reasoning* by choosing relevant attributes for a type of project. These attributes should be neither too general nor too specific to be useful to layout designers. (Case-based reasoning is an active area of current research in artificial intelligence and design.)

100 MW NUCLEAR	3x1300 MW	FOSSIL
Construction Bldgs and Misc Facilities	1 acre	10 acres
Parking Areas	2 acres	15 acres
Laydown Areas	4 acres	90 acres
Permanent Areas	3 acres	85 acres
TOTALS:	10 acres	200 acres

Table 2.4: Area Estimates of Temporary Facilities
(Table from [Rad 83])

The practice of conceptualizing reference plants appears to be common in the power industry, and reference plant designs and construction field manuals are generally available at large engineering firms. This suggests that the payoff for using these documents must be high enough to warrant their generation. But, though they could constitute a good starting point for site layout design, the information in them is often so deeply buried in these documents that it becomes hard to retrieve. A more accessible way of storing this information, for instance in an easy-to-query computer system, would therefore be most useful. In addition to this kind of generally applicable rules of thumb, however, the reference plant description needs to be complemented with project-specific information. What the project-specific layer of knowledge entails is described in the following section.

2.5.3 Project-Specific Knowledge

Many factors that make a given project unique have been abstracted out or were compounded together with others, so that they did not appear explicitly in the rules and guidelines given in the above two sections. These factors are *location-specific*, such as site boundaries, topology, proximity to other facilities, available utilities, and climate; *project data that vary over time*, such as total construction duration, site population, and material delivery schedules; specific decisions on *permanent facility design* and *organizational characteristics*, such as contract structure, responsibility assignment, and work organization. Several of these factors have a substantial impact on the requirements for temporary facilities, and should therefore be included in the layout model.

This is where the papers of [Tatum 81] and [Weidemier 86], which describe individual projects, open the eyes of readers not familiar with the intricate concerns of site managers. So many issues are at stake in site layout that it may be virtually impossible to model them all. Moreover, it is hard to get a grasp on what issues occupy the mind of the person laying out a site. Because I could not find a single document describing exactly how any one person had *derived* the dimensions or location of a facility when given a peculiar site, I had to limit myself here to providing a somewhat general example, which could have fit equally well in the preceding section. Figure 2.7 illustrates how a parametric rule, found in a construction field manual, was instantiated with site-specific data on peak construction labor force. Chapter 5 will look at two specific power plant sites in more detail.

Many of the factors that play a role in the layout of temporary facilities are mentioned in a multitude of project documents. One step towards facilitating site layout design would be to provide easy access to and cross-referencing between such documents. Another step would be to use these factors as parameters in a model. Yet a major shortcoming is that the way in which these factors affect the layout may not have been articulated at the time a model was developed. Only at the moment that such a model is used, the person using it may realize that the factor is important and should be introduced. If a model is to emulate closely the way a person designs, the model must allow the user to customize it. One way of doing this is to build a system with which the user can easily interact. These ideas will be fleshed out in Section 2.6, which discusses SightPlan's research direction. First, I will summarize the ideas concerning the layers of knowledge that represent available data from management practice.

Car Pooling Factor: describes the number of workers that come to the site in one car.
≈ 1.4 workers per car for manual employees and ≈ 1 for non-manual employees;

Unit Parking Factor: gives the area required per car, including aisles and driveways.
≈ 360 sqft / car or 40 m^2 / car

Example: assume the following **peak construction labor force**:
Single Unit ≈ 1850 workers; Double unit ≈ 2,800 workers
For a double unit: 2,800 workers / 1.4 workers/car ≈ 2,000 cars
2,000 cars x 350 sqft/car ≈ 700,000 sqft ≈ 16 acres
or 2,000 cars x 40 m^2/car ≈ 80,000 m^2 ≈ 8 ha

Figure 2.7: Excerpt from Construction Field Manual on Facilities for Nuclear Power Plants (Figure from [Stone and Webster 79])

2.5.4 Conclusions Regarding Layered Knowledge

Three layers of knowledge can classify known facts pertaining to construction site layout. Though these layers may not be defined uniquely, there seems to be a natural division between them. It is definitely true, though, that the knowledge they contain is *not* agreed upon industry wide. [Rad 83] surveyed 36 construction companies on their approach to site layout. His work revealed that *companies had similar practices*, but when asked about specific data, *large variations in actual numeric values* became apparent. Rad provides minimum, maximum, and average values, but did not say to which extent the data collected from different companies were comparable. Table 2.5 shows the results of Rad's survey.

	MIN	MAX	AVERAGE
Car Pooling Factor	1	4	1.7
Unit Parking Factor	250	400	330 sqft
	22.5	36	30 m^2
Craft Change Houses Area per Worker	1	30	11.3 sqft
	0.09	2.7	1 m^2
Number of People per Brass Alley	100	250	175

Table 2.5: Ranges on Area Estimates
(Table from [Rad 83])

This conclusion does not come as a surprise; if general agreement on the data had been found, some paper or textbook certainly would have reported that very finding. So, without taking the precise numeric values for granted, it is still worthwhile to realize that:

1. Across the industry there exists a set of variables, implicitly agreed upon, that help determine sizes and locations of temporary facilities.
2. There are rules-of-thumb that provide estimates for these variables.
3. These rules can be *articulated.*
4. Field practitioners find these estimates *useful*, even if they are only first-order estimates for sizing and locating facilities on site.

Once these variables and rules are aggregated into one system, that system will constitute a body of accumulated field experience that 1) can be tested, validated, and improved upon, 2) can be communicated and effectively transferred to other people, and 3) can be tied together with other knowledge on layout design or construction management.

2.6 SightPlan's Research Direction

As mentioned in Section 2.4, the goal behind SightPlan was to develop a computer program that would model closely the steps field managers take while laying out temporary facilities on a construction site. In order for such a system to provide its user with adequate support, it will need to know about field construction practice. Section 2.5 expanded upon the kinds of knowledge from field practitioners available for computer implementation. SightPlan will build upon that knowledge.

SightPlan concentrates on site layout for power plant construction. Indeed, most documented information on site layout relates to this type of construction. Possible reasons for this are that: 1) the cost of the temporary facilities is substantial on power projects and large quantities of materials are involved, so the expected gain of doing a better job on the layout is high; 2) the plant designs and construction methods are fairly standardized so that generalizing about site layout conditions is a reasonable undertaking, and publishing papers on the subject is relevant; and 3) because it is an industrial type of construction, managers are quite pragmatic about finding a solution to the layout problem. In addition to this, finding power plants under construction and obtaining cooperation from field managers on case studies turned out not to be too difficult.

As mentioned in section 2.1, there are basically two ways to apply the different pieces of field knowledge to layout generation, although there can be interaction between them. First, there are one-time requirements that establish the need for facilities and govern facilities' size and/or shape and that can be met before generation of a particular solution starts. Second, there are requirements that affect the facilities' location and that play a role during the generation of a solution. The first set of requirements is used to define input to the layout problem to be solved by SightPlan. The second set is used to model how a person decides step-by-step what facility to pick next and how to position it on site, a method called *constructive assembly.*

The following chapter will discuss what the constructive assembly method entails and how it indeed closely models the way people design layouts. For the reader interested in artificial intelligence, I will place SightPlan in a broader context of on-going research work by reviewing literature on spatial reasoning and design. The reader mainly interested in the construction management aspects of this work can skim through the conclusions of Chapter 3 and jump ahead to Chapter 5.

From the existing expert system technology, I have selected an architecture that lends itself to the implementation of SightPlan's layout method. The BB1 blackboard architecture on which SightPlan builds is described in Chapter 4. The two cases on which SightPlan relies are detailed in Chapter 5. Their implementation models, in which SightPlan approaches site layout from a preplanning standpoint, are discussed in Chapter 6. The main accomplishment of this research is that it proposes a framework for representation of site layout knowledge and that it demonstrates how such knowledge can be applied using a generic computer architecture for solving spatial arrangement problems.

Chapter 3

Issues in Modeling Design

The preceding chapter presented the problem of laying out construction sites (Section 2.1) and outlined how construction managers approach their layout task (Section 2.2). The finding that managers use constructive assembly of arrangements as their problem-solving method, together with the description of the knowledge available from management practice, are essential to building a system that models this design process. An appropriate representation of the knowledge for site layout used by field managers, and an architecture for implementation, still need to be chosen.

This chapter views site layout as a typical design task. As such, it situates SightPlan in the context of research on AI approaches towards modeling design processes and designed artifacts. This clarifies the choices made regarding the implementation of SightPlan, and allows me to justify more clearly how SightPlan may contribute to the state of the art in design modeling.

3.1 Layout Design Viewed as an Arrangement Task

Hayes-Roth defines a *task* as "the process of using a particular problem-solving method to solve an instance of a particular problem class" [Hayes-RothB 87]. She also defines a *problem class* by its characteristic inputs and outputs, and a *problem-solving method* by the knowledge a problem solver uses and the operations it performs in order to solve a particular problem. For example, any problem in the class of *arrangement problems* provides as inputs: a set of symbolic objects, a context, and a set of constraints. It requires as output one or more arrangements of the objects in the context so that each arrangement satisfies the constraints. Alternative methods for solving arrangement problems are *selection*, *refinement*, and *construction*.

Site layout consists of spatially arranging selected temporary facilities on a given site to meet a set of predefined constraints. It is an arrangement problem in the *domain* of civil engineering and construction. The class of arrangement problems incorporates problems in other domains, such as garment cutting, landscaping, newspaper layout, architectural design, factory layout, and VLSI layout. Such arrangement problems are also design problems when the design task is narrowly defined as *an iterative process that consists of sizing, shaping, and positioning objects in space and time in order to create an artifact.* One might add to this definition *the acts of designing the objects and the constraints to be met between them.* This view on the design task is not generally accepted, of course, and every researcher adheres to her or his own conception of the meaning of design. See, for example, [Gero 89] for different viewpoints on design and design research in AI. Yet, the given definition enables one to look at models of the design task that researchers have developed for applications in various domains and extract what these models have in common. Across domains, models can share generic problem-solving methods and general implementation architectures. Within domains, each uses very specific knowledge to gain problem-solving power. Making this distinction yields more insight in design modeling.

A model of a design task must represent 1) the *input* and 2) the *output* of its problem class, and 3) commit to a *problem-solving method* that will solve problem instances of that class. Questions a model builder needs to answer are therefore:

1 WHAT IS THE INPUT TO THE MODEL?

Issues related to problem formulation, variable selection, representation language, and knowledge acquisition—the so-called "bottleneck" of AI—need to be addressed.

2 WHAT METHODS ARE AVAILABLE TO SOLVE THE PROBLEM?

The model builder must have the knowledge to select a problem-solving method, and also needs knowledge for the application of the selected method.

3 HOW ARE THE RESULTS INTERPRETED?

The model user must view the results of problem-solving in light of the initially posed problem and any simplifying assumptions incorporated into the model.

The following three sections will address each of these questions. References are presented as examples from a large body of research, rather than as parts of an exhaustive review. One of the few papers reviewing the state of the art in a design domain is [Finger 89] on mechanical design. This chapter only outlines key research issues in

AI systems for modeling design. It portrays the intellectual context within which SightPlan developed.

3.2 Spatial Representation and Reasoning

Spatial representation and reasoning are subjects of a body of literature that spans a variety of disciplines, such as motion planning, robot vision and image processing, natural language understanding, linguistics, cognitive psychology, and engineering design. Some of the difficulties that spatial—and in similar ways, temporal—representation and reasoning generally encounter relate to human perception and cognition, abstraction for representation, and communication of ideas. To overcome hurdles posed by these difficulties, a model builder needs to make many pragmatic choices.

One problem with modeling space is the **representation problem**. Davis' work on MERCATOR for spatially mapping the world that a robot perceives provides a good example of the complications that arise when a robot deals with incomplete, uncertain, and possibly contradictory sensory information [Davis 81, 84]. The robot must maintain a coherent internal representation of the information it obtains, given that it does not perceive everything at once. Nearby objects appear in more detail than do remote objects; objects are hidden behind others and therefore only are seen from certain perspectives; and so on. Davis finds that a simple cartesian coordinate system does not provide a rich enough representation to store all information needed for this task.

Choosing any representation scheme implies some loss of information in translation. This applies for pictorial as well as verbal representation languages. Languages cannot be freed from paradoxical interpretations; they have a limited vocabulary and semantic ambiguity. Consider, as an illustrative example, spatial prepositions in English, such as "closer than" and "near." How many such expressions are there? What do they mean? Can they express all relationships possible?

Thirteen topological relations exist between two one-dimensional intervals, as might exist in time or in linear space. Allen lists these relations and chooses a set of words in English to uniquely name them: "before," "equal," "meets," "overlaps," "during," "starts," and "finishes" (Table 3.1) [Allen 84]. In this context, some of the words retain their intuitive meaning; for instance, "between" and "before." Others need to be interpreted restrictively and are meaningful only in selected interpretations; for instance, "starts" and "finishes" are meaningful only in a time dimension.

Relation	Symbol	Symbol for Inverse	Pictorial Example
X before Y	<	>	XXX YYY
X equal Y	=	=	XXX YYY
X meets Y	m	mi	XXXYYY
X overlaps Y	o	oi	XXX YYY
X during Y	d	di	XXX YYYYYY
X starts Y	s	si	XXX YYYYYY
X finishes Y	f	fi	XXX YYYYYY

Table 3.1: 13 Topological Relations Between Two Intervals
(Figure from [Allen 84])

In this syntax, 169 (or 13^2) topological relations exist between two rectangles in two-dimensional space. Yet, the English language does not provide 169 *single* words to uniquely name each one. (Nor would most other languages; usually there is no real need for enumerating all relationships.) How many words would one need to label relations between arbitrary *shapes* in two-dimensional space? Image three-dimensional space, and time! Obviously, inventing new words for each specific purpose would remedy the apparent shortage, but that is in most cases not necessary. In daily usage, a few simple terms describe several combinations of positions of two objects, and this expressiveness suffices.

Depending on the model's grain size, one could consider two objects *next-to* each other as *adjacent* objects. But context comes into play here. Two whales swimming "adjacent to" each other may be tens of yards apart. Two ants following each other "at a great distance" may be several feet apart. The distance in each of these situations is much greater than that between two people sitting "next to" each other!

Because it is not always desirable to represent shapes in full detail, spatial abstraction is appropriate. The interested reader can find many examples of abstraction mechanisms in natural language in [Tommelein 88] and in the linguistic studies that served as its source documents [Herskovits 82, 85, 86; Hobbs 85; Pick 83; Retz-Schmidt 88; and Talmy 83]. Some common abstraction mechanisms are the following:

- **IGNORE OR SIMPLIFY SHAPE**
  "How far is San Francisco from New York?" Cities are mentally abstracted to points between which the distance can be measured.
  "A *round* apple:" This is a simplification of the apple's shape.

- **SHARPEN BOUNDARIES**
  "He sat at the edge of the water." Though water moves with the tides, "edge" is abstracted to be a line with well-defined boundaries.

- **DO AWAY WITH SUBJECTIVE RELATIONS**
  The position of a third object can impact the relation used to describe two other objects (Figure 3.1). This contextual information is often ignored, and spatial relations such as "to the right of" are approximated as strictly binary relations.

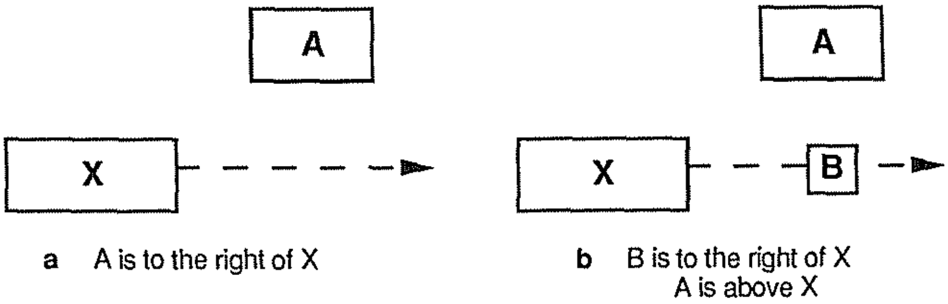


Figure 3.1: Presence of Objects in the Environment other than the Objects Involved in the Relation (Figure from [Herskovits 85])

- **INTRODUCE SYMMETRY**
  One often introduces symmetry when thinking of a relation such as "next to." In fact, in common use it is often not symmetrical. One could say, "The trailer next to the turbine building." But it would be rather awkward to talk about, "The turbine building next to the trailer," because a trailer is more mobile and smaller than a turbine building and it is therefore usually not used as a point of reference to locate a larger and more permanent building (example modified from [Talmy 83]).

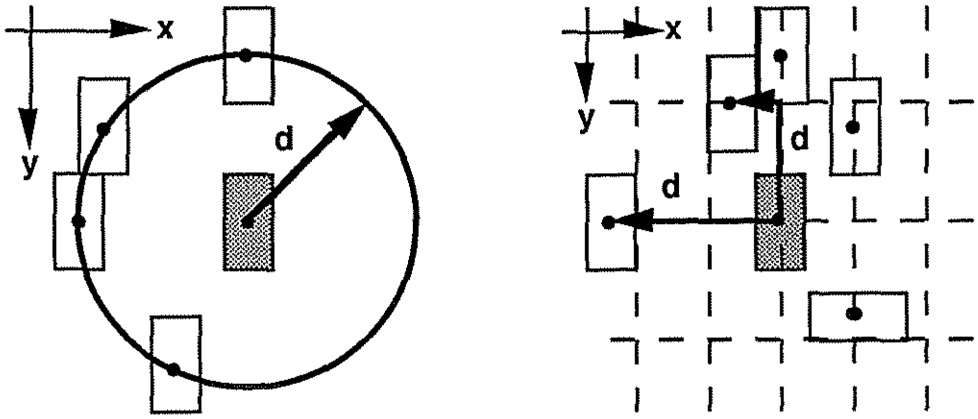
An AI system that models human intelligence would sensibly model spatial entities after a human's internal representation. What that human representation is, however, remains unclear (see, for example, [Anderson 78, 79; Hayes-RothF 79; Pylyshyn 79]). I will therefore resort to modeling spatial reasoning by building a system that satisfies geometrical and topological constraints. Constraints are expressed by spatial prepositions with simple meanings. SightPlan objects are abstracted and all are represented by rectangles.

A knowledge engineer who crafts knowledge must pragmatically choose a suitable representation and abstract what has to be represented accordingly. Some guidelines for abstraction are:

- Abstract away details irrelevant to problem solving.
- Add desired properties to the object's description.
- Keep details that are natural for explanation.

There are many representations of space, each with its own computational advantages and disadvantages. For example, to represent solid models there are quadtrees and octrees, 3-d solid models with hidden line elimination and shading, wire frames, naturally curved forms and shapes, vertices and node networks, and so on. SightPlan represents all of its objects as simple or aggregated rectangles. To define a distance metric in space there are: the Euclidian distance, the Manhattan distance, and the minimal orthogonal distance (Figure 3.2 a, b, c respectively). The latter distance metric is an unconventional one. SightPlan uses it because it is computationally very efficient.

Besides the representation problem that the knowledge engineer faces, the **communication problem** arises during the knowledge acquisition phase (Figure 3.3). This is a *language problem* in that the expert and the knowledge engineer may interpret words differently. But it is also a *knowledge problem* because different experts may disagree with each other. For more detail on this subject the reader can consult [Mittal 84] and [Hart 85]. However, I decided on many abstractions in SightPlan pragmatically to deal with the representation and the communication problems of construction site layout. In this context, I selected a small set of spatial relations and restricted their meanings tightly so that they are unambiguous and expressive enough for my application.



a Euclidian Distance Between Centerpoints

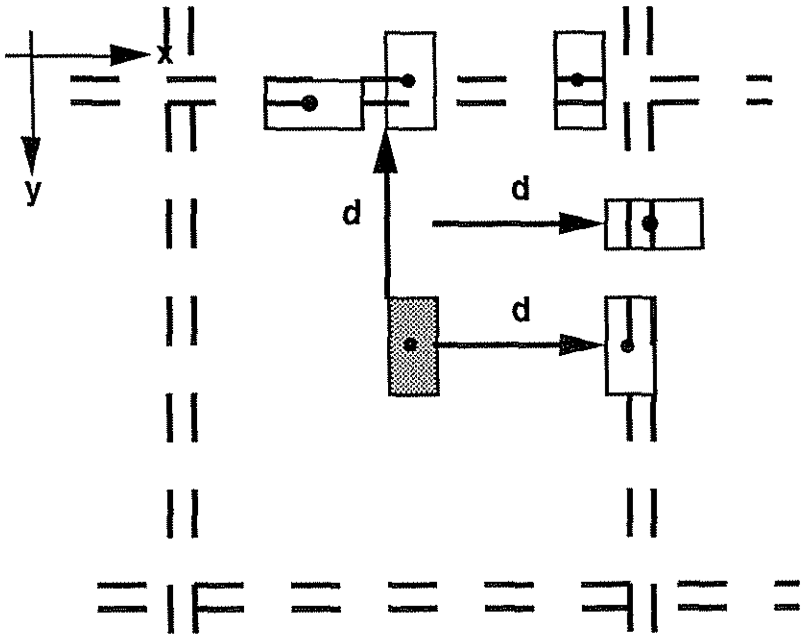
$$d = \sqrt{\Delta x^2 + \Delta y^2}$$

Any point on the circle is a position for the white rectangle at a distance d from the grey rectangle.

b Manhattan Distance Between Centerpoints

$$d = \Delta x + \Delta y$$

The white rectangle has only a finite number of positions on the grid at a distance d from the grey one.



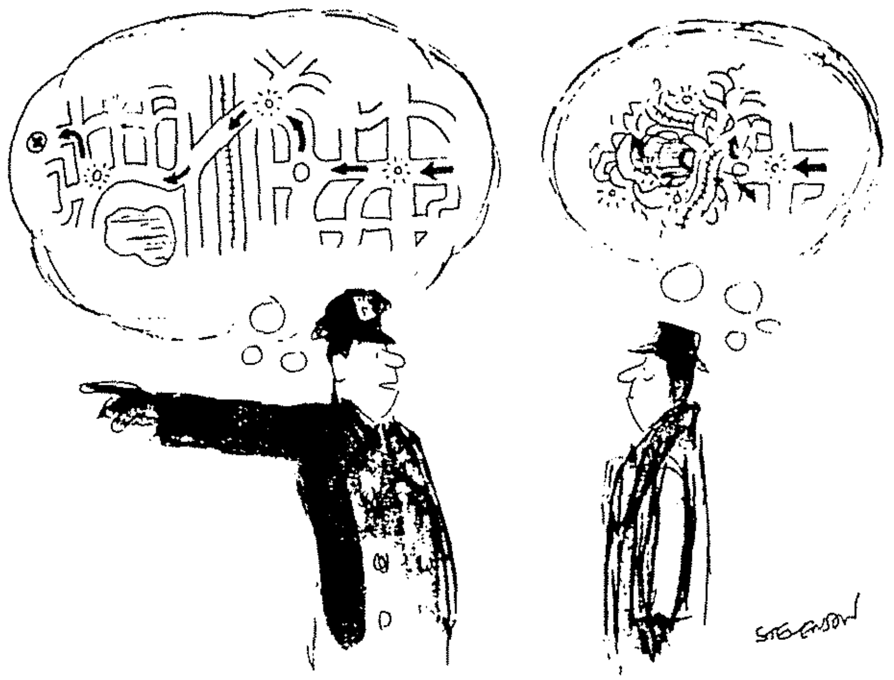
c Minimal Orthogonal Distance Between Edges

$$d = \min(\Delta x, \Delta y) \text{ for } \Delta x, \Delta y \geq 0$$

The dashed lines show the set of possible locations of the centerpoints of the white rectangle when it is at a distance d from the grey rectangle.

Figure 3.2: Three Different Metrics to Define a Distance

Because a geometrical representation is only an abstraction, it excludes much information. Therefore, when used in knowledge-based systems, it must join with other forms of knowledge representation that capture missing data. Because geometrical representation is so restricted in data content, it may not generally be the right choice to underlie a system that must later annotated with other data. As opposed to such "decorated geometry" (a word coined by Smithers in [Smithers 89]), a knowledge-based system must flexibly allow changing representations. Among others, Howard made this same criticism of CAD systems used as design tools rather than drafting tools. Evolving design tools will model the structure as well as the behavior of a component symbolically, using geometry as just one attribute of each component [Howard 89].



Drawing by Stevenson; © 1976 The New Yorker Magazine, Inc.

Figure 3.3: The Communication Problem

3.3 Design Problem-Solving Methods

Arrangement problems are inherently combinatorial. Their complexity depends on the number and type of objects to be positioned, and on the number and type of constraints to be met. Also, the number of possible arrangements generally increases rapidly as additional dimensions of space or time enter the problem statement. Furthermore, if the problem is one of routine design to the problem solver, it is usually easier to solve than one that requires more creative design. Thus, problem solvers will adopt different methods depending on the type of problem they face and on the knowledge available to them to apply the chosen method. A method can further entail several alternative strategies to reach a solution. Methods and strategies are often domain-independent (to some degree at least). It is therefore worth looking at research on layout design in various domains to extract those generic methods. From the available methods, I will then select one for use in SightPlan.

The following are problem-solving methods to solve design problems. Note that sentences marked with "(‡)" at the end are definitions taken from [Hayes-RothB 87], and that I quote only work that is illustrative or directly relevant to SightPlan.

- **SELECT AN ARRANGEMENT THAT SATISFIES THE CONSTRAINTS FROM A PRE-ENUMERATED SET OF ALTERNATIVES. (‡)**

Requires knowledge of: Alternative arrangements and selection criteria.
Example: A travel agent selects one of several tour "packages" that include all of the destinations requested by the client. A reader selects a book in the library. A construction manager selects a crane by picking one of those available is the yard.
Implementations: A data base of tour packages. A card catalog in a library. A company's equipment inventory list.

- **REFINE A PROTOTYPICAL ARRANGEMENT SO AS TO SATISFY THE CONSTRAINTS. (‡)**

Requires knowledge of: A prototypical arrangement and a refinement method.
Example: An architect refines a prototypical U-shaped kitchen design to include the special appliances requested by the client.
References: [Minsky 75], [Schank 77], and [Sowa 84] provide foundational ideas on "prototypes."
Implementations: VT [Markus 88] designs elevator systems. It uses plausible reasoning to construct an approximate design and successively refines it.

[Gero 88b] chunks knowledge about structural design into prototype elements.

- **MODIFY OR ADAPT AN ALMOST-CORRECT ARRANGEMENT TO SATISFY THE CONSTRAINTS. (‡)**

Requires knowledge of: Almost-correct arrangements and a method to modify them.

Example: A tool designer modifies an existing tool to fit a new machine. A carpenter modifies a formwork panel to fit the corner of a large wall.

- **GENERATE A COMPLETE ARRANGEMENT THAT SATISFIES THE CONSTRAINTS. (‡)**

Requires knowledge of: A procedure or algorithm for generating complete arrangements.

Example: A psychologist uses a multi-dimensional-scaling algorithm to generate a spatial model of subjects' similarity ratings of related concepts.

Issues: A generation procedure may guarantee that it will succeed, and in that case it is often called an algorithm. If a system that uses a generation method obtains multiple intermediate or final solutions, it will need some test function. This trade-off between how far a system develops a solution versus when it evaluates that solution is pervasive in design. A system can apply constraints both during generation or during evaluation. Shifting the locus of the application of constraints can affect both the efficiency and the solution quality in the design process.

Implementations: FOSPLAN [Yessios 71] performs space planning using a language to represent objects and constraints between them, and a grammar to join objects into composites. It will always reach a solution, but may not satisfy all conditions. HI-RISE [Maher 85] designs the preliminary structure of a high-rise building. The system hierarchically generates and tests the components it designs. [Dixon 84] performs the routine task of designing a standard V-belt drive.

- **IMPROVE OR REARRANGE AN ARRANGEMENT THAT ALREADY MEETS SOME CONSTRAINTS SO THAT IT BETTER MEETS THOSE OR ADDITIONAL CONSTRAINTS.**

Requires knowledge of: An initial arrangement that already consists of all desired constituents, a criterion to determine how to rearrange, and an evaluation function to decide when to stop.

Example: Children rearrange the chairs in a classroom so that they can all sit in one circle.

Issues: A system that uses an improvement method requires another design method to obtain its initial arrangement. It needs a criterion to determine which objects to rearrange (for example, switch positions of two or three objects that violate many constraints but occupy the same area), and an evaluation function to measure improvement (for example, compare the layout relationships with preferences expressed in a relationship diagram). Depending on the initial arrangement and the criterion for rearrangement, improvement systems may not converge to a good solution at all, or they may reach local optima only. Improvement methods become more cumbersome as the number of objects and constraints increases.

Implementations: CRAFT [Buffa 64], IMAGE [JohnsonT 71], and [Rodriguez-Ramos 82]

- **ASSEMBLE OR CONSTRUCT AN ARRANGEMENT THAT SATISFIES CONSTRAINTS. (‡)**

Requires knowledge of: a method for constructing arrangements, typically from parts that are predefined in shape and size.

Example: A person solves a jig-saw puzzle by placing pieces one at a time. A technician assembles a computer from components that were shipped separately.

Issues: The **constructive assembly method** generates a solution stepwise by determining a context to work in, selects an object and a constraint to satisfy, and computes for which locations the object meets the constraints. This results in one or multiple positions for the object. A system that uses this method may work on one or several arrangements at a time. One arrangement may be an alternative to another, or could be an elaboration of the other. Arrangements can be merged, and so on. Criteria for selection may be: random, based on size of the object, based on the efficiency of the computation needed to satisfy the constraint, or may be based on other heuristics derived from human expertise. They can be predefined or altered during problem solving.

In order to control the number of combinations that might be constructed, a construction system can be guided by a strategy. Such a strategy may vary for different applications based on the number of objects and constraints that are involved (for example, architectural layout is often limited to tens of objects, industrial site layout to tens or hundreds of objects, VLSI chip layout to many thousands objects). A strategy may dictate how to partition the problem into smaller ones, which objects to pick, etc. Another way of partitioning the problem is by separating the adjacency problem (topology) from the location and dimensioning problem (geometry). Although this is often taught in architectural layout design, there do not appear to be methods that easily make the transition between both phases.

Depending on the positioning actions they permit, construction systems must provide mechanism for truth maintenance, backtracking, and constraint propagation.

The early construction systems laid out single arrangements only. They relied on picking one object at a time, computing one or several positions, possibly checking those positions against constraints, and picking one position before proceeding with a following object. They followed an early-commitment approach. This necessitated backtracking and picking a second position when no positions could be found for subsequently placed objects. More recently, systems such as PROTEAN, LOOS, and SightPlan keep track of all locations of all objects at all times, thus providing the problem solver with the capability of pursuing a least-commitment approach. PROTEAN and SightPlan can also explicitly reason about several arrangements at a time.

Representations of objects vary from single rectangles, to composite rectangles, to polygons or other shapes, and objects in two or sometimes three dimensions. Constraint vocabularies typically are restricted to about ten geometrical or topological binary relations.

Implementations: Whitehead's system lays out single-story buildings [Whitehead 65]; CORELAP (Computerized Relationship Layout Planning) [Lee 67] and ALDEP (Automated Layout Design Program) [Seehof 67] lay out industrial facilities; GPS (General Space Planner) lays out mechanical rooms [Eastman 71, 72, 73]; DPS (Design Problem Solver) lays out equipment or furniture [Pfeffercorn 75a, 75b]; IMAGE lays out residential neighborhoods and fire stations [Weinzapfel 75]; R1 configures computer systems [McDermottJ 80, 81, 82; Bachant 84]; PROTEAN elucidates protein structure [Hayes-RothB 85b, 86; Brinkley 86; Buchanan 86]; LOOS lays out rooms [Flemming 88]; WRIGHT lays out kitchens [Baykan 87]; CONSITE [Hamiani 87] and SightPlan [Tommelein 87a, 87b] lay out temporary facilities on construction sites; and CADOO lays out ship propulsion compartments [Andre 86, 87]. Although the examples described in these references typically pertain to applications in a single domain, many of them can be used to construct layouts in other domains as well.

Constraint propagation systems might be classified as implementations of construction methods. They generalize the construction method used in the above-mentioned systems to deal with constraints involving many variables and not just geometric ones. CONSTRAINTS [Sussman 80] is a language for expressing almost-hierarchical descriptions; MOLGEN [Stefik 81a, 81b] uses constraint posting to represent interactions between subproblems; EDS (Edinburgh Designer System) [Popplestone 86] and [Chan 86] are both general frameworks for managing constraint-based design.

- **<u>TRANSFORM BY ANALOGY</u> AN ARRANGEMENT IN ONE DOMAIN TO AN ARRANGEMENT IN ANOTHER DOMAIN.**

Requires knowledge of: An analogous problem in another domain for which a solution arrangement is known.

Example: Consider the following spatial metaphor as the basis for analogical reasoning: "Go through the report and put together a summary that gets across the major points, along the lines of last year's." (from [Kautz 85])

Implementations: TRANALOGY (skill TRANsfer by anALOGY) [JohnsonMV 88] transfers PROTEAN's protein elucidation skills to site layout skills for SightPlan.

- **<u>INTERACTIVELY COOPERATE</u> ON DESIGNING AN ARRANGEMENT IN A JOINT HUMAN-MACHINE ENVIRONMENT.**

Requires knowledge of: Both humans and machines need design problem-solving knowledge.

Example: A customer instructs a bank teller machine to transfer money between two accounts. A designer uses a data-base query language to find the dimensions of a structural element.

Issues: Knowledge-guided search is common in interactive systems for non-routine design problems. There is no prescribed sequence of steps available to solve such problems, yet the types of steps are usually known ahead of problem-solving. Often, the knowledge of the cooperating agents is complementary; for example, humans provide control knowledge to guide the design steps taken by machines. Control knowledge may command changing representations, ending a generation step, and commencing analysis and evaluation.

Although interactive systems may lead to solutions inefficiently, users have tight control over the solution process and thus are directly responsible for the result. Interactive systems can provide powerful support tools not available otherwise.

Implementations: Interactive CORELAP [Moore 71] scores layouts interactively created by a layout engineer. This program was a forerunner in recognizing that users may want to "improve upon" machine-generated layouts. PALLADIO [BrownH 83] represents VLSI chips at several levels of abstraction and allows its user to switch perspective on the representation. PRIDE (Pinch Roll transport Interactive Design environment/Expert) [Mittal 86] designs paper-handling systems. BUILD [Rosenman 86] uses building regulations rules to generate a layout, and, conversely, checks if kitchens—laid out by using a graphical interactive interface—

meet building regulations. [Rad 82] implements a very simple interactive method for determining the size and location of temporary construction facilities.

There exist many more design problem-solving methods, such as model-based reasoning, case-based reasoning [Kunz 87; Slade 88; Goel 88]. Some work fits in more than one category, and some applications use several methods. Although the above categorization may be debatable, classifying problem-solving methods previously used for spatial layout applications helps to build a perspective on the generality of models for design. I chose the constructive assembly method for SightPlan because it needs no initial layout and literally can start from scratch.

3.4 Evaluation of Solutions

Assessing the quality of a solution requires some means for evaluation. This often is not an easy task. A solution must be viewed in the light of all simplifications and abstractions introduced to formulate the problem. It must integrate with the overall problem of which it may have been only a subproblem. Is the solution descriptive, prescriptive, or normative? Must its quality be assessed based on optimizing one criterion? Is a multiple-objective criterion a better measure for its quality? Is robustness over time important?

SightPlan's Expert Model descriptively models site layout field practice; I chose to address evaluation qualitatively and implicitly by using all constraints in a generative manner. That is, my strategy uses all constraints to construct the layout. This corresponds with field practice in which an operational site layout is usually considered satisfactory.

Clearly, any reasonable solution procedure for a design problem such as site layout needs to evaluate solutions during or after their generation. However, previous attempts to define criteria for evaluation were very limited in the number of factors they took into account (such as material flow, worker travel time). Thus, they assessed only one or a few aspects of the value of a layout. Despite the willingness to craft better means of analysis, the evaluation of all the dimensions of site performance (safety, line of sight, access, and so on) in a quantitatively rigorous way exceeds data gathering and computational resources with current technology.

Therefore, SightPlan's Computational Model illustrates how the program can use an evaluation function. In this model, SightPlan distinguishes hard from soft constraints by their use: hard constraints help construct the layout, soft constraints help evaluate the solution. Hard constraints are those that must be satisfied by a solution. Soft constraints preferably should be satisfied, but need not be. SightPlan's evaluation is a very simple function of distances between objects in the layout. This function, although it helps to select a single solution, is not a good measure to assess a layout's practical quality. Finding better ways to analyze and evaluate the quality of designed layouts for construction sites is an area in which substantial additional research is needed.

3.5 Layout Design in Specific Domains

Many application domains share issues they address: modeling design issues and problem-solving methods. The categorization in Section 3.3 suggested that methods successful in one domain can carry across and solve problems in another domain. These general methods, however, are typically *weak* problem solvers. That is, they can succeed in finding solutions, but may not do so efficiently. Conversely, *strong* methods are efficient but may not always apply. For instance, AI problem-solving methods often use general search techniques (which are weak methods) and address problems that algorithmic OR techniques (which use strong methods) could not solve. See, for example [O'Keefe 85] and [Van Hentenryk 88] for a comparative study between some AI and OR methods. Often, AI modeling efforts start by using a weak method and subsequently tailor heuristics to specialize the method for characteristics specific to the application domain. The following example domains—with problems that are similar at a first glance—show how researchers addressing specific problems focus on different problem characteristics to refine their solution methods and to gain problem-solving power.

- **BIN-PACKING**

Bin-packing illustrates the complexity of layout. The problem is to place each of a given set of predefined objects in the minimal number of bins. Objects can vary in size and number, and constrain each other by their geometry in that they cannot overlap. The bin-packing problem is NP-complete, yet there exist effective heuristics to solve the problem [Garey 79]. Many layout problems are similar to bin-packing. They must fit unequally sized rectangles in a given space—a problem that is hard enough on its own—but, in addition to geometrical constraints, many other constraints apply. Such constraints mostly relate to people or material movement between objects, and the timeliness of the placement. To deal with these constraints, simple bin-packing heuristics must be refined or replaced by others.

- **ARCHITECTURAL FLOOR PLAN LAYOUT**

Architectural layout consists of arranging rooms, subject to constraints. Design research often decomposes this problem into a topological step (arranging shapeless rooms according to relationships between them in a graph) followed by a geometrical step (transforming the shapeless rooms into rectangles, and the graph connections into adjacencies).

Past research explored ways of altering the graph so that it can guarantee planarity of the layout, systematically enumerating all topologically feasible and tightly-packed rectangular dissections, and incorporating circulation areas in the layouts [Whitehead 65; Flemming 78; Bloch 79; Baybars 82; Roth 85; Roth 88]. Current work proposes architectural representation at different abstraction levels [Galle 86], and graph representations to track all possible layouts during a least-commitment layout generation [Flemming 89].

Most architectural layout systems build on a rectangular representation of space. Section 3.2 argued that by doing so, they overconstrain themselves, and may not show sufficient flexibility to address issues related to layout. Very often architectural systems treat layout as an isolated problem, that is, they seem not integrated with building performance tools.

- **VLSI CHIP LAYOUT**

VLSI layout is a problem of a totally different scale than architectural layout, and thus approaches the problem differently. Brown compares the two problems: "laying out a 'typical' VLSI chip these days—chip 10 mm on a side, lambda down to 0.2μ—is equivalent to a town planner trying to design a town equal in area to the entire landmass of western Europe, all at urban street densities" [BrownA 88]. Solution methods must address this difference of scale, and graphical and computational support tools such as VLSI design-rule checkers are therefore omnipresent.

VLSI layout consists of laying out functional cells in an orthogonal grid on a silicon wafer; cells are rectangular and of standard or varying modular size. Layout is one of the last steps in the silicon compilation process. This process starts with a functional or behavioral description of the chip to be designed. Intermediate steps to form a chip are: construing a schematic-level description, describing the circuit at stick level, defining the geometry at switch-level, and finally, placing and routing.

Standard methods such as construction and improvement gain power by geometrically restricting the layout elements and by using sophisticated scoring functions (such as simulated annealing [Kirkpatrick 83]). A designer often chooses elements from a library of standard parts. Some conventions on a chip layout are: *Gate arrays* are rows of basic cells divided by routing channels, so they largely reduce the layout problem to one of routing. *Sea-of-gates* allow any area for placement of functions or routing, but their variable size functional cells and over-the-cell routing make automated layout difficult [Shragowitz 88a, 88b]. In this domain, heuristics for placement are closely tied to the resources needed in routing. Example objectives imposed on the design are minimum length of interconnections, or minimum area for the layout, depending, among other factors, on whether or not the chip is custom-designed or mass-produced. Because the placement and routing problems are tightly coupled, knowledge-based interactive environments for chip design are promising support tools [BrownH 83].

- **CONSTRUCTION SITE LAYOUT**

Construction layout is characteristic in that: 1) layouts are very loosely packed and objects can have any shape; 2) layouts as well as objects they arrange change in size and shape over time; and 3) each layout is unique. Circumstances (such as a downtown construction site) may require a tight packing of all facilities, but this is not a desirable situation because ultimately construction operations become inefficient due to temporal and spatial interference.

Although design methods span across domains, the above examples illustrate that knowledge about the domain varies widely, and imposes constraints on defining and positioning objects, and on solution layout evaluation.

3.6 Summary of Choices Made for SightPlan

SightPlan uses a very simple representation of 2-dimensional space to reason about spatial constraints that need to be satisfied between rectangular objects in a layout. The constructive assembly method assembles its solution arrangements. By construction, these arrangements satisfy the strategically selected constraints.

To implement SightPlan, I chose a general architecture for reasoning about action, called BB1. This architecture provides flexibility in accommodating not only the selected problem-solving method and spatial representation, but others as well. If SightPlan were thus to change or extend, the architecture would permit that. Chapter 4 gives more detail on the architecture and the implementation of SightPlan.

Chapter 4

The Layered Architecture of SightPlan and BB1

SightPlan is an expert system that assists construction managers in laying out the temporary facilities on a construction site. The system consists of knowledge specific to construction site layout, but it is layered on top of a domain-independent blackboard architecture named BB1, and uses information represented in other layers, such as the one specific to the class of problems solved by the constructive assembly method. Figure 4.1 shows the different layers of SightPlan: LISP is the programming language in which the entire system is implemented; BB1 is a domain-independent blackboard architecture; ACCORD is a language for use in the constructive assembly method; PROTEAN is an application that elucidates protein structures; GS is its 3-d geometry system; SightPlan is an application that lays out construction sites; GS2D is its 2-dimensional geometry system. Both PROTEAN and SightPlan make use of ACCORD.

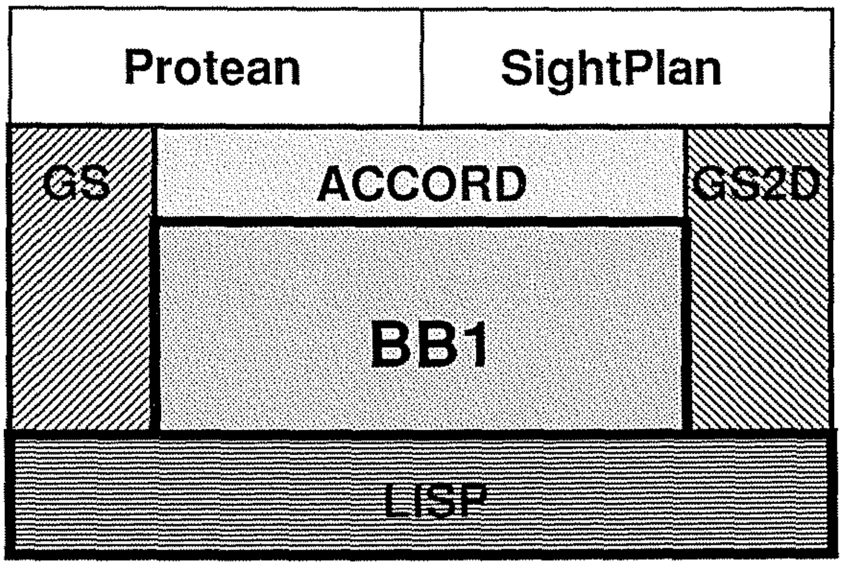


Figure 4.1: Layers of the SightPlan System

The BB1 blackboard architecture was conceived and developed by Dr. Barbara Hayes-Roth [Hayes-RothB 83, 84a, 84b, 85a] at Stanford University and has been implemented by Micheal Hewett, M. Vaughan Johnson and others involved in the BB1 project over several years [Garvey 87; Hewett 88a, 88b, 88c; JohnsonMV 87; Schulman 87]. The architecture originally was designed to address the class of assembly problems. It has since been used as a foundation for application systems spanning a wide range of domains. Applications to date include:

- PROTEAN, which elucidates protein molecule structures [Hayes-RothB 85b, 86; Brinkley 86; Buchanan 86],
- GUARDIAN, which monitors patients in the intensive-care unit of a hospital [Hayes-RothB 88d],
- WATCH, which inductively abstracts control knowledge [Gans 89; Confrey 89],
- and FIRST, which designs structural components by reasoning about previous cases [Daube 88, 89].

Other systems were applied to intelligent processing of materials [Pardee 87], tutorial instruction [Murray 88], aircraft tactical planning and control, living room layout, and the traveling salesman problem. These systems have in common that they use strategic reasoning while incrementally and opportunistically constructing a solution to their target problem.

I used version 2-1 of BB1 to develop SightPlan. This version is implemented in Common Lisp and runs on a Texas Instruments Explorer™. Let us now look at what the BB1 architecture provides and how an application system such as SightPlan builds on it.

4.1 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***) (Figure 4.2). None of the KSs can solve the entire problem on its own, but each may 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. (Those who are familiar with the concept of Object-Oriented Programming will see that, in this sense, BB1 is "anti-object-oriented programming" as the KSs cannot directly communicate with one another.) In each step towards the solution, one—and only one—KS gets to execute its proposed action to make certain 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*—which, 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 embodied by the scheduler, is both **incremental** and **opportunistic**.

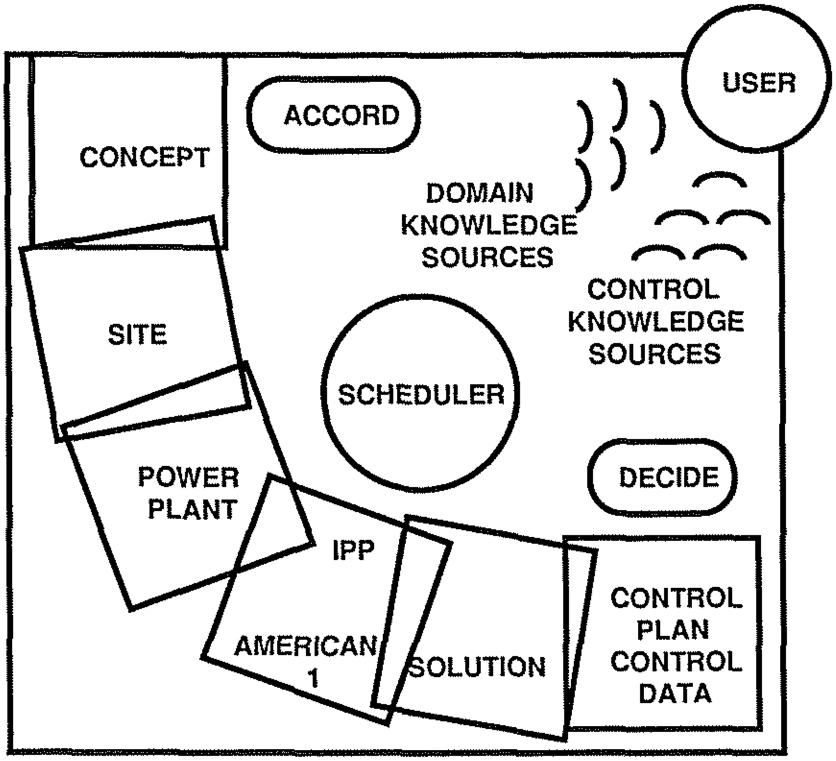


Figure 4.2: The Blackboard Metaphor

4.2 Conceptual Graph Representation

Concepts in the BB1/SightPlan world are represented in a conceptual graph [Sowa 84], which can capture everything needed to define and solve the problem. The underlying BB1 scheduler then makes the appropriate inferences. **Frames** represent concepts that can have any kind of user-defined attributes or links to other objects, and that can inherit attributes over specific links. Both the components of the architecture (such as blackboards or knowledge sources) and the physical objects in the application domain represented in the model (such as construction facilities or laydown areas) are part of this graph. Figure 4.3 shows the abstraction hierarchy of a prototype SightPlan system.

That is, it shows concepts representing *types* linked to other *types* by means of CAN-BE-A links, concepts representing *types* linked to *examples* by EXEMPLIFIED-BY links, and concepts representing *examples* linked to *instances* by INSTANTIATED-BY links. Figure 4.4 explains these links.

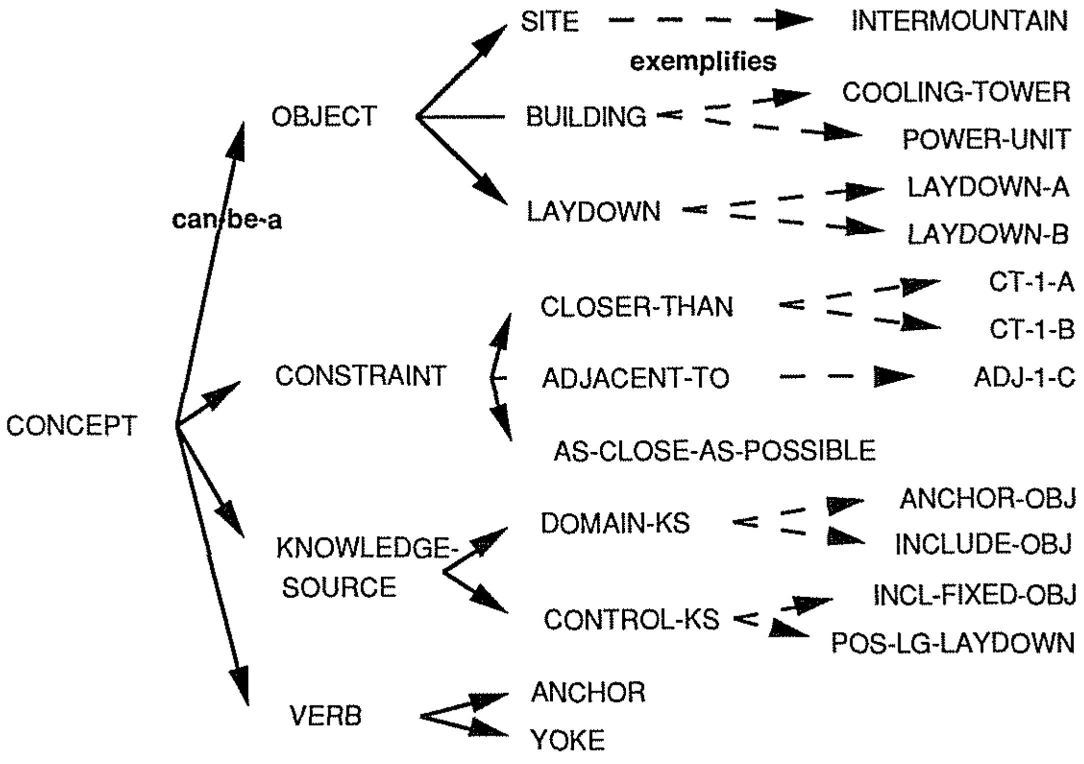


Figure 4.3: A Conceptual Graph of a Prototype SightPlan System Displayed as an Abstraction Hierarchy of Object Types and Examples with CAN-BE-A and EXEMPLIFIED-BY Links

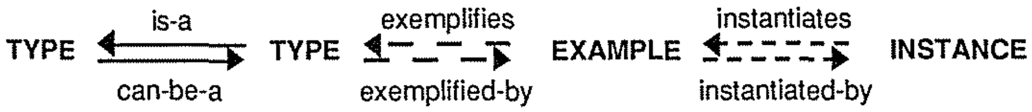


Figure 4.4: Links in the Abstraction Hierarchy

4.3 Multiple Layered Blackboards

Though the formalism of conceptual graphs can represent any entity, I have implemented only what is needed for SightPlan. 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***). Figure 4.5 gives a schema representing various layers and shows how they conceptually tie in with other. This figure illustrates how multiple application systems can co-exist in the same conceptual graph representation: PROTEAN is an application in the biochemistry domain. Intermountain and American 1 are two separate SightPlan implementations in the construction management domain. Dividing the system up in layers makes it easy to build a new application by substituting the needed BBs only.

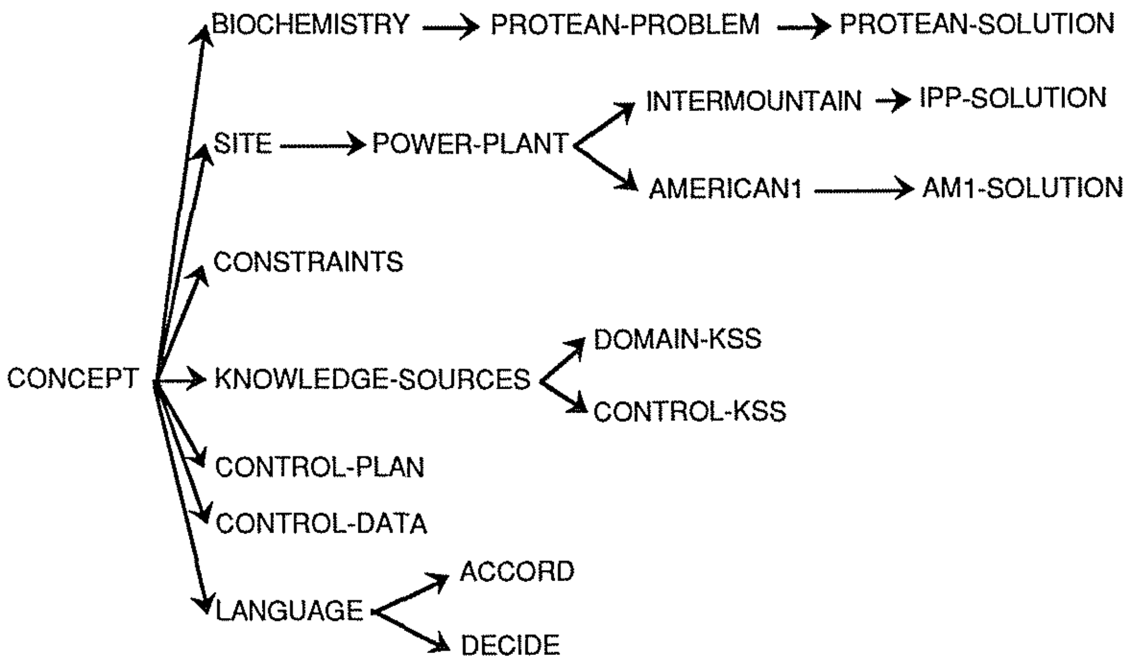


Figure 4.5: Schema of BBs Layering the Conceptual Graph
The labels represent BlackBoards in the SightPlan system.

The CONCEPT-BB groups the most abstract concepts from which domain-specific concepts stem. The SITE-BB groups those concepts defining objects, for instance those related to site layout and construction management, 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 *Intermountain Power Project*—are on the INTERMOUNTAIN-BB, those that exist on another site—for instance, the *American 1* project—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 CONCEPT-BB can be specialized to accommodate any representation of more specialized worlds.

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

4.3.1 Site BlackBoard

The SITE-BB contains 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 contains **adjectives** that field practitioners commonly use, such as: a "*wide* road" as opposed to a "*narrow* road," or a "*large* laydown area" as opposed to a "*small* laydown area." Concepts are classified by *level* to make the BB more comprehensive (Figure 4.6). For example, the PHYSICAL-OBJECTS level represents the physical objects *building, laydown*, and *road* with their relevant attributes and links (Figure 4.7 shows the *building* frame), whereas the MODIFIER level groups the adjectives (Figure 4.7 shows the *large* frame).

PHYSICAL-OBJECTS

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

MODIFIERS

LARGE
IMPORTANT
SMALL
EFFICIENT
LONG-TERM
PERMANENT

CONTEXTS

SITE
SUB-AREA

CONSTRAINTS

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

Figure 4.6: Excerpt of the SITE BB with Objects Ordered by Level

Note that the modifiers contain the attribute *function-definition*. When applied to the modified object (that is, the object linked to the modifier with link *modifies*), the function returns a numeric value.

Some objects included in the problem already may have a location in a given context before SightPlan's problem-solving starts. This is the case with the object POWER-UNIT-1 in Figure 4.7; the object has the attribute *location* whose value is the name of the site at which it is located (here, Delta, Utah is 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 4.7: Frames Representing a Physical-Object, Object Types, and a Modifier in the SightPlan Application

4.3.2 Constraint BlackBoard

The CONSTRAINT-BB represents concepts used as **spatial prepositions** or **adverbs** in common English (such as *adjacent-to* and *near*). Frames that name constraint types represent these concepts expressing unary, binary, or n-ary relations between objects (Figure 4.8). These types are exemplified by the actual example constraints that involve the related example-objects. The CONSTRAINTS level on the SITE-BB contained constraint types such as *closer-than*, *further-than*; each constraint type in turn labels its separate level on the CONSTRAINT-BB that groups the constraint's examples (Figure 4.9).

CLOSER-THAN

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

CLOSER-THAN-45-1

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 4.8: Frames Representing a Constraint Type and a Constraint Example

CLOSER-THAN

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

IN-ZONE

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

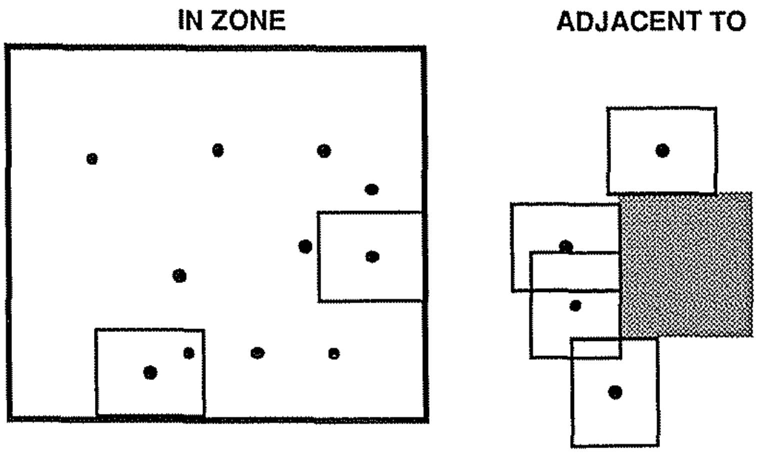
AS-CLOSE-AS-POSSIBLE

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

Figure 4.9: Excerpt from the CONSTRAINT-BB Displayed by Levels

Since SightPlan's scope is restricted to spatial layout, the system deals only with geometrical and topological constraints. Figure 4.10 graphically represents some of the constraint types implemented in SightPlan.

Whereas SightPlan reasons about the selection of objects to position and about the constraints that need to be satisfied, it relies upon a separate *constraint engine* to actually process the constraint. That is, SightPlan passes the sets of possible locations of each of the involved objects to that engine, and receives those sets after the constraint engine has reduced them to satisfy the constraints.



LEGEND

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

Figure 4.10: Graphical Representation of SightPlan Constraint Types

In defining each constraint type, a function takes on 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. Several assumptions limit the computation effort required by SightPlan's constraint engine; these assumptions are discussed in Section 4.5.

4.3.3 Knowledge Source BlackBoards

Knowledge sources in BB1 are *if-then rules*, so the Knowledge Sources Blackboard contains the rules that BB1 will apply to make its inferences from the current state of the BBs. These rules, however, are not designed to "chain" together as they are in traditional 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: **triggerconditions**, **preconditions**, and **obviationconditions**. *Triggerconditions* state when the KS becomes applicable. In addition, since the conditions of KSs refer to concept types, the KS uses *contextvars* to specify which examples in the specific domain of application apply. So, when more than one example of the concept type applies, multiple **knowledge source activation**

records (*KSARs*) are generated, one for each example, as shown in Figure 4.11. *Preconditions* state when the KS is executable, and *obviationconditions* state when the KSAR no longer applies.

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 4.11: Instantiation of Multiple KSARs from one KS

The then-part of a KS tells BB1 what changes to make to BBs when the KS in question executes. Although multiple KSARs may compete for execution at any time, the BB1 scheduler picks only one at a time. The different if-parts and then-parts of a knowledge source are shown in Figure 4.12.

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	OBVIATIONCONDITIONS	say when a KSAR should no longer be considered
THEN	ACTIONS	

Figure 4.12: Structure of a Knowledge Source in BB1

BB1 is an **incremental** problem-solver; actions execute one at a time. An **action's** execution *causes* **events** of adding or modifying objects on BBs. Because an object was changed, a new **state** is *promoted* (Figure 4.13). By definition, triggerconditions depend on events, preconditions depend on states, and obviationconditions (usually) depend on states.

Figure 4.13: Links Relating Actions, Events, and States

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

BB1 distinguishes two types of knowledge sources: domain knowledge sources and control knowledge sources.

4.3.3.1 Domain Knowledge Sources

Domain Knowledge Sources are application-dependent and specific to the problem-solving method that is used. In SightPlan, for instance, 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.

Figure 4.14 illustrates a domain KS, and Figure 4.15 gives the type hierarchy of domain KSs on the SIGHTPLAN-KS-BB.

For expressing a desired action, Domain KSs use the ACCORD language. ACCORD will be discussed in more detail in Section 4.3.4 on Languages in BB1. A domain KS's action makes changes to the SOLUTION-BB. Figure 4.16 illustrates how the domain KSs are part of the meeting room setting, and how they operate on the SOLUTION-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 4.14: Example of a Domain Knowledge Source with some of its Attributes and Links

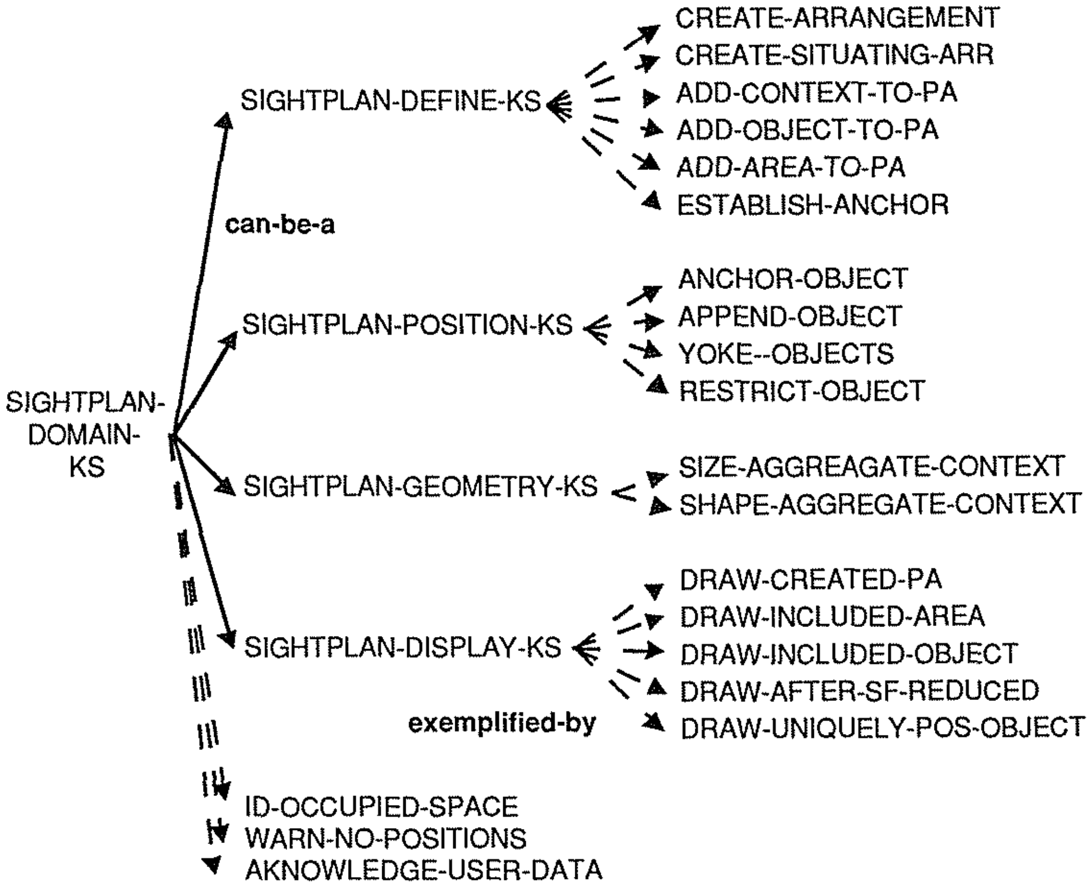


Figure 4.15: Type Hierarchy of Domain KSs on the SIGHTPLAN-KS BB

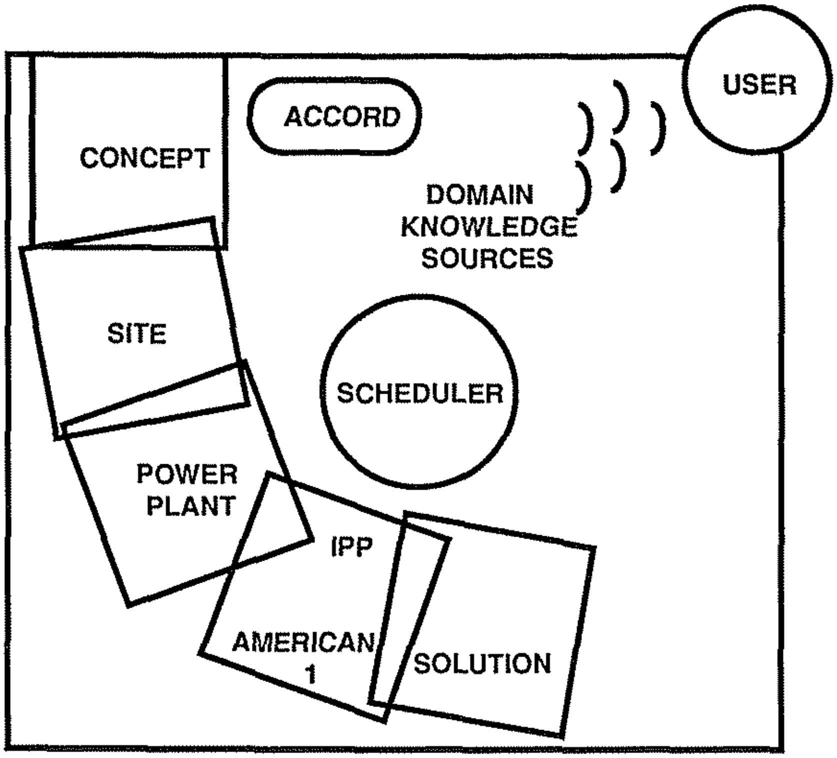


Figure 4.16: Cooperation of the Domain Knowledge Sources in BB1

4.3.3.2 Control Knowledge Sources

Control knowledge sources contain so-called *metaknowledge*, which takes the form of BB information that will allow the 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. Figure 4.17 illustrates how the control KSs are part of the meeting room setting, and how they operate on the CONTROL-PLAN-BB.

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 4.18 displays a control plan part way through 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. The scheduler uses foci to determine which of the executable KSARs is most desirable at that cycle. *Heuristics,* which implement foci, prescribe what function should be used by the scheduler to compute this desirability. I will return to the question of how the scheduler determines desirability when I describe the rating mechanism in the BB1 control loop.

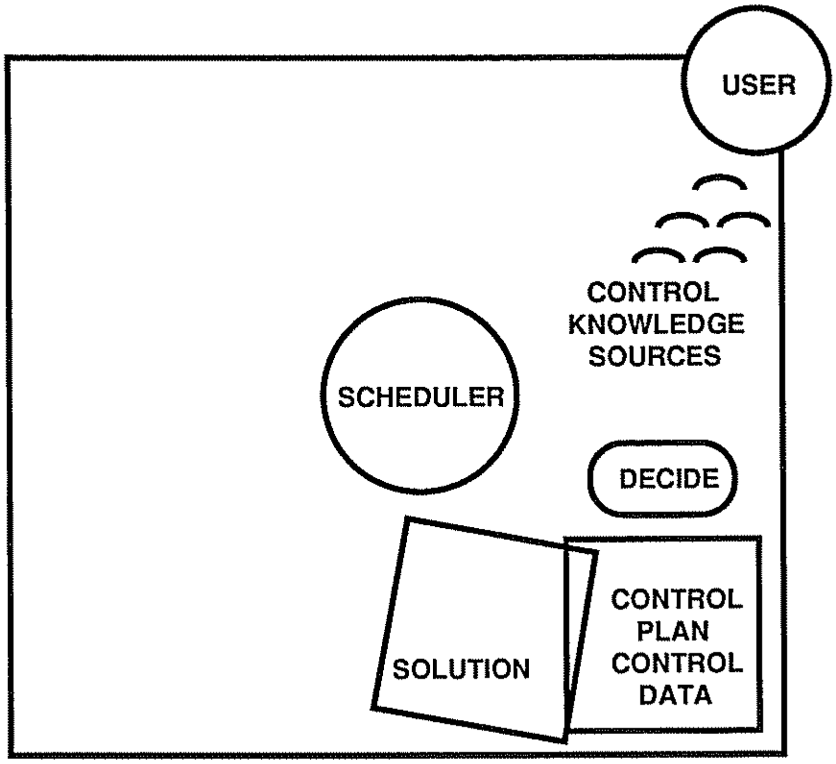


Figure 4.17: Cooperation of the Control Knowledge Sources in BB1

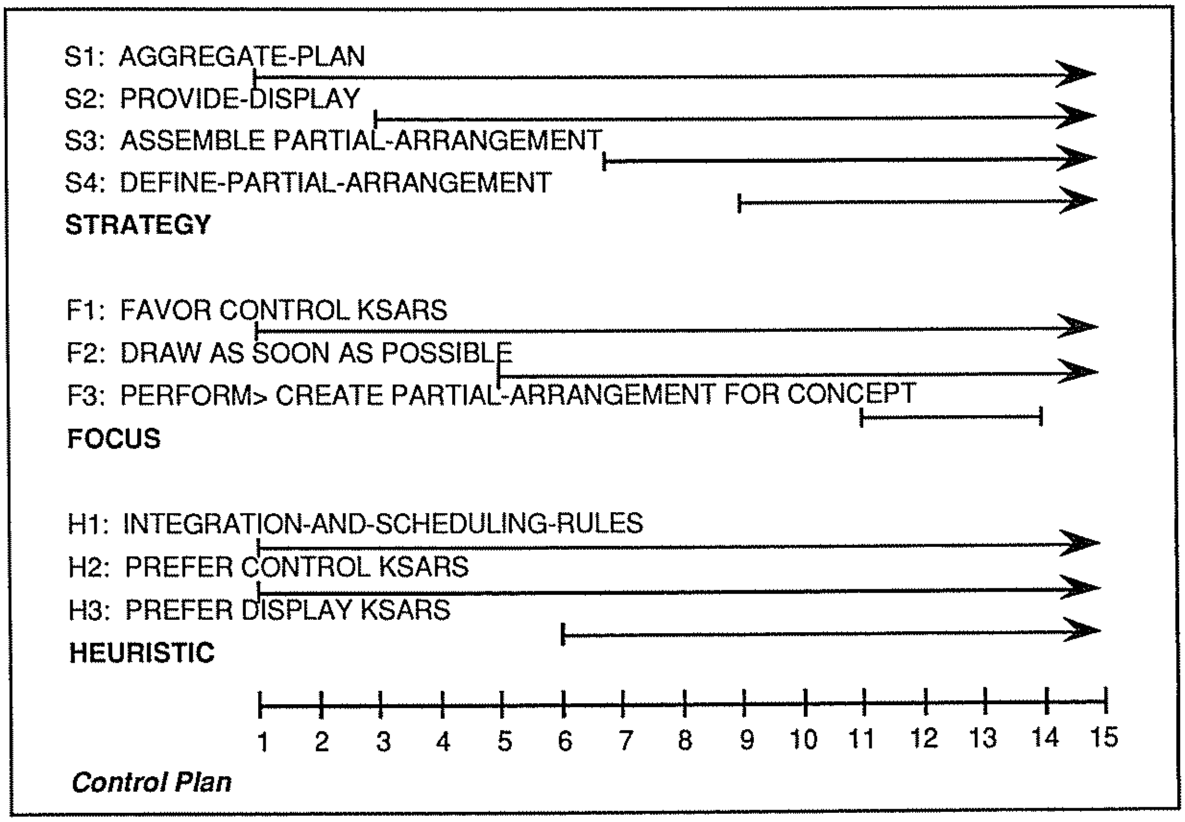


Figure 4.18: Control-Plan BB Displayed by Cycle

Figure 4.19 shows an example Control KS. Control KSs use their own language, named DECIDE, to express the types of actions to take. This language will be mentioned again in Section 4.3.4 on Languages in BB1.

```
                          POSITION-ALL-OBJECTS
level: SIGHTPLAN-KS.CONTROL
attributes:
triggerconditions:  (($EVENT-LEVEL-IS CONTROL-PLAN.STRATEGY)
                   ($EVENT-TYPE-IS MODIFY)
                   ($CHANGED-ATTRIBTUTE-IS CURRENT-PRESCRIPTION)
                   (MEMBER 'POSITION-ALL-OBJECTS
                        ($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 4.19: Example of a Control Knowledge Source
with some of its Attributes and Links

All KSs in BB1 are treated in the same manner: as individual independent units that compete for execution. Control KSs have the same format as Domain KSs (compare Figure 4.19 and Figure 4.14), so they first trigger, then generate one or more KSARs, and finally, a KSAR becomes executable before its actions can be executed. Sometimes, though, an application designer may know in advance that a certain sequence of control actions will need 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**. Examples of skeletal plans are shown in the description of the SightPlan models, Figures 6.1, 6.2, 6.25, 6.31, and 6.41. A skeletal plan is a tree-hierarchy of concepts that prescribe strategies, foci, and heuristics. These concepts have an attribute "strategic-generator" which pre-specifies 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. Furthermore, other domain-independent Control KSs can be defined in that layer; for example [JohnsonMV 87] created KSs that allow a strategy to recommend goal-directed reasoning.

Figure 4.2 illustrated how control and domain KSs are part of the same competitive environment. Thanks to control knowledge sources, a BB1 application can dynamically alter its strategy and opportunistically select its actions.

4.3.4 Language BlackBoards

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

ACCORD, AN APPLICATION LANGUAGE FOR CONSTRUCTIVE ASSEMBLY

Figure 4.20 shows the type hierarchy of ACCORD.

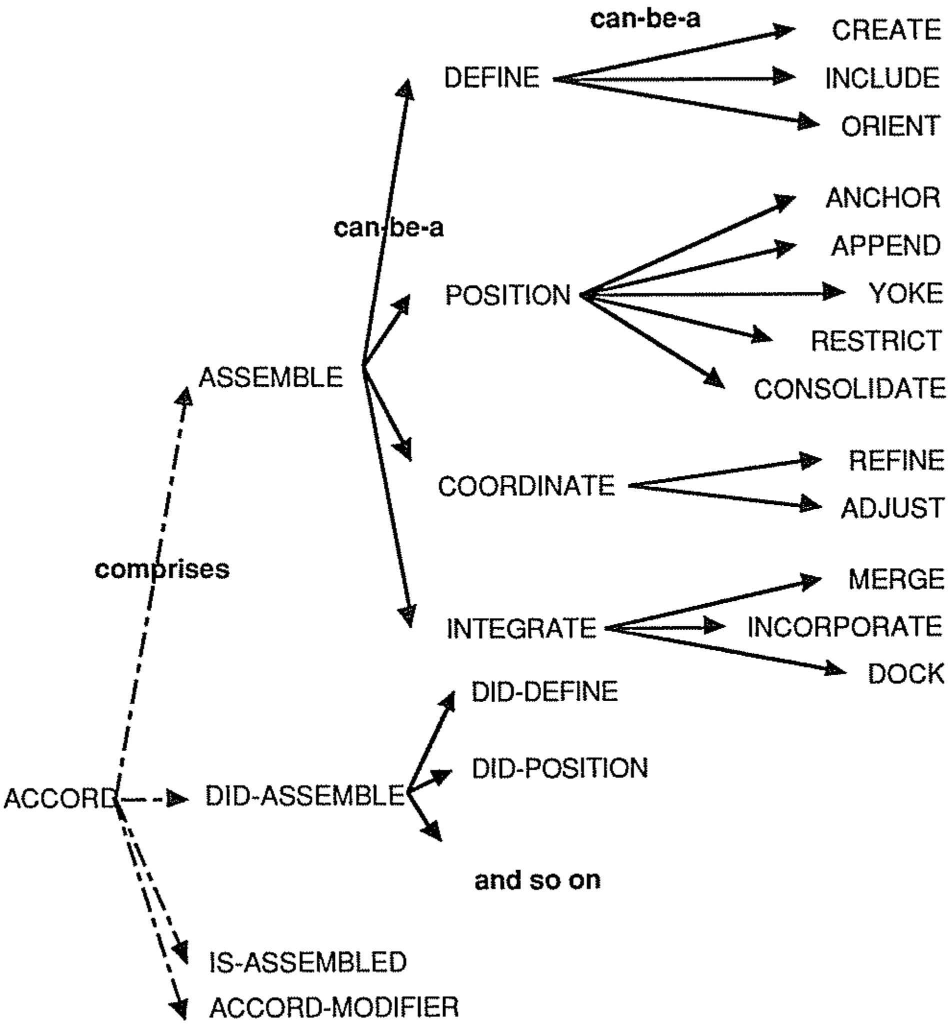


Figure 4.20: Type Hierarchy of ACCORD

The ACCORD language permits KSs to express their triggerconditions, preconditions, or obviationconditions in a vocabulary established for application of the constructive assembly method. The action of a KS consists of adding or modifying object(s) on BB(s), and can be expressed with calls to the native BB1 low-level functions (such as $ADD and $MODIFY). These descriptions, however, are rather cumbersome to write or read by people who author or use KSs. Also, because 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-RothB 88b]. Another advantage of using a language is that it would make the matching between desired action types (foci) and possible actions (executable KSARs) very easy (as I will show in Section 4.4.2). Besides providing a vocabulary to express actions, ACCORD encompasses vocabulary to express *caused events* and *promoted states*.

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 4.21: Definition of the ACCORD Action "ANCHOR"

The language on the ACCORD-BB predefines all possible verbs to express Actions, Events, and States. 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. Action templates and their low-level translations share variables, namely those that are marked with a "@" preceding their name as specified in the template (such as @anchor and @anchoree). Thanks to such definitions, KSs can use the template without having to worry about the translation, which is taken care of by a language translator. Figure 4.14 showed how a knowledge source prescribes the ANCHOR action, and Figure 4.21 shows how that action is defined and translated into native BB1 low-level functions. PROTEAN was the first user of ACCORD, but this language needed to be further implemented to accommodate SightPlan's actions.

DECIDE, AN APPLICATION 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 4.22 shows the type hierarchy of DECIDE.

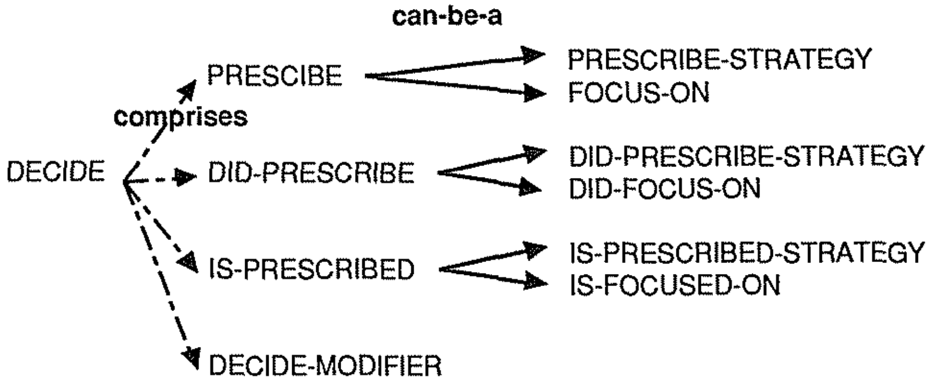


Figure 4.22: Type Hierarchy of DECIDE

4.3.5 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 splits 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 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.

4.4 BB1 Execution Cycle

4.4.1 BB1 Control Loop

The BB1 *scheduler* goes through the **execution cycle** described in Figure 4.23. A cycle starts with the execution of one action, which causes events to happen. The system tries each event of the current cycle to trigger any of the KSs. For each triggered KS, the scheduler generates the example objects in the context of the run and generates appropriate KSARs. All of the KSARs—whether just generated or pre-existing from previous cycles—have their preconditions and obviationconditions checked. If preconditions are found to be true and obviationconditions are not 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 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 4.23: Structure of an Execution Cycle in BB1

4.4.2 Rating Mechanism

At any cycle there can be multiple executable KSARs. This is desirable because, as one design feature of the blackboard architecture, the scheduler can decide which of the competing alternatives 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 receives a numeric rating 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 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 uses an application language, establishing a rating proceeds somewhat differently. In that case no heuristic implements the focus, but, instead, a matching scheme is implied by the language. For example, Figure 4.24 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.

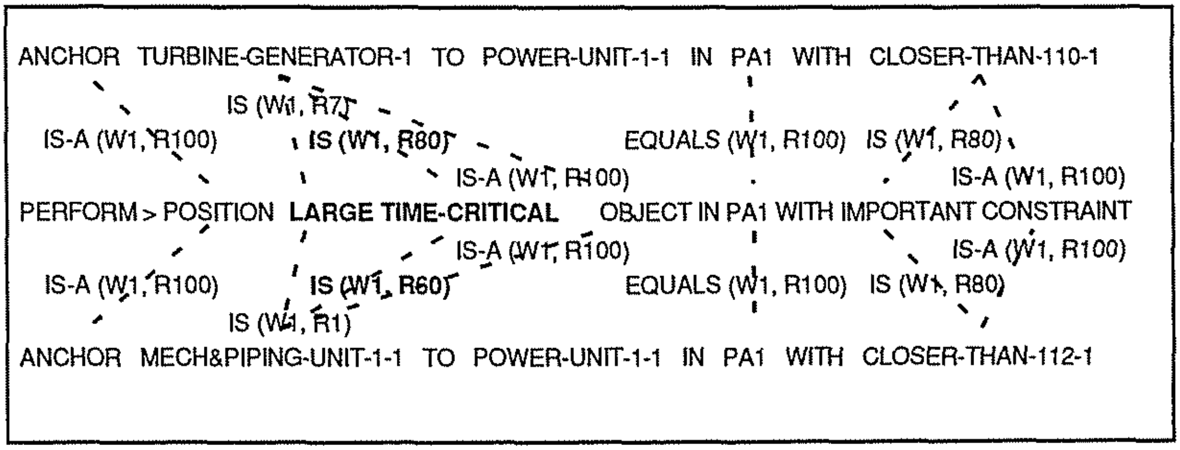


Figure 4.24: Rating Mechanism explained by ExAct [Schulman 87]

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 the work of [JohnsonMV 87] on goal-directed reasoning. SightPlan only uses rating based on action sentences.

4.5 GS2D Constraint Engine

The constraint engine used in SightPlan, named **GS2D** for "Geometry System—2 Dimensional," is domain-independent. It was developed by Tony Confrey and François Daube [Confrey 88], who used many of the ideas of PROTEAN's three-dimensional constraint engine, called **GS** [Brinkley 86]. As a simplification of GS, GS2D allowed the user to reason about rectangles in two-dimensional space. GS2D's computation on rectangles with 2.5 degrees of freedom is sufficiently simple for GS2D to run efficiently in Commonlisp on the Texas Instruments Explorer™, whereas GS' computation on objects with six degrees of freedom had to be written in C to run overnight on a Silicon Graphics Iris™ workstation.

GS2D takes as its input sets of possible locations of objects and a constraint to be met between those objects, and it reduces those sets to output only those locations where the objects meet the imposed constraint. In that way, impossible positions are excluded, but any location where an object might meet the constraint for a possible location of the other object(s) is maintained. In that way, GS2D is capable of supporting a least-commitment strategy for SightPlan to pursue.

Given the specifications imposed on GS2D that 1) only single rectangular entities are to be represented, and that 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 was chosen to be represented by the entity's so-called **essential area**. Figure 4.25 shows the convention on the representation of an essential area, and Figure 4.26 is an example of how a set of possible locations for a rectangle is represented.

```
((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 4.25: Convention on the Representation of an Essential Area

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

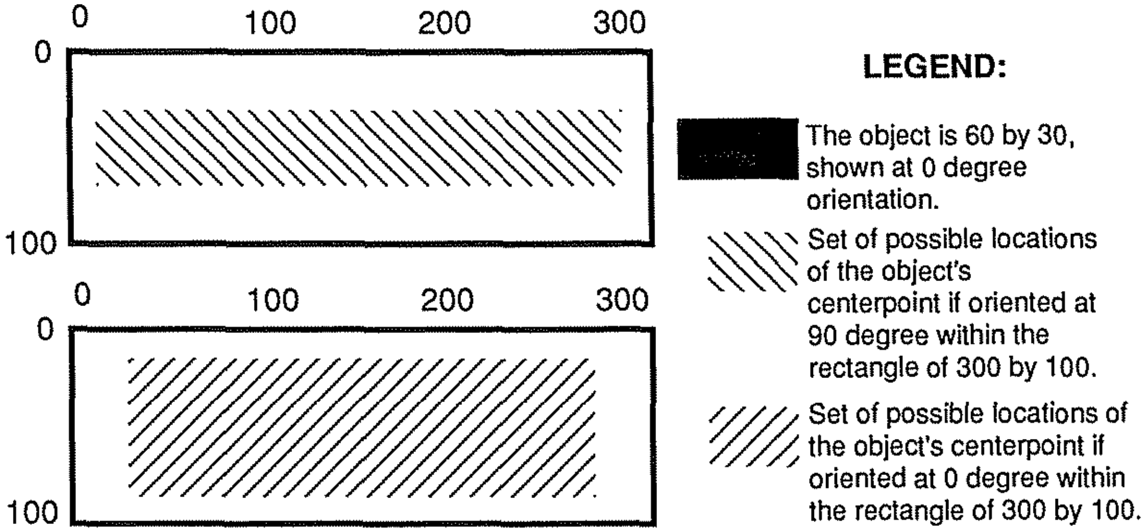


Figure 4.26: Example of an Essential Area of a Rectangle and its Graphical Display
The rectangle has dimensions 60 by 30 and was positioned
in an area of dimensions 300 by 100.

The essential area constitutes a comprehensive—though not the most concise—description for multiple locations of an entity. A single position of an entity could, of course, be represented by three numbers only (3 degrees of freedom in 2-d space, when the entity's dimensions are given). However, because GS2D has to deal with multiple positions for each entity most of the time, the essential area representation was adopted and provides a clear notation while it carries only a small overhead.

Figure 4.27 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, according to the design of GS2D, distances between entities are defined in terms of the minimum orthogonal distance between the edges of objects, and this may be counter-intuitive. As I have argued in Chapter 3, each simple definition of distance carries some disadvantage of this kind. Figure 3.3 c illustrated the definition of distance with an example.

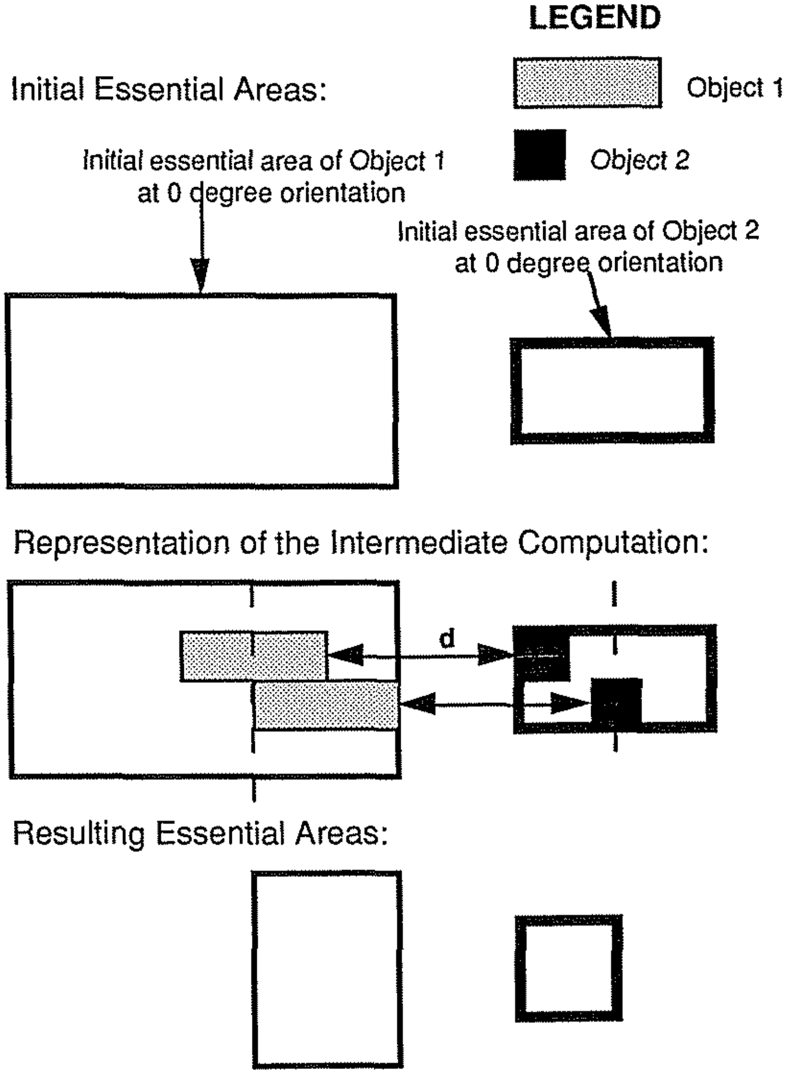


Figure 4.27: Application of the CLOSER-THAN Constraint to Two Objects

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—possibly faster—constraint engine if it needed to.

4.6 Graphical Display Packages

4.6.1 SightPlan—*Sight* after all!

The reader may have wondered why SightPlan is not spelled *SitePlan*, since, after all, it is a system that generates *site* layouts. We arrived at our spelling soon after we started working on a prototype system to do site layout. M. Vaughan Johnson launched the idea of using bitmaps for both the graphical display and the data structure to keep track of essential areas. Though some of the computations we wanted to perform turned out to be too time-consuming and we gave up on that idea, the need for graphical display of intermediate and final results of the system became blatantly obvious.

Whence, **To SEE —> Sight** and **To PLAN —> Plan**

The need for display did not stem from SightPlan itself, as the program's reasoning is totally independent from display, nor did it emerge from the constraint engine, as GS2D makes use of a numerical data-structure for computation on essential areas. It did become apparent while I was debugging SightPlan: I needed some easy way to check whether GS2D's computations returned the expected results, and the numerical representations of possible sets of locations were often too cumbersome to decipher. Visualization of sets of objects by means of dashed rectangles on a display of the site was extremely useful, and of course it made the SightPlan model more closely approximate what people do when they sketch partial layouts. SightPlan's graphical output is used throughout Chapter 6 to illustrate intermediate and solution layouts.

Since SightPlan may want to reason about when to display objects and in what manner to display them, objects in the skeletal plan and domain KSs were added to the system (Figure 6.1.2). Of course, these KSs had to compete for execution with other KSs of the system, so the total number of reasoning cycles required for problem solving increased. A remedy to this increased complexity was to do the display remotely and to have a separate processor worry about it. We designed a remote display system, named SightView, for this task.

4.6.2 SightView

SightView was to run on a computer with a color monitor so that the graphics could be shown in color. Using MPW Pascal, Tony Confrey implemented the graphical system. He interfaced it with Bill Yeager's C code, needed for the TCP communication between the Macintosh II™ and the Explorer™. The setup of one machine running SightPlan and the other running SightView is shown in Figure 4.28.

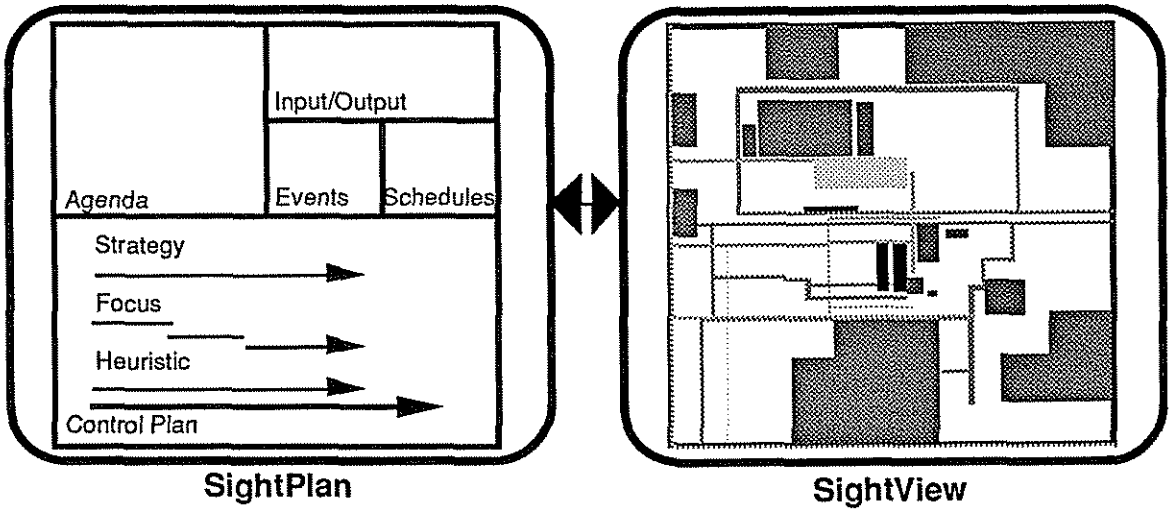


Figure 4.28: SightPlan, the expert system for construction site layout, Interfacing with SightView, the remote graphical display.

SightView receives from SightPlan an object's name, its essential area, and a color code. It displays either a rectangular shape for cases in which the essential area consists of a single point location, or the set of hollow rectangles for all other cases. Using SightView's menu bar and pop-up menus, a user can choose to selectively display only objects at point locations, only objects with sets of locations, or can remove objects from the screen. In addition to this feature SightView has **interactive graphics**. A user may decide to demarcate a sub-set of possible locations or may pick a single position out of an essential area. SightView then sends that reduced essential area back to SightPlan, which has KSs dedicated to acknowledge such external input. SightPlan can choose to incorporate this information in its knowledge base and to perform further reasoning about it. In this way, a user of SightView effectively acts as another knowledge source in SightPlan.

4.7 BB1 Communication Interface

To facilitate the exchange of information between SightPlan and SightView, the system uses the Communication Interface (*CI*) [Hewett 88b] (*Figure* 4.29). The CI package allows the BB1 architecture to interact with remote processors for either the input or the output of its information in such a way that the BB1 execution cycle will suffer the least from interruption. The CI can run on its own processor, and it routes incoming information over the appropriate connections.

In this setup, I removed those KSs in SightPlan that were needed to reason about when to display objects (note that I could have done this while using SightView without the CI as well). When blackboard changes occur, the system sends off those changes concurrently (in the sense of "in the same action") to the CI, which routes it to SightView. The user of SightView then decides what to display and when. Eventually, SightView itself could be extended to be more intelligent about this. In this way, the three cognitive tasks performed by the joint system—reasoning, communication, and perception—are distributed over SightPlan, the CI, and SightView (Figure 4.30).

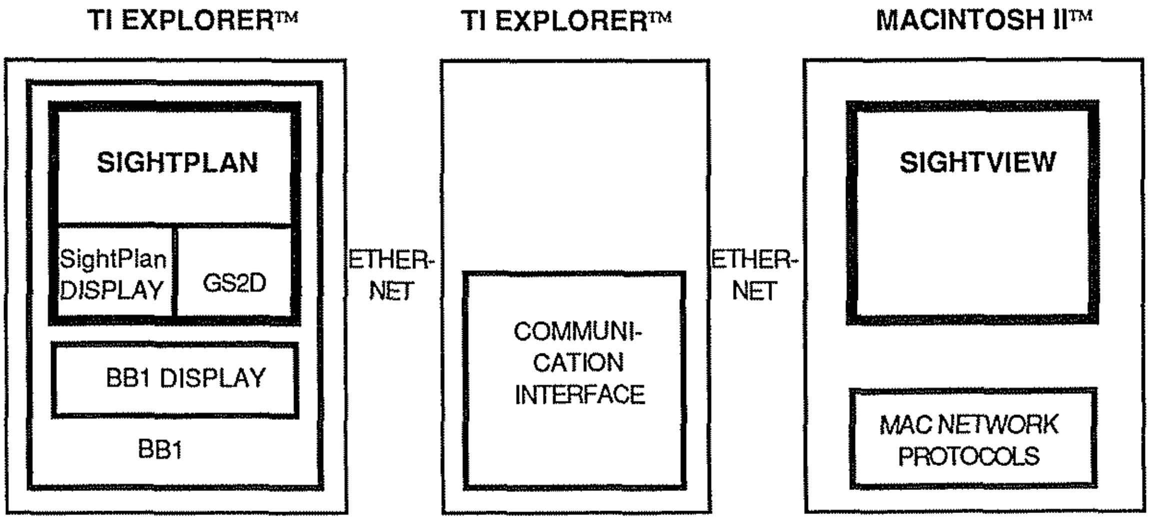


Figure 4.29: The Communication Interface
(Figure modified from one by Tony Confrey)

REASONING <—> COMMUNICATION <—> PERCEPTION

SightPlan CI SightView

Figure 4.30: Distributed Processing of Reasoning, Communication, and Perception

4.8 Summary and Motivation

BB1 is a domain-independent blackboard architecture that encourages reasoning for incremental and opportunistic problem-solving. Some of its main design strengths are that it can:

- Reason about what actions to pursue next.
- Explain how it selected one action over another.
- Dynamically alter ITS control plan.
- Provide for a layered environment with frame-based representation.
- Make available generic blackboards and knowledge sources.
- Make explicit what strategic information is applied to select preferred actions.
- Communicate with other agents in the world.

BB1 was chosen for the development of SightPlan for several reasons. First, with its powerful blackboard metaphor and conceptual graph representation, BB1 provides a versatile environment for implementing application systems. Since the SightPlan model captures the strategies field practitioners use for site layout, a mechanism for making explicit that type of metaknowledge was required, and BB1 provided control KSs to satisfy this requirement. I also wanted a frame-based representation for domain knowledge, and BB1 relies on that representation. Second, BB1 was available. I had access to its source code and to implementations of other application programs from which I could reuse ideas as well as code. This allowed me to quickly craft a first prototype of SightPlan by adapting an existing one of PROTEAN with the help of a BB1 group member. Finally, and maybe the best reason for using BB1, is that I obtained encouraging support and enormously helpful feedback from the designers of the system, who were also interested in seeing BB1 tested on a real-size application.

Though BB1 may not have been the only environment in which I could have implemented SightPlan, it has definitely been a good choice. For additional reading on other blackboard systems, see [Nii 86a, 86b].

4.9 Implementation of an Application in BB1

Having described the different pieces of knowledge and information that embody an application in BB1, what remains to be done is to select from existing systems those generic layers that also apply to SightPlan, and to delete the others while adding in new layers that pertain specifically to site layout. In particular, the next chapter will introduce two case studies that provided project-specific information on the Intermountain power plant and on the American-1 power plant, stored on the INTERMOUNTAIN-PROBLEM-BB and on the AMERICAN1-PROBLEM-BB respectively. Objects on these problem-BBs are examples of those on the SITE-BB. More knowledge about the layout of power-plant sites will then be represented on the KS-BB, where a fairly general pool of domain KSs for SightPlan awaits triggering, and where the control-KSs tailored to describe two different solution strategies will post strategic criteria. What these strategies are, how they were derived, and how they affect problem solving is discussed in Chapters 5 and 6.

Chapter 5

Case Studies

The knowledge posted on the various SightPlan BlackBoards was modeled after case studies performed on two projects. As mentioned in Chapter 2, SightPlan focuses on site layout for industrial construction, more specifically on coal-fired power plants, for several reasons. First, coal-fired power-plant construction is well-established in construction practice. Second, I had obtained field manuals with guidelines to direct this type of layout. Finally, after identifying the projects I wanted to study, I was encouraged by finding managers of each of the parties involved on the projects willing to cooperate, so I would learn about their field experience. This chapter briefly describes the two projects studied for SightPlan, discusses knowledge acquisition, and gives a summary of how each site was laid out. The first project is the Intermountain Power Project located in Delta, Utah; the second is the American 1 power plant located in King City, California.

5.1 Methodology for Project Selection and Site Interviews

The first project would constitute the major case study for SightPlan. I selected the Intermountain Power Project (from now on referred to as *IPP*) by starting with an extensive search for power plants under construction in the US. [Budwani 85], [Edison 84], [Smock 87], and [Power Engineering 87] provided lists of plants at various stages of completion, from which I was able to select several that met my criteria (600 to 800 megawatts range, coal-fired, under construction or recently completed). For each of these I tried to contact the owners (often a utility), architect-engineers, and construction managers who had been involved with the plant's construction. At the same time, I asked them to suggest a project for my study—if they could think of one more suitable than the one I inquired about. In some preliminary interviews I conducted at home offices of construction engineers (Dillingham in Pleasanton, California; Bechtel in San Francisco and Los Angeles, California; and Stone & Webster in Boston, Massachusetts), I learned, among other things, about construction field manuals and special rigging studies.

After these interviews, my search was narrowed to only a few projects. I tried to locate the person responsible for the construction site layout for these projects—not an easy task, as it turned out, because laying out the site is often only a secondary assignment for one of the managers in the field—and found out if he or she would be willing to collaborate. By coincidence, IPP's Design Mechanical Engineer at the project managers' home office had kept a site arrangement drawing that highlighted areas for long-term laydown areas for contractors; he in turn referred me to the Lead Mechanical Coordinator on site who had made that drawing but had not kept a personal copy. Thanks to the cooperation of these two engineers, I obtained access to all the information I needed. I had many telephone conversations, made several trips to the project manager's home office, and spent three days at the IPP site and two days in the architects/engineer's home office to learn about the project. Section 5.2 will detail the results of this investigation.

The second project would constitute a case study to validate the first model of SightPlan since it had been based on IPP. I selected American 1 for several reasons. First, a project with smaller scope than IPP would avoid extensive interviewing sessions. Second, smaller power plants—mostly co-generation plants of 100 megawatts capacity—were much more accessible, in that many more of them are under construction and some are located relatively close to Stanford. (At an early stage in my research I had spent time on the site of Stanford's co-generation plant, then under construction.) Third, testing SightPlan on a power plant site of different scope and type would demonstrate the degree of generality of its strategy. Finally, the Field Construction Manager at IPP had worked on two co-generation plants after completing IPP, and he referred me to the Construction Manager at the American 1 plant. I spent a day on site and again obtained much valuable information. Section 5.3 will describe American 1 in detail.

My goal in this phase of knowledge acquisition was to capture expertise of two kinds. First, how do field practitioners go about selecting temporary facilities needed on a specific project and how do they size them? In order to restrict the problem's complexity, I focused on temporary building structures and on long-term laydown areas. Second, I hoped to learn what strategy the practitioners used to allocate space for those facilities.

Before conducting interviews, I prepared by obtaining site arrangement drawings, construction schedules, and any other available documents related to site layout. This acquainted me with the project, helped identify what temporary facilities had been allocated, and yielded a list of questions regarding the arrangement. On my interviewing trips I took paper and pencil along as well as an audio tape recorder to record—when possible—all my conversations with office and site personnel. This proved to be most valuable because many issues became clearer when I heard them a second time on tape. On both sites I was given an extensive tour of the plant facilities by my main contact person, and I got permission to tour the facilities on my own and to take pictures.

The conversations on which facilities to locate and where to locate them shed light on the issues, but I found that the best way to learn about site layout was to talk over a site arrangement drawing and to have my field expert mark up the drawing as we proceeded in our conversation. That process not only made explicit the design steps taken to generate a layout; it also obliged the expert to articulate the sizes of the located facilities and provided a means to verify whether or not the layout produced in the course of the conversation did correspond with what had been implemented on site. This approach worked because both projects were near completion by the time I got to their site, and hence, all considerations mentioned by the people I talked to were necessarily after the fact. It is, of course, quite likely that in describing how the layout was generated the expert made the process appear more rational than it had been in fact. Hindsight makes one forget the many minor problems and solutions that arise in any complex project.

IPP provided the chance to transcribe and summarize my recordings overnight so that, the day afterwards, I could have my field expert read through them and correct them if necessary, or add additional information where things were not clearly explained. The following paragraphs describe the projects and the compiled results of my findings.

5.2 Case 1: The Intermountain Power Project (IPP)

5.2.1 Project Description

IPP is a coal-fired power plant designed for four units of 750 megawatts, two of which have been constructed (Figure 5.1). It is located in Delta, Utah on a site of about 1850 acres (not including the area reserved for evaporation ponds). The project was conceived and initiated by the Los Angeles Department of Water and Power (LADW&P) and will be a main source of electric energy for the City of Los Angeles. LADW&P, the project manager, hired Black & Veatch Architect-Engineers (B&V) for the design of the

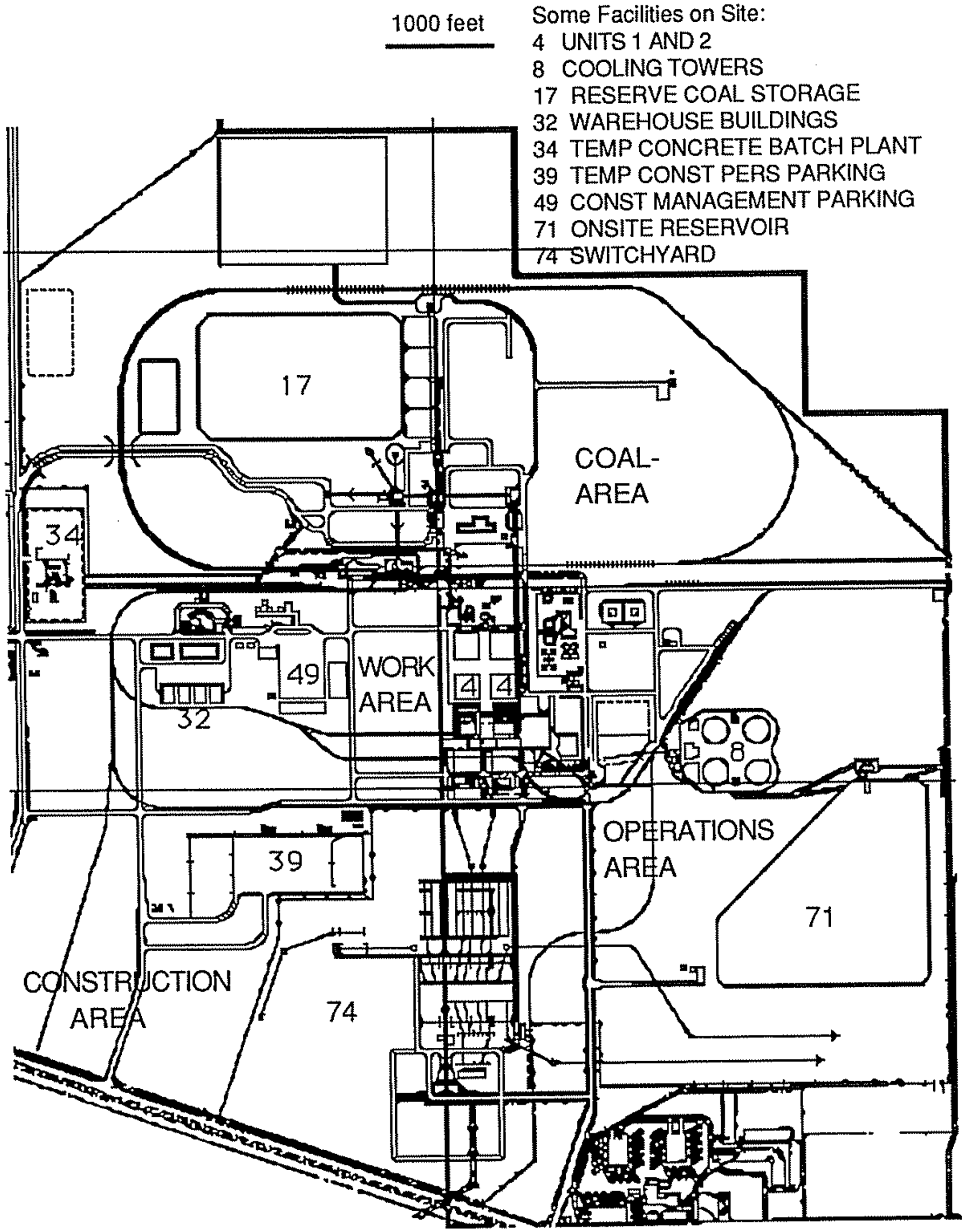


Figure 5.1: Site Arrangement of the Intermountain Power Project with Permanent and "Long-term Temporary" Facilities Generated by the Architect/Engineers

plant, and contracted with Bechtel Construction (Bechtel) to be the Construction Managers. With 1500 megawatts, constructed in a time span of six years, and at a construction cost of about $3.5 billion, this project is one of the few of this size constructed in the 1980s. For more information on the successful construction of IPP, see [Boltz 87] and [Reinhardt 87].

5.2.2 Layout Protocol Description

The description of the layout protocol that follows is a simplified interpretation of the layout process as it was described by the different parties. It captures the work of both the architect-engineers (AE) and the construction managers (CM) on IPP. Each party generated a layout design as needed for its specific task, so the result closely relates to the party's period of involvement and responsibility on the project.

Besides designing the permanent facilities, including power units, support buildings, permanent roads and railroads, the AE also laid out the temporary structures comprising warehouses, office buildings, first-aid facilities, entrance gates and brass alleys, security buildings, and management and labor parking lots. These are the buildings and construction support facilities that would be needed for almost the entire duration of the project, and some of which would later be used for maintenance of the plant in operation. They are listed in Table 5.1. Of course, all structures associated with the construction workers' entrance to the site had to be grouped together. For practical reasons, many of the other long-term temporary facilities were clustered in the same area so that they would not clutter up large open spaces on site that might be used for other purposes.

The AE also made rough estimates of needs for laydown space for construction and anticipated their grouping on the site. Accordingly, the AE extended the railroad and road grid to include construction railroads and roads. Upon completion of the design task, the AE produced the *site-arrangement drawing* (Figure 5.1), which was submitted together with a milestone schedule to the CM. As it turned out, at the beginning of construction the project owners revised the scope and decided to proceed with only two units instead of the planned four.

Part of the CM's task was to decide on the layout of the long-term laydown areas for approximately twenty-five major contractors (Table 5.2). The Lead Mechanical Coordinator was assigned to do this.

construction management office building
site service building
temp water storage tank
const personnel security labor office (a and b)
warehouse buildings (a through d)
concrete/ash access guard house
temp concrete batch plant
const water tank
temp const guardhouse
const materials security office
temp const personnel parking
temp token booths/brass alleys (a and b)
temp welder qualification testing
welding services building
temp additional contractor office area
temp air, mapp gas and oxygen area
temp fuel and lubricants area
temp paint and solvent storage building
const management parking
temp construction landfill
non-destructive weld testing

Table 5.1: "Long-term Temporary" Facilities laid out by the Architect/Engineers on IPP

Turbine-Generator
AQCS
Coal Yard Stacker Reclaimer
Coal Handling Equipment
Scrubber Additive Conveying Syst.
Superstructure
Steel Coal Handling
Handrail
GEESI
B&W laydown
Substructure - Centric
Sub Struct Coal Handle
Super Struct Coal Handle
General Construction Utilities
Insulation Laydown
Reserve area
Mech&Piping Units 1 and 2
TG Laydown (Mechanical)
Electrical Cable & Spool Yard
Misc. Laydown
Elect. Prefab.
Site Services
20"ø & 84"ø circ water
20"ø circ water
Unit II piping material

Table 5.2: Long-term Contractor Laydown Areas laid out by the Construction Managers on IPP

Starting with the site-arrangement drawing, the Lead Mechanical Coordinator identified all areas occupied with permanent facilities while checking the construction start and completion dates of each, all access roads and all otherwise unavailable areas on site. From the site arrangement, the CM inferred which area the AE had anticipated for long-term laydown. Since unit 1 would go on-line before completion of unit 2, a section of the site to the south-east of unit 1 was reserved for plant operation and thus could not be used for long-term construction laydown. The area immediately surrounding the power units was kept open as a work area and for short-term laydown. A temporary railroad extension gave access to the south-west corner of the site, so all laydown areas for contractor work on power units 1 and 2 would be concentrated in that so-called construction area. Contractors working on coal-handling facilities would be located in the coal storage area. Material laydown for the cooling towers and circulation water piping would be located near the cooling towers.

For each contractor, the CM specified the needed area, identified access requirements, determined whether or not major pieces of material would need to be moved to and from the laydown area, and established how critical the contractor's activity was. Based on this information, he *ranked* the areas by overall importance and picked the one ranked first to find an appropriate location for it on site. This meant determining in what area that laydown had to be (zoning constraint), determining whether or not the laydown needed to be adjacent to a railroad (adjacency constraint), and making sure that it did not overlap with roads or any of the fixed facilities on site (non-overlap constraint). Finally, if several alternative positions remained after these constraints were met, the CM satisfied the *preference* of the contractor to be as close as possible to the place of installation of the work in the permanent facility by picking the *best* position from the alternatives. Then, the CM repeated this process with the second contractor's laydown, and so on. The results of this process were finalized by highlighting and labeling areas on the site-arrangement drawing (Figure 5.2).

Before the award of contracts, contractors bidding the job were told what area would be available to lay down materials on site, so they could plan their work. Upon their arrival on the project, their assigned areas were then further subdivided to specifically accommodate their individual needs. For example, one contractor defined rows for material laydown and created aisleways for hydraulic cranes to reach and pick up materials (Figure 5.3).

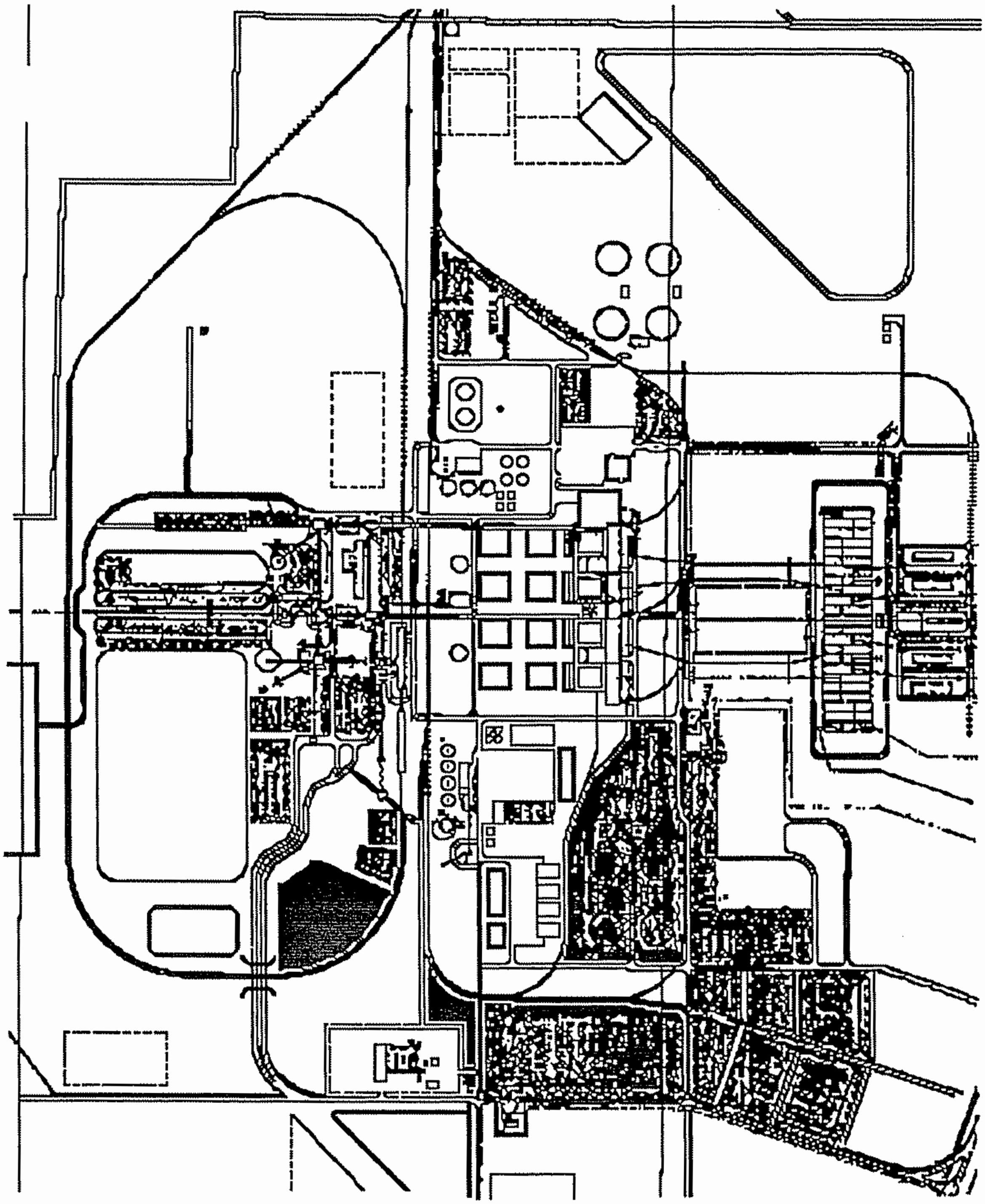


Figure 5.2: Site Arrangement of the Intermountain Power Project with Highlighted Long-term Laydown Areas Generated by the Lead Mechanical Coordinator Working for the Construction Manager

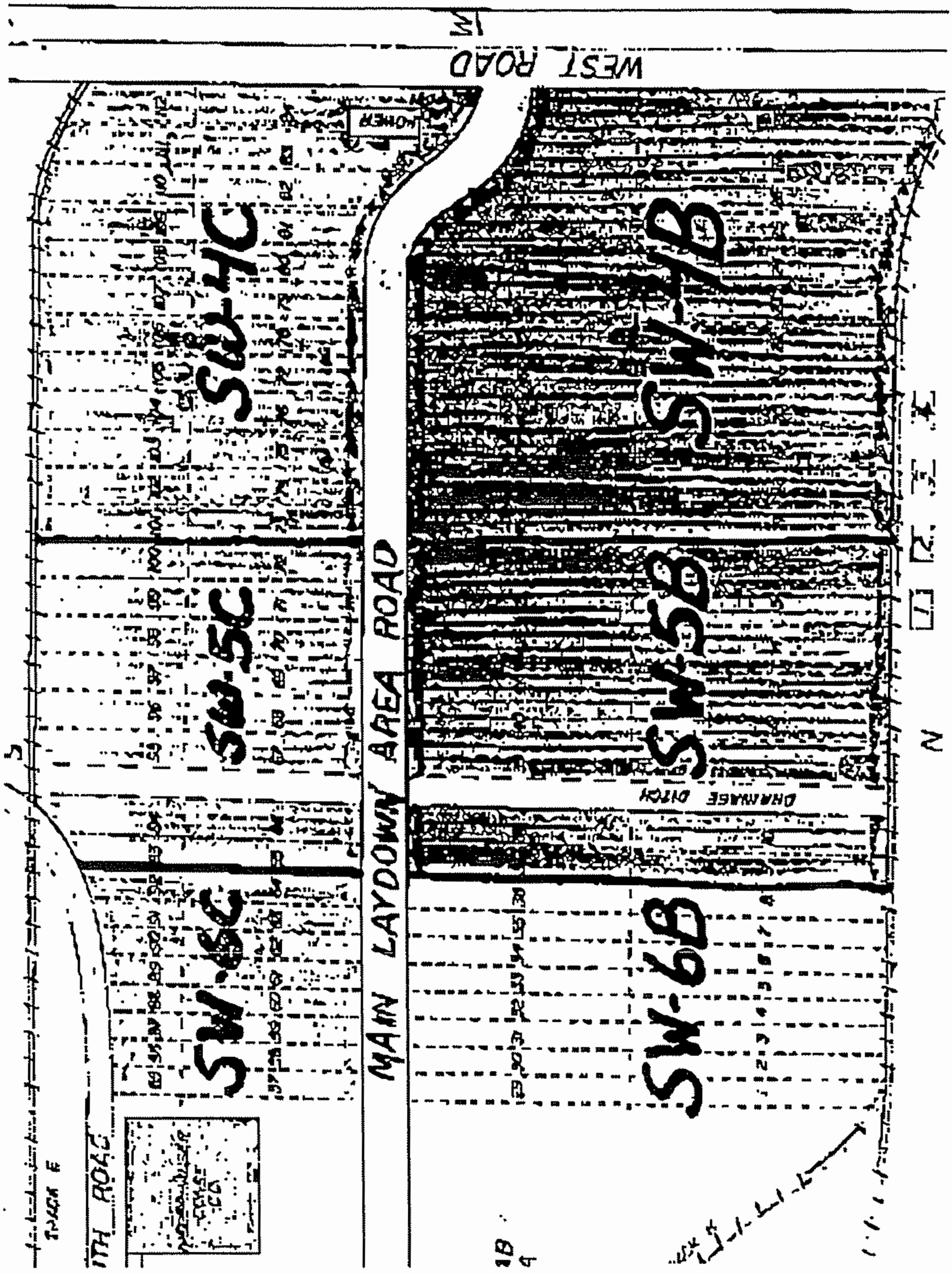


Figure 5.3: One Contractor's Laydown Area on the Intermountain Power Project further Subdivided to Accommodate a more Detailed Laydown Arrangement

The above description is necessarily a caricature of the layout process applied at the IPP site; although simple, it provides sufficient detail for the SightPlan implementation. This implementation will be discussed in Section 6.1, which describes SightPlan's Expert Strategy.

5.3 Case 2: The American 1 Project (AM1)

5.3.1 Project Description

The American 1 project is a co-generation plant, that is, a power plant that generates both electricity and steam (Figure 5.4). It is located in King City, California on a site of about 14 acres. The plant can burn either natural gas or diesel fuel and produces a total of 120 megawatts of which 90 megawatts are generated by the main combustion cycle and 30 megawatts are generated by the steam cycle. The project is located adjacent to a food-processing factory, and its main function is to supply steam for dehydrating garlic; for the owners of the factory the generation of electricity is almost a by-product. The owners hired Bechtel Construction to do the permitting, engineering, procurement, construction, startup & testing, and operations & maintenance of the project. This type of plant—co-generating about 120 megawatts and constructed in 20 months' time at a cost of about $100 million—is typical of many power plants constructed by independent power producers during the 1980s.

5.3.2 Layout Protocol Description

Because American 1 is so much smaller than IPP, the layout process was much simpler. Despite that, the process was sufficiently similar in principle for it to be a good test case to validate the strategy based on IPP. Basically, the construction manager identified the temporary support structures and laydown areas that would be needed on site and laid them out. The engineering and the construction of the plant were done by closely cooperating groups within the same company. The construction managers seized that opportunity to request some changes to the permanent layout in order to facilitate construction. For example, the earth berm built up from excavated materials alongside Metz Road and to the left of the plant entrance was reduced and the fencing moved outward to allow more fenced-in space for the temporary structures. In that way, the construction office trailer and the row of covered fabrication shops for pipe fitters, civil, and electrical crafts could be located in that prime space. In close collaboration with the field superintendents, the manager then allocated laydown areas for materials, some of

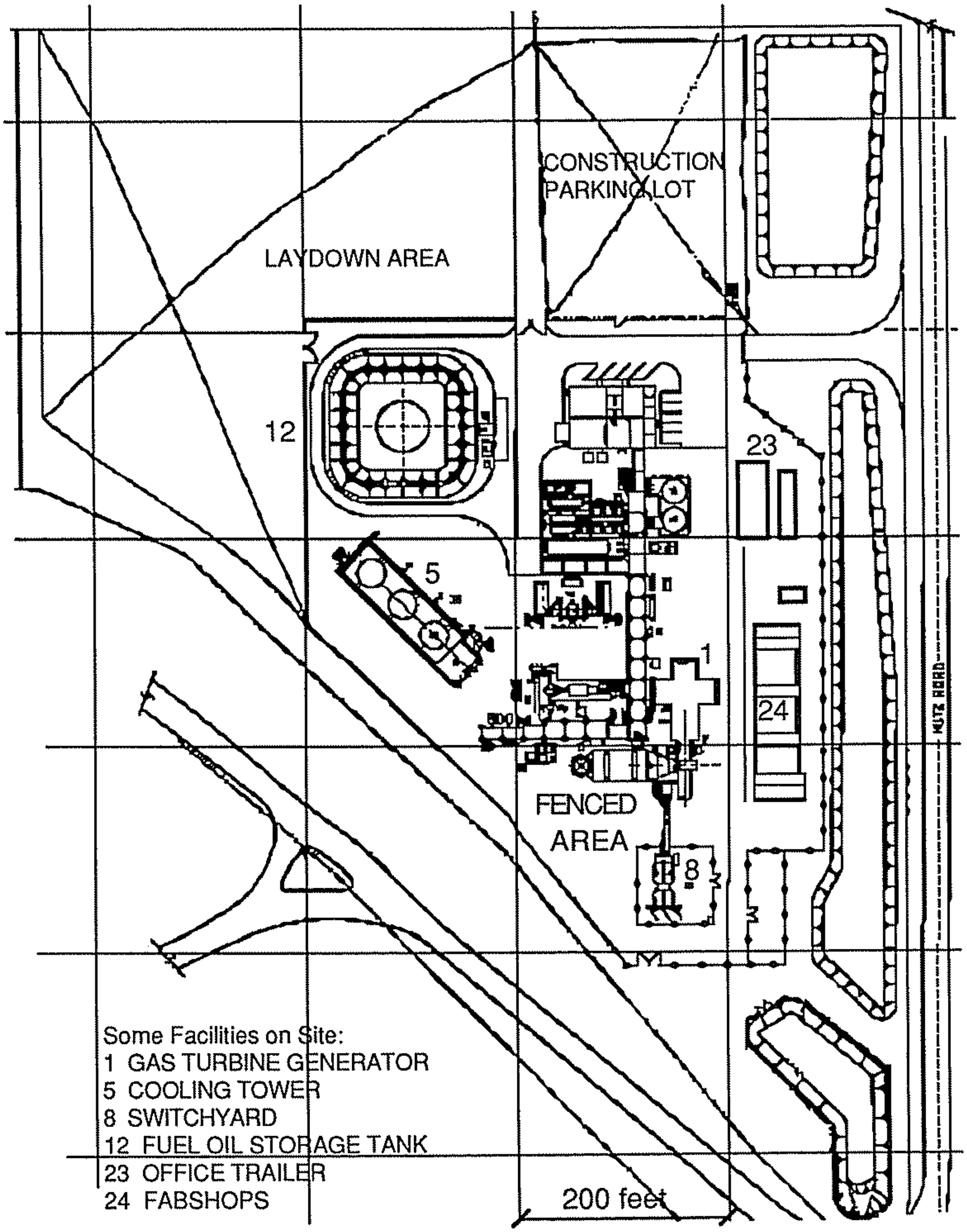


Figure 5.4: Fencing Plan of American 1
with some of the Temporary Facilities Laid Out by the Construction Manager

which needed to be within the fenced area of the site, while others could be on more open grounds. All of the temporary facilities on American 1 are listed in Table 5.3.

- Construction Office
- Fabrication Shops for Pipe fitters, Civil, and Electrical
- Civil (Lumber and Forms)
- Generator Turbine Inlet
- Electrical Conduit and wire
- Insulator
- Electrical Cable
- Piping
- Underground Piping and Valves
- Valves
- Pipe Hangers
- Fabricated Piping
- Cable Tray
- Scrap Iron
- Pipe Insulation (S/C)
- Cooling Tower (S/C)
- Scrap Formwood and Wood
- Gilroy Surplus

Table 5.3: Temporary Structures and Laydown Areas on American 1

On IPP, I had spent most of my time discussing the layout of long-term facilities. The expert had explained that, because the site was so spacious, there had been no need to reallocate any of the areas allocated for this layout for second use at later phases of construction. That is, the layout he generated was a static one. Since the American 1 project was so much less complex and built over a short time period, the field managers themselves raised the issue of how the layout had changed over time. One example the construction manager showed on the arrangement drawing was that the construction office was located first in a single trailer, then was moved to the control and maintenance building upon that building's completion, remained there until the building was needed for plant operation, and finally was relocated in a double trailer (Figure 5.4 and Table 5.4).

Single Wide Trailer	1-Sep-87	30-Jan-88
Perm Facility Office	1-Feb-88	31-Oct-88
Construction Startup Office	1-Nov-88	30-Apr-89

Table 5.4: Different Locations over Time for the Same Use:
Housing the Construction Office

Another example of change of the layout over time was the reuse of an area of about 6,000 sq ft, adjacent to the fabrication shops, very close to the gas turbine generator, adjacent to a road providing easy access, and fenced in. That area was considered prime space for laydown and thus would be in use at all times. As shown in Table 5.5, some tight scheduling was necessary to allocate this space for different purposes over time.

Civil—Lumber and Forms	1-Dec-87	30-Apr-88
Generator Turbine Inlet	1-May-88	30-Jun-88
Electrical Conduit and Wire	1-Jul-88	31-Dec-88
Insulator	1-Oct-88	30-Apr-89

Table 5.5: Different Uses over Time for the Same Laydown Area

Section 6.2 will discuss how the information from the American 1 project validated the Expert Strategy, and suggests how one might model a site layout that changes over time.

5.4 Comments on the Information Obtained

5.4.1 Sizing Temporary Facilities

Both of the projects under study were at the start-up phase by the time I reached the site, and it was difficult to learn about the rules used for predetermining the size of facilities, even though I asked managers repeatedly how they estimated these sizes. Because of this difficulty, I was obliged to provide SightPlan with the input of facilities with predefined geometry so that its task would be limited to laying them out on site. I am still convinced, though, that people do use rules of thumb to size facilities, although they may feel uncomfortable articulating such rules because of their apparent simplicity or lack of foundation. A test for this conviction would be to observe a person making the very first layout arrangement, before construction has started. Such a study might disclose a trial-and-error process alternating between sizing facilities and locating them, which would demonstrate that separating those two tasks is unrealistic. This study is suggested as one of a number of proposed future refinements to SightPlan.

5.4.2 Space Available on Site

In the cases of both the IPP and the American 1 sites, construction personnel felt there was plenty of space for laydown areas. This distinguishes these two sites from, for instance, sites of high-rise buildings to be erected in a city's downtown area. The main issue that needed to be addressed by the person allocating the space was not *how much* space to allocate, but rather, *where* to allocate it. The various laydowns could be positioned at several possible locations, and the subcontractors did not have strong objections to the areas proposed by the construction manager, so the overall problems are *apparently* underconstrained. As will be discussed in Section 6.1.3, this affects the strategy that is pursued to generate the layout.

5.4.3 Layout Drawings Showing Temporary Structures

Layout drawings showing temporary structures are often disposed of after project completion. It was by pure coincidence that I obtained the marked-up layout drawing showing the long-term laydown areas on the IPP site; and the field construction manager on American 1 redrew his layout in reply to my inquiry. That documents reporting on temporary facilities are themselves temporary is easy to understand. They have little if any informative value upon construction completion and they represent items that change over time and therefore would require a lot of record keeping if they were to represent the site situation at all times. Yet, the accumulation of such records over many projects might provide data from which one might learn how to improve site layouts, and would thus be worth doing.

At a first glance, layout drawings seem to reflect the site layout taken at a snapshot in time. In fact, upon closer inspection, they usually contain information that relates objects at different moments in time with one another. In Figure 5.4, for example, the construction manager drew the laydown for underground piping and valves *overlapping* with the fuel oil storage tank. Since the tank was only to be constructed at a later date in the project, the area it occupied could be used for other purposes early on in the project. It is worth mentioning that field practitioners appear to look at a layout drawing while keeping in the back of their minds how it will change over time. An example of this became apparent when I explicitly asked the construction manager on American 1 to use the five copied layouts I gave him to first identify the major phases of construction and then to draw the layout plan for each of these. He did not follow these instructions and instead marked up only one plan while labeling areas with several items and their associated time period on site. I can only guess why he did it this way rather than the way I had suggested. One possibility is that there are so many activities going on at the same time on site, and there is so much continuous change, that it may be difficult to pick major phases for the layout. Another possibility is that even when major phases could be identified, a number of facilities would remain at the same location in contiguous phases, but they would have to be redrawn on each of several drawings. Doing this would require uninteresting repetitive work that the manager may want to avoid. To clarify how people perceive sites evolving over time is definitely an area of research in spatial reasoning that should be further investigated.

Chapter 6

SightPlan Experiments

SightPlan explicitly represents the strategy it follows to construct a solution. Such a strategy is important in determining what solution will be generated. As described in Section 2.3.2, layout methods generally rely on heuristics, and most of the time end with a "satisfactory" solution because they cannot guarantee *a priori* that they will obtain optimal layouts. Therefore, developing an initial strategy for SightPlan, and comparing that with an alternate one, would be worthwhile as an experiment.

Several strategies were crafted and applied to the two case studies described in Chapter 5. The first model—pursuing 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. The second model—pursuing the *Temporal Strategy*—applies that same Expert Strategy to another construction site, that of American 1 (AM1), with the intent of validating and assessing the strategy's generality. The smaller scope of AM1 made the implementation more manageable. Thus, I was able to extend the Expert Strategy to do reasoning about changes in the layout over time. The third and last model—pursuing the *Computational Strategy*—was designed to make better use of the power provided by the computer than can be achieved by mimicking a person's strategy. These models will be described next and compared with one another.

6.1 Expert Strategy on IPP

6.1.1 Description and Scope of the Expert Strategy on IPP

The first model reflects the division of tasks that exists in civil engineering as it is practiced today. In addition to designing and laying out the permanent facilities, the Architect-Engineers (AE) on IPP also laid out what I called "Long-term Temporary Facilities" (these were given in Section 5.2.2 in Table 5.1). After the AE completed their work, the Construction Managers (CM) laid out the "Long-term Laydown Areas" for the major contractors on site (given in Table 5.2). Short-term temporary facilities are left out of this model. Including them would have substantially increased the complexity of the computations, and would not necessarily have led to more interesting discoveries.

This division of tasks found a representation in SightPlan: SightPlan generates layouts at two *snapshots* in the design-construct life time of the project. One snapshot is taken probably around Time 5 in Figure 2.4, before the moment of completion of the Final Layout Model. The second one is taken probably around Time 9 in the same figure, after the civil contractor commenced work on foundations, but before other major contractors arrive on the site. SightPlan treats the layout it generates at each of these snapshots in time as a static layout.

6.1.2 Implementation of the Expert Strategy on IPP

Managers laying out the IPP site followed a recognizable sequence of types of actions. SightPlan implements this Expert Strategy by sub-strategies, foci, and heuristics. Because these are posted on the control plan in an predetermined sequence during problem solving, they could be structured in a skeletal plan, as shown in Figures 6.1 and 6.2. Despite the fixed order of strategic steps, the order of domain actions is opportunistic; they compete for the highest match against the current control during problem solving.

Table 6.1 charts how the Expert Strategy functions over the duration of a problem-solving session. The far left column lists the cycle numbers. To the right of it are the foci of the control plan. Further to the right are the executed domain actions. Let us consider one cycle and look at the information in that row in the table. The focus shown is the one associated with the highest-rated action of this cycle. It is this focus' heuristic that contributes most to make this action the highest-rated. If other foci are active in that cycle, then they contribute only in minor ways to this action's rating. To the right of the shown focus is the domain action that gets rated highest in the current execution cycle, and thus executes. Note that the figure fails to show, among other things, how a strategy could consist of several sub-strategies, how it might group several foci together, how multiple foci could be running in parallel at any cycle of the control plan, or what heuristics are used to implement the foci. A number of cycles in which SightPlan initializes its control, and some intermediate steps of that nature have been omitted. Similarly, cycles with display actions that show the user intermediate layout results are disregarded. In the following text, cycle numbers shown in parentheses correspond to the cycle numbers in

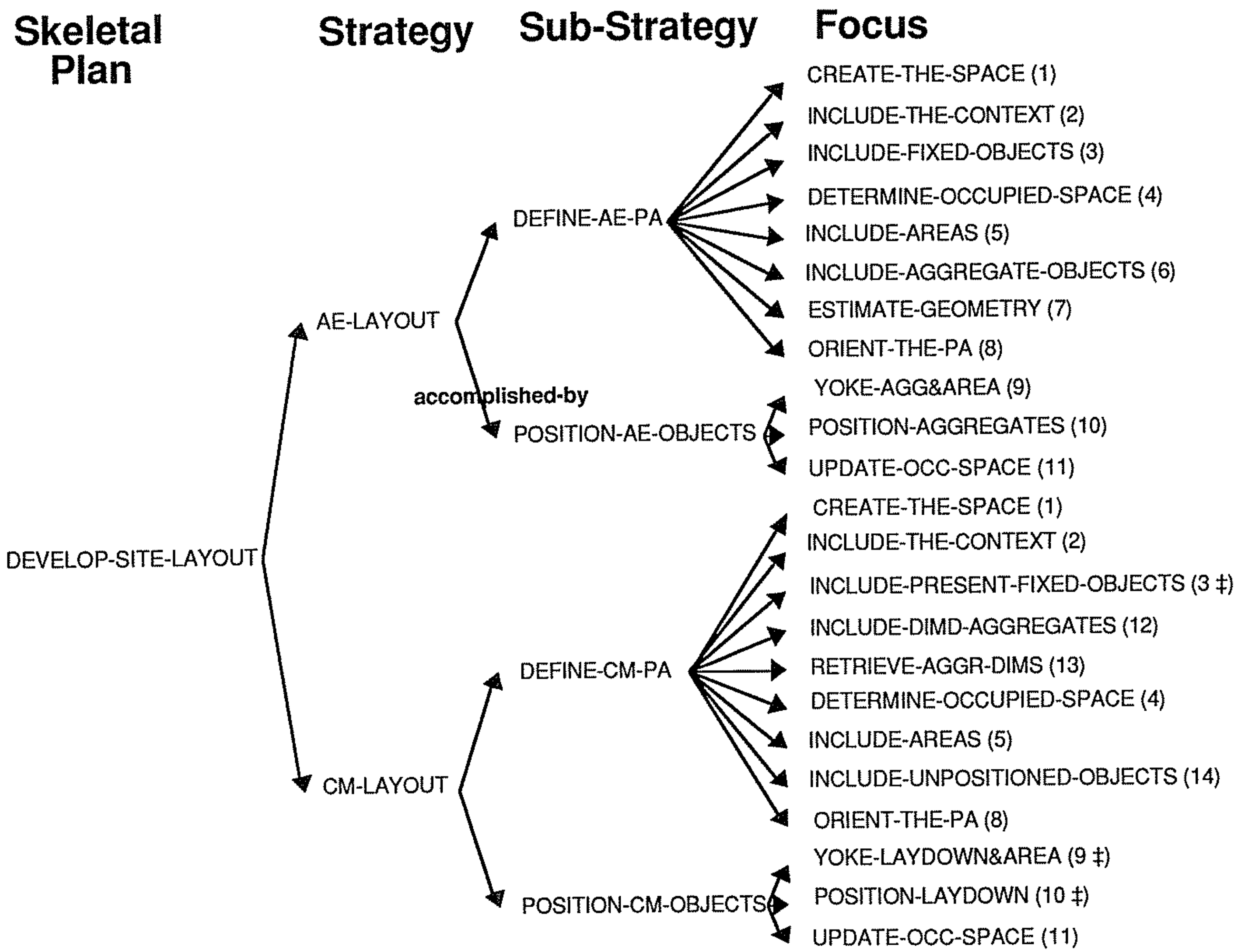


Figure 6.1: First part of the Skeletal Plan of the Expert Strategy Applied to IPP. The numbers in parentheses label the foci so that this skeletal plan can easily be compared to those of Figures 6.2, 6.25, 6.31, and 6.41.

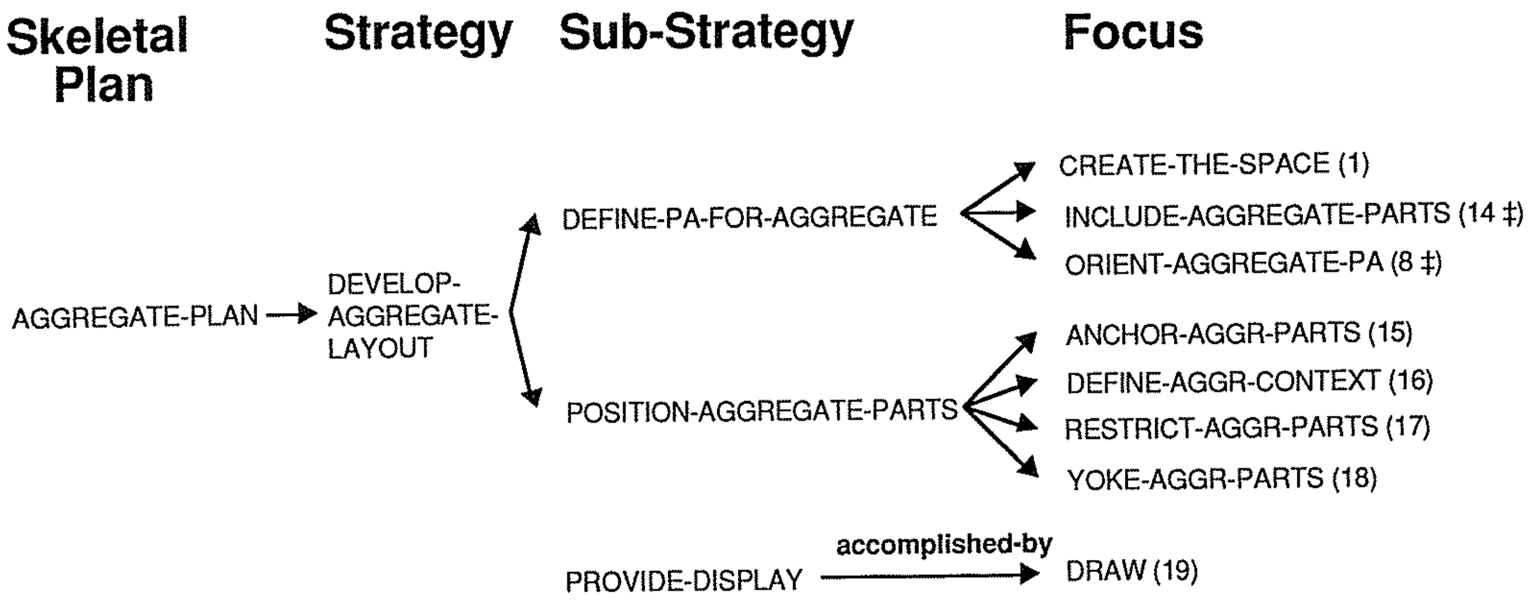


Figure 6.2: Second part of the Skeletal Plan of the Expert Strategy Applied to IPP
The numbers in parentheses label the foci so that this skeletal plan can easily be compared to those of Figures 6.1, 6.25, 6.31, and 6.41.

Table 6.1, and figures that accompany the text show SightPlan's display of how the layout evolves.

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, Figure 6.3) 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, Figure 6.4. Note that I edited the bitmaps and added labels next to rectangles on the figures after SightPlan displayed the layout.). *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.

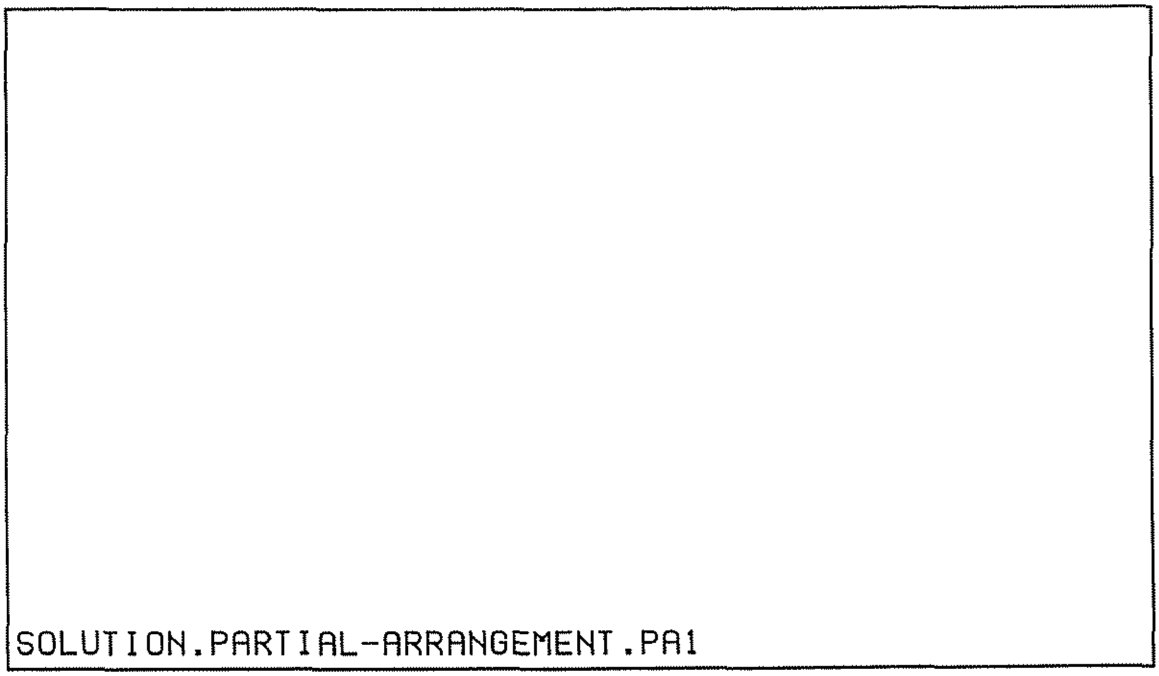


Figure 6.3: SightPlan creates PA1

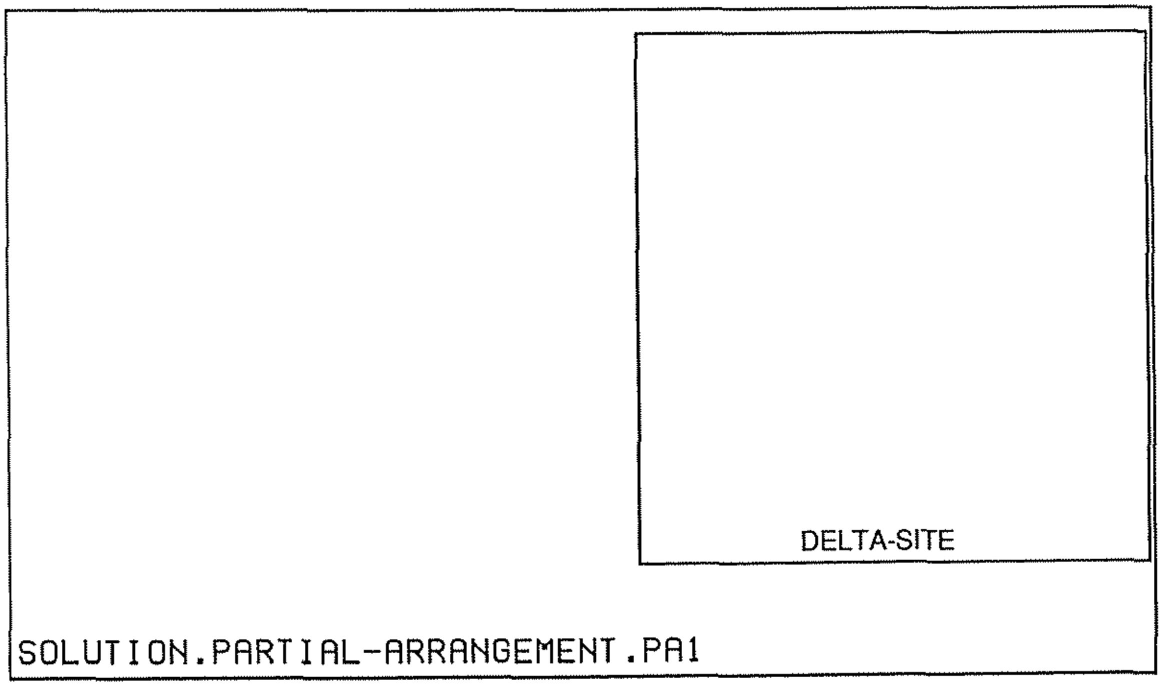


Figure 6.4: SightPlan includes the Context of the site in Delta, Utah

Following this, all objects that have a predefined and fixed location on the site are included (Cycle 24, Figure 6.5). 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.

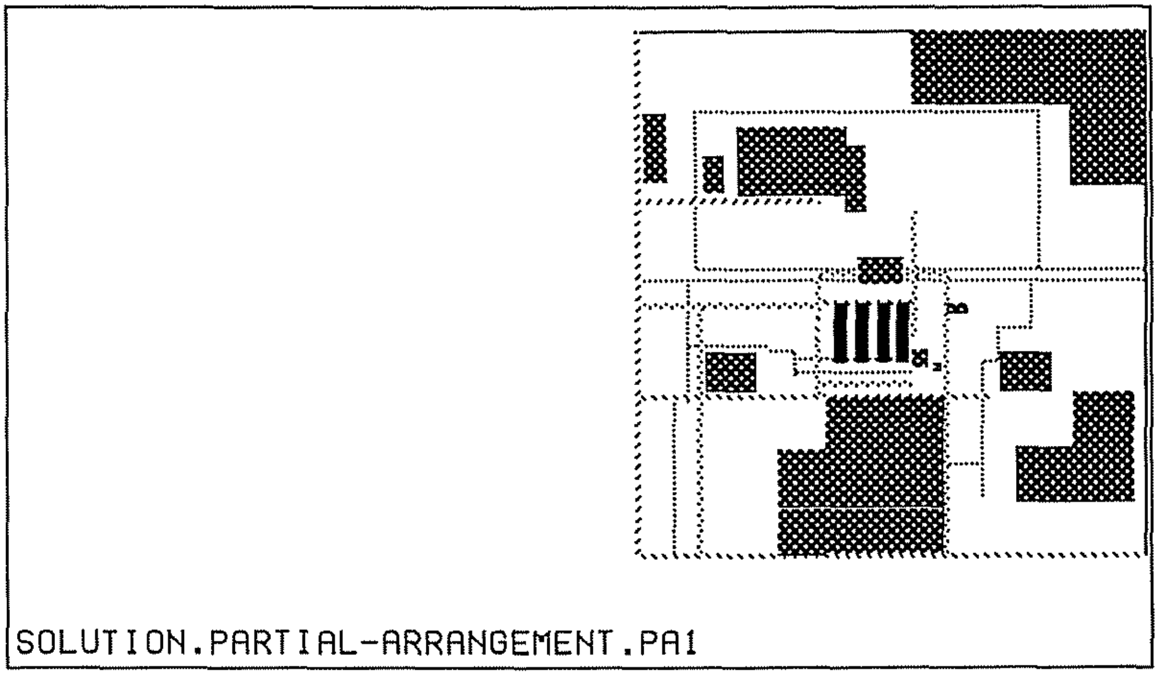


Figure 6.5: SightPlan includes all Permanent Facilities on IPP

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 6.6 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 not available anymore.)* 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 6.7).

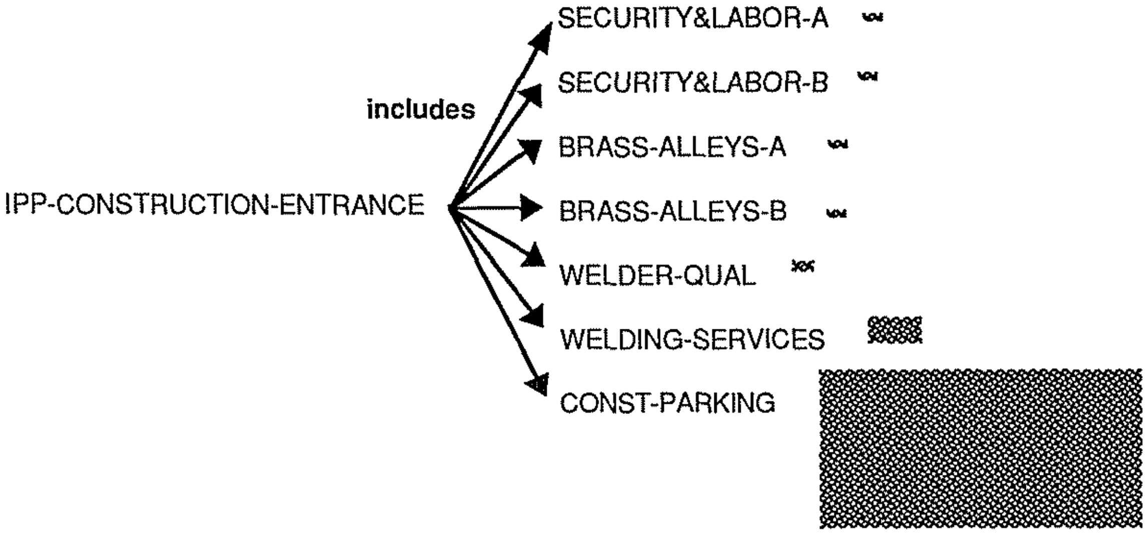


Figure 6.6: Example of the IPP-CONSTRUCTION-ENTRANCE Aggregate Object Showing the *includes* Links to its Parts, and the Geometry of the Parts

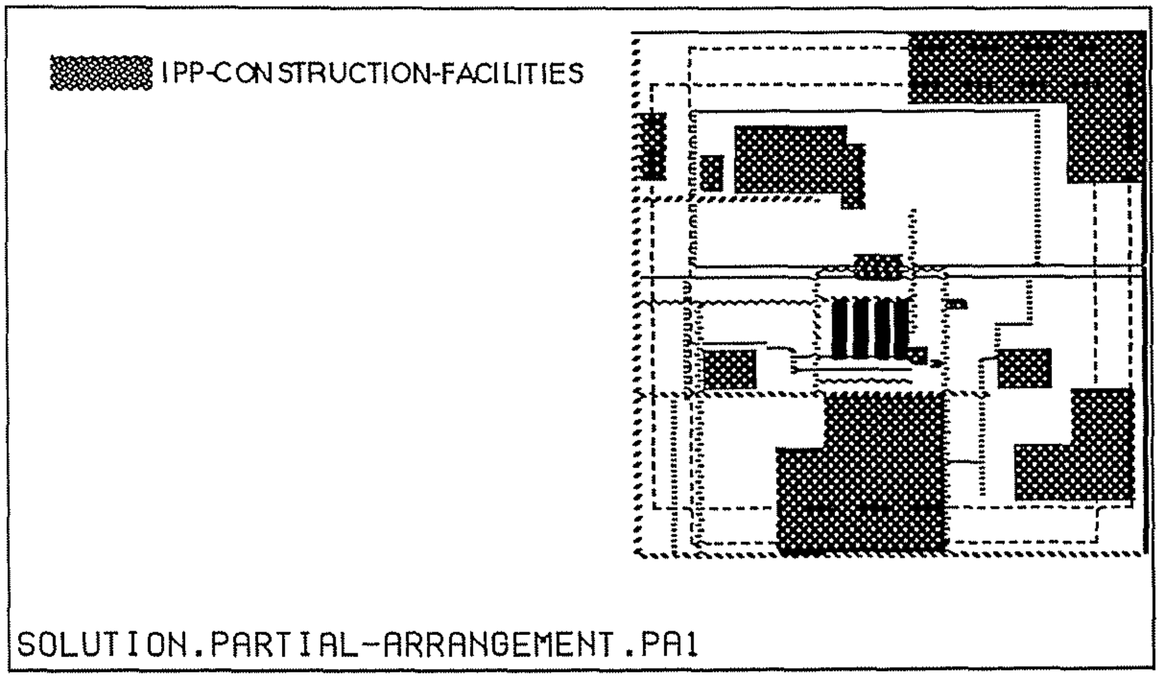


Figure 6.7: Sized and Shaped FACILITIES Aggregate included in PA1

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, Figure 6.8).

```
SOLUTION.PARTIAL-ARRANGEMENT.PA2
```

Figure 6.8: SightPlan builds an Alternate Arrangement for the ENTRANCE

No specific context is known, but all the parts of the ENTRANCE need to be included (Cycles 60-72, Figure 6.9). (Note that some intermediate steps are for display actions.) SightPlan picks the largest one as anchor of the arrangement (Cycle 77, Figure 6.10), with respect to which the other parts will be positioned.

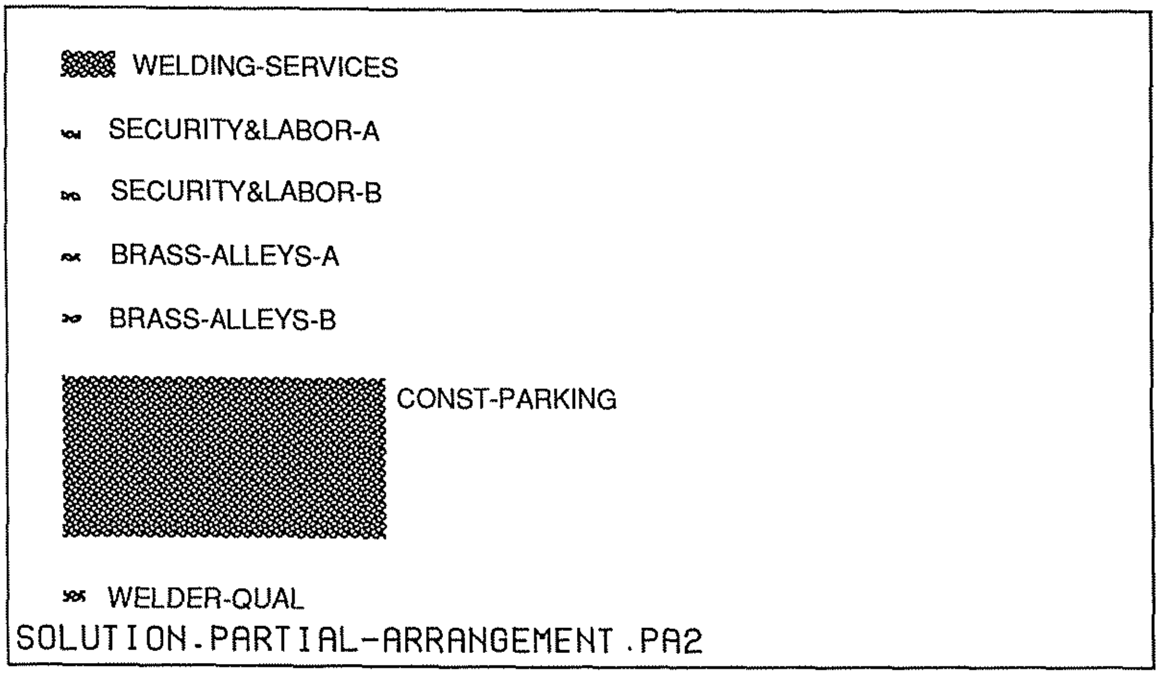


Figure 6.9: PA2 Includes the Parts of IPP-CONSTRUCTION-ENTRANCE

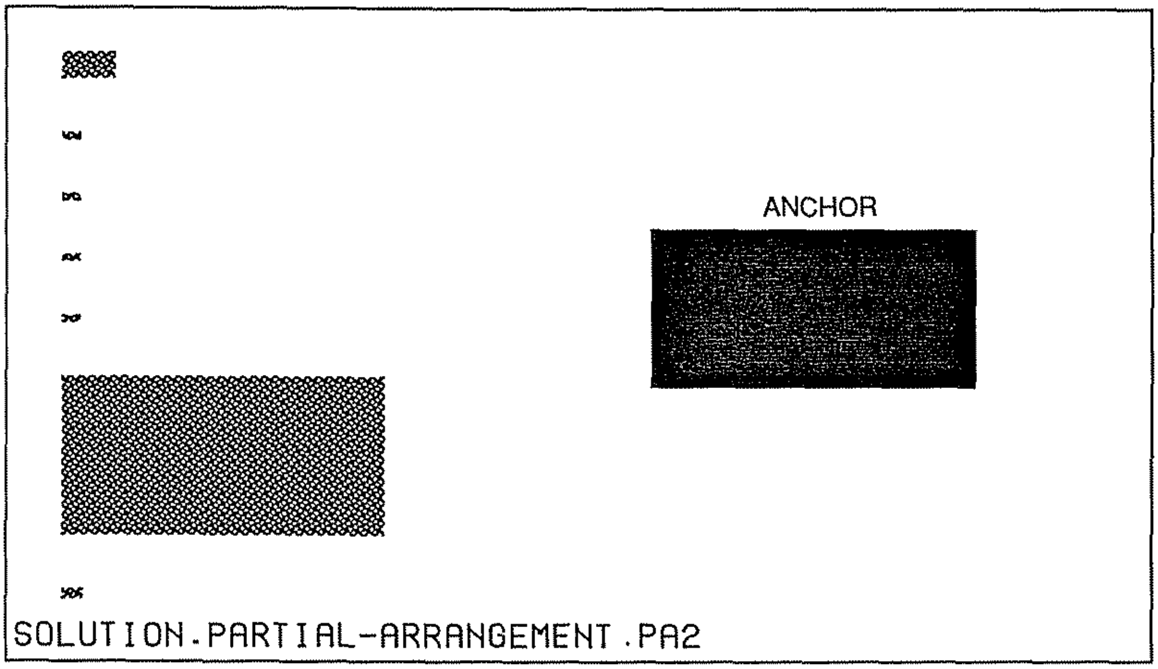


Figure 6.10: Largest Part is chosen for the Anchor

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 (Figure 6.11), "North-of" and "between-short-sides" constraints (Figure 6.12), "closer-than" constraint, and/or "adjacent-to" constraint (Figure 6.13) with the anchor (Cycles 82-136, Figure 6.14).

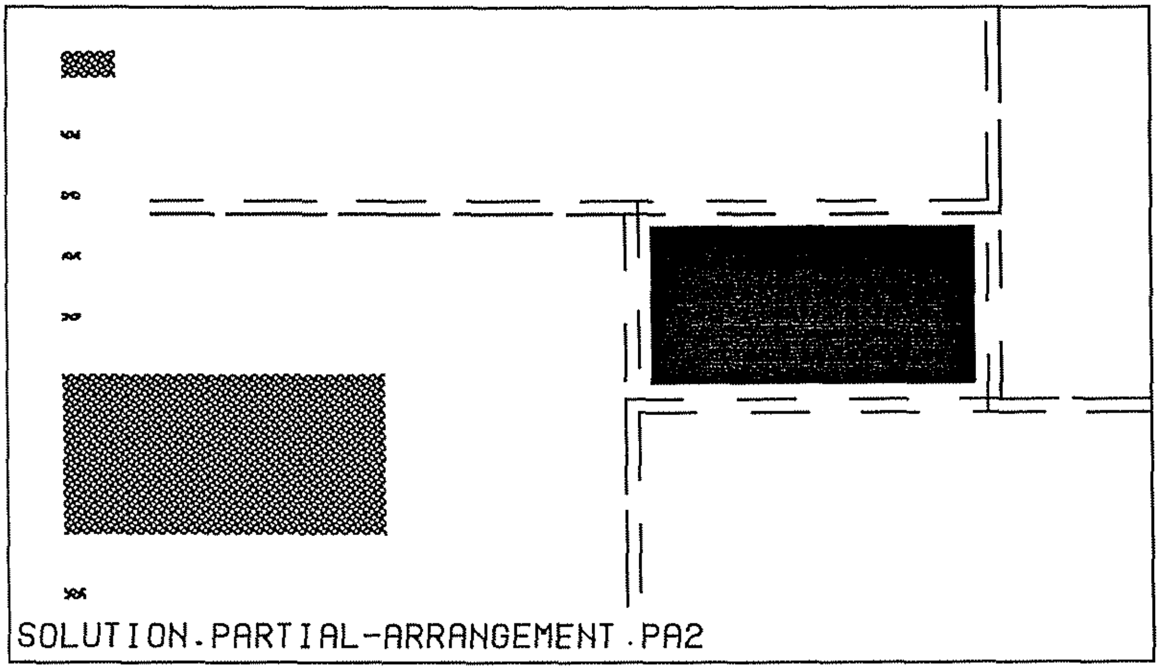


Figure 6.11: Largest Part Meets "non-overlap" Constraint with the Anchor

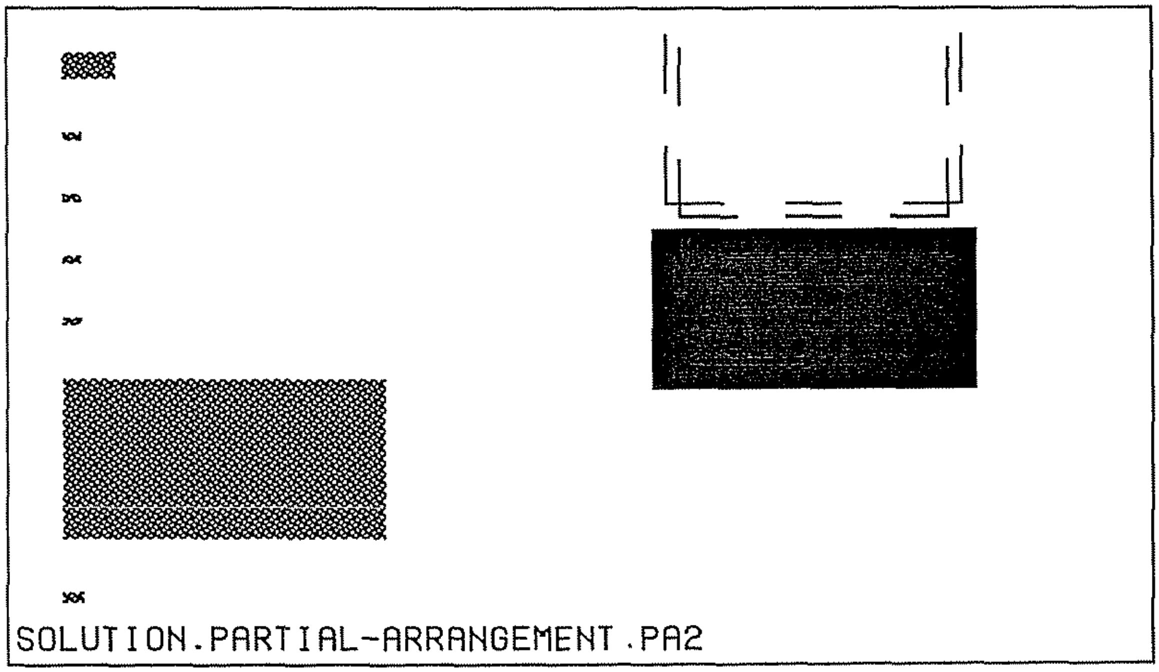


Figure 6.12: Largest Part Meets "North-of" and "between-short-sides" Constraints

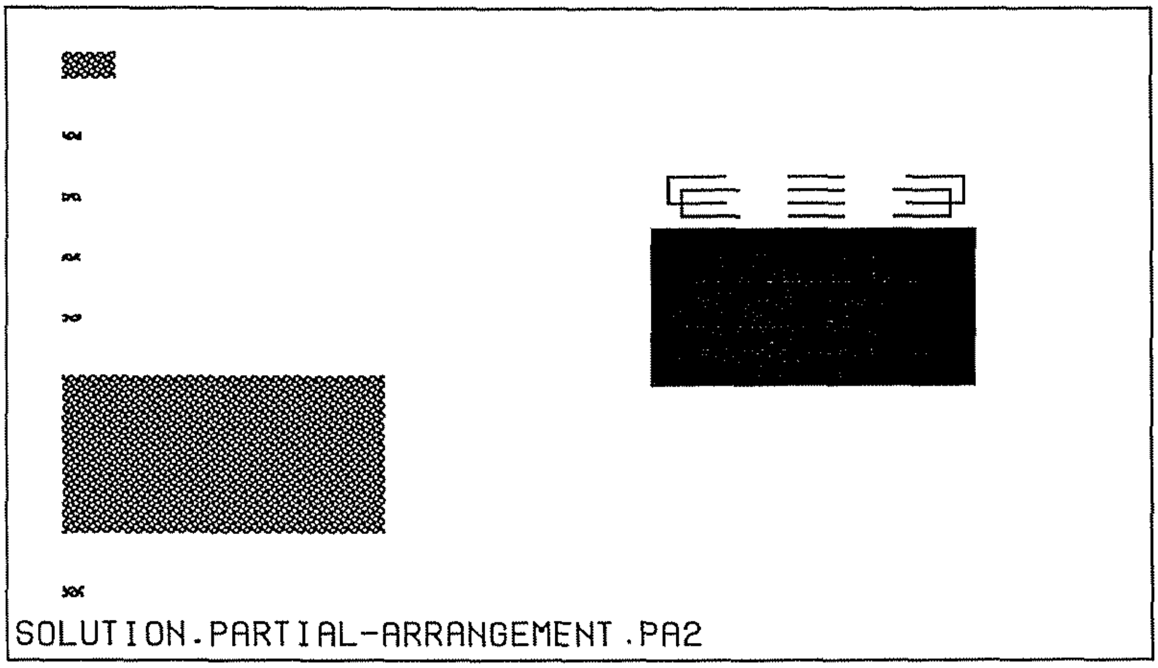


Figure 6.13: Largest Part Meets “closer-than”Constraint with the Anchor

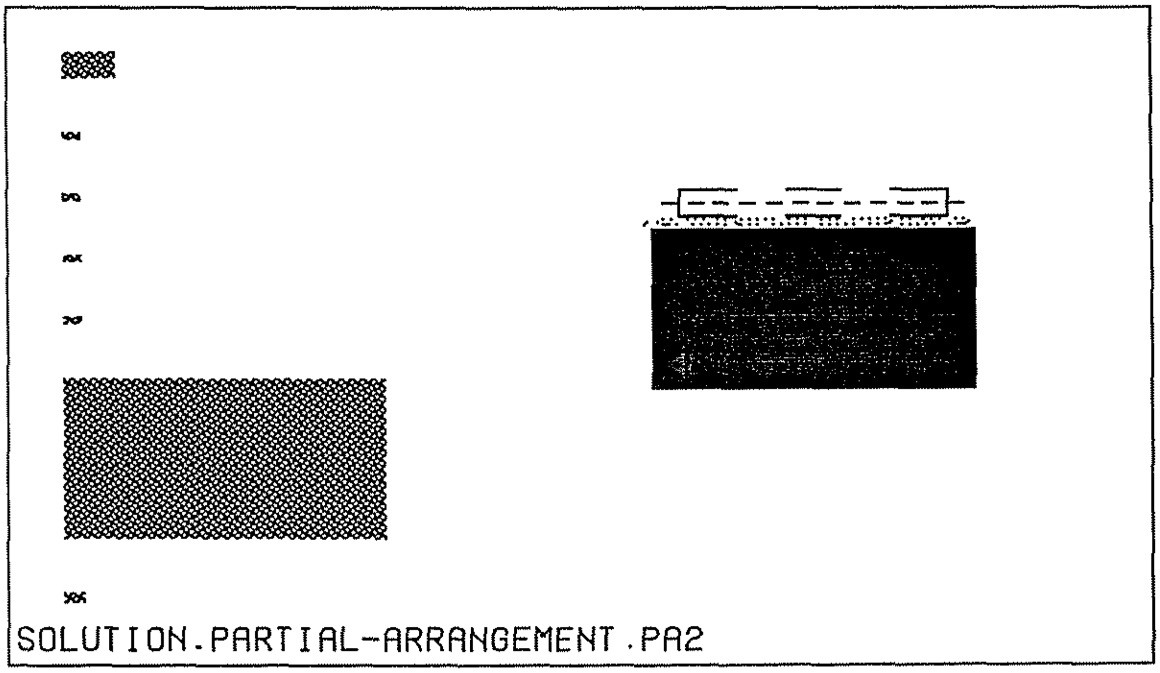


Figure 6.14: All Parts have met their Constraining Constraints with the Anchor

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, Figure 6.15).

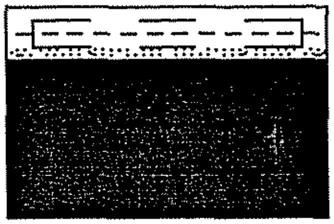


Figure 6.15: SightPlan Defined Boundaries on Context of Partially Laid Out PA2

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, Figure 6.16). SightPlan chooses the first constructed power unit (unit 1) as anchor in PA1 (Cycle 152, Figure 6.16).

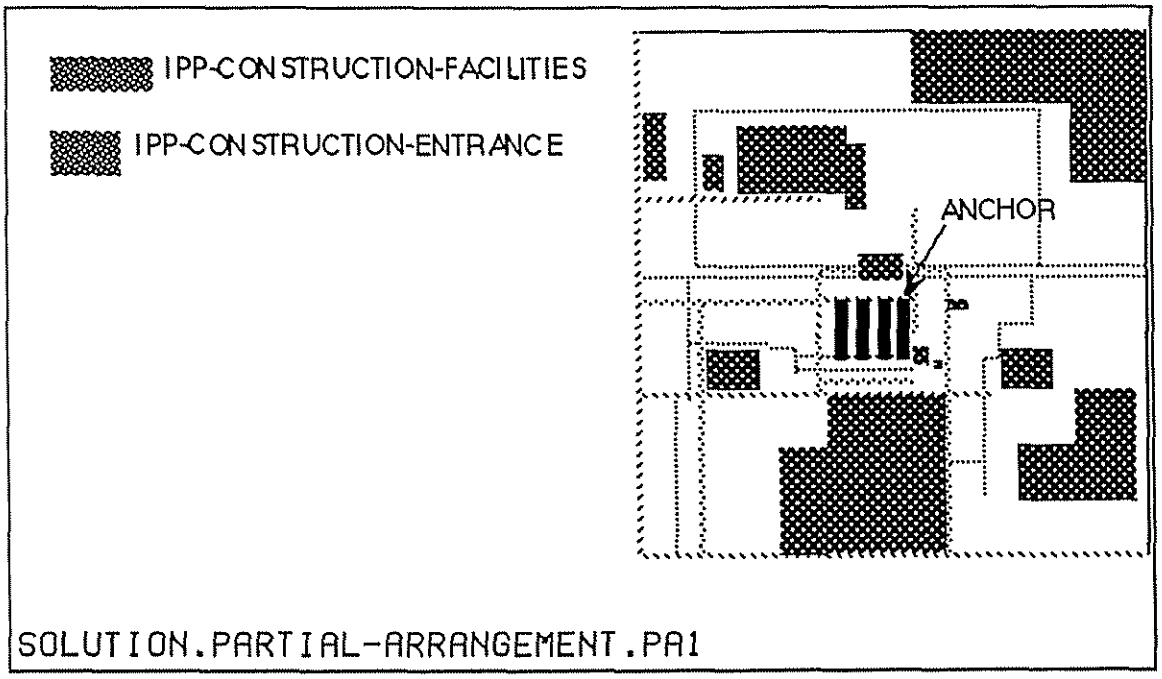


Figure 6.16: Two Shaped and Sized Aggregates in PA1

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, Figure 6.17).

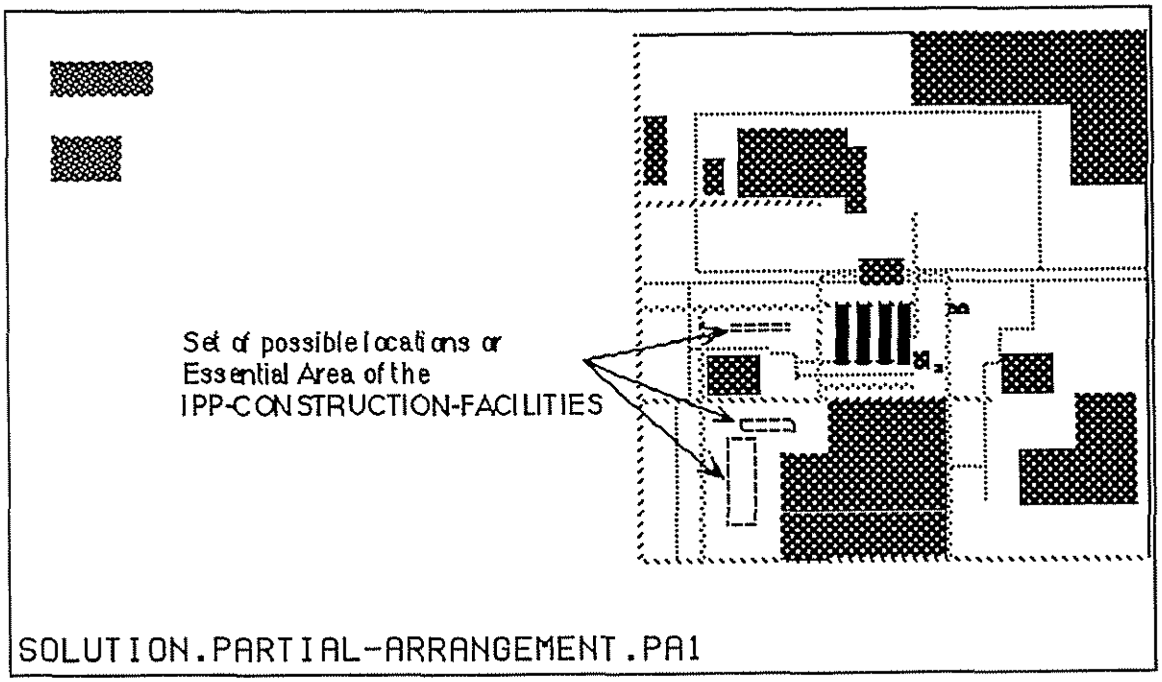


Figure 6.17: IPP-CONSTRUCTION-FACILITIES has met all constraints except the preference constraint "as-close-as-possible to POWER-UNIT-1"

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, Figure 6.18) and updates the occupied space to account for this positioning (Cycle 166).

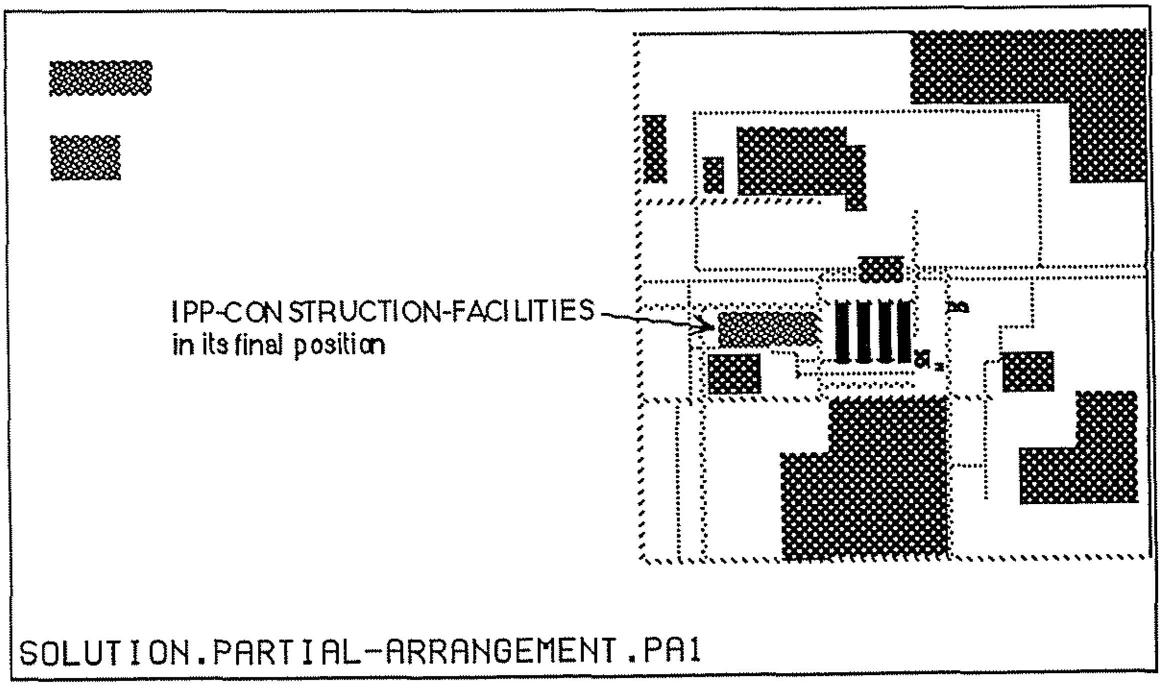


Figure 6.18: Best Position for IPP-CONSTRUCTION-FACILITIES

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. Figure 6.19 depicts SightPlan's AE solution layout.

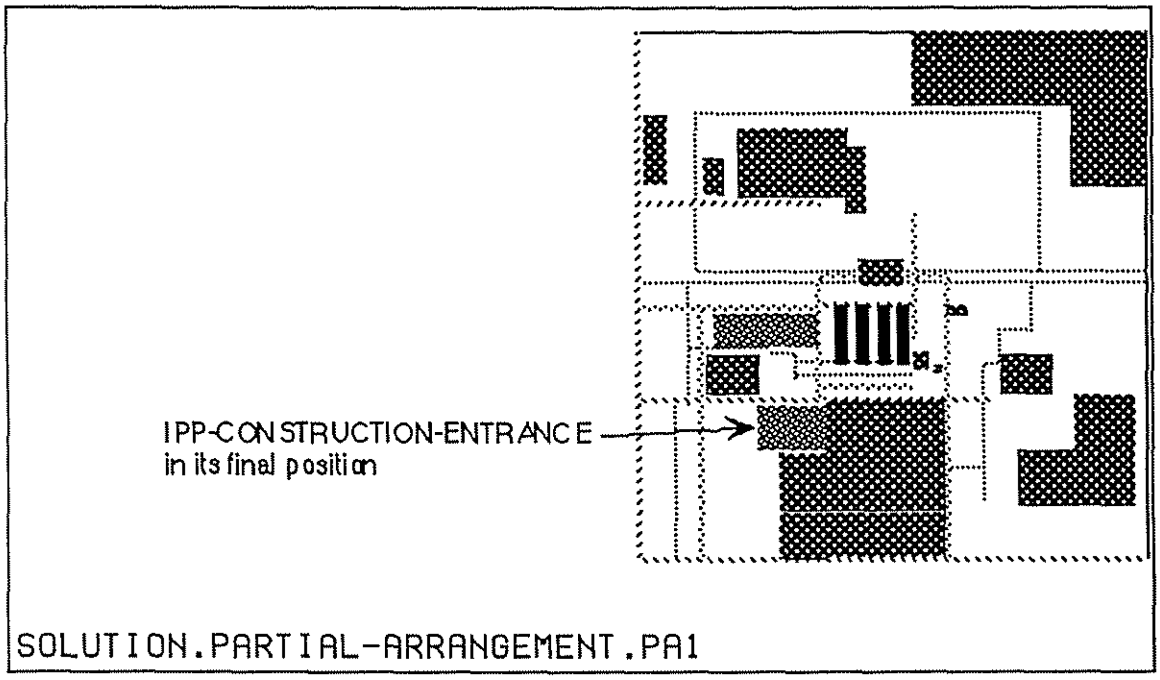


Figure 6.19: SightPlan's solution to the AE's Layout Task on IPP

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. Figure 6.20 shows the resulting layout. 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.

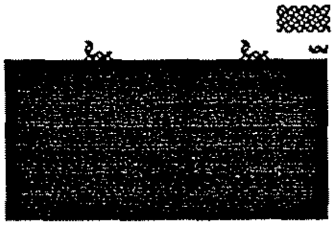


Figure 6.20: Completed ENTRANCE Aggregate Layout

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, I 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, Figure 6.21). Again, unit 1 is chosen as anchor (Cycle 44, Figure 6.21).

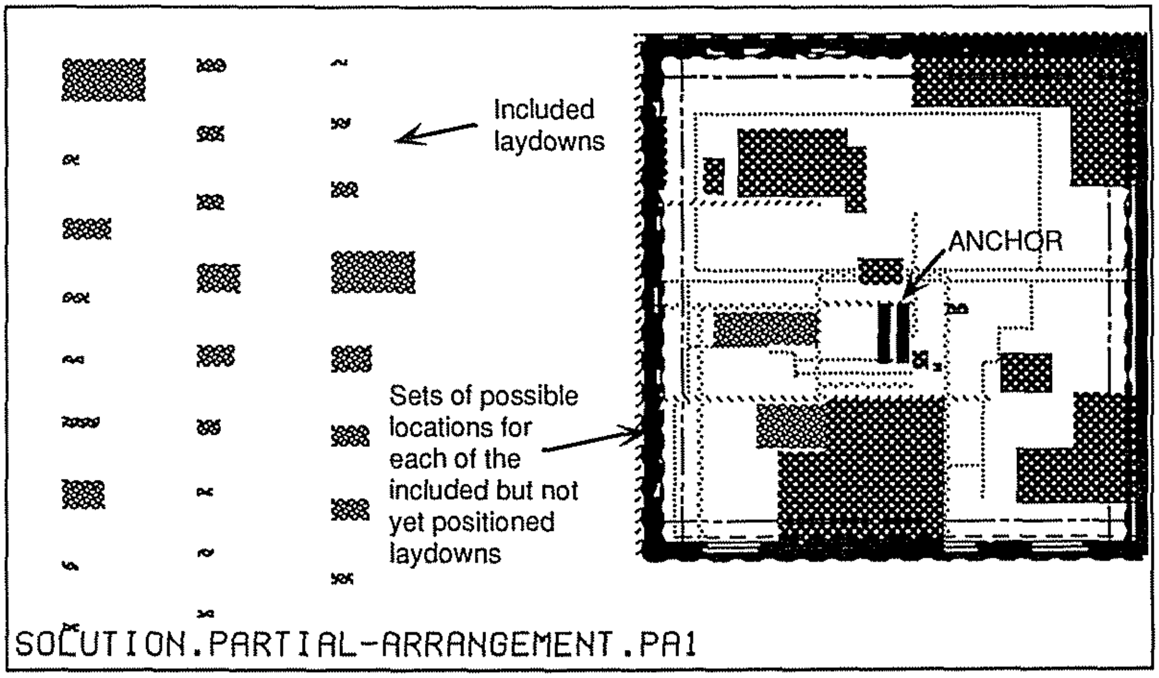


Figure 6.21: All LAYDOWNS Included in PA1 for Layout by the CM

Each of the LAYDOWNs is located according to its zoning constraint (Cycles 50-136, Figure 6.22).

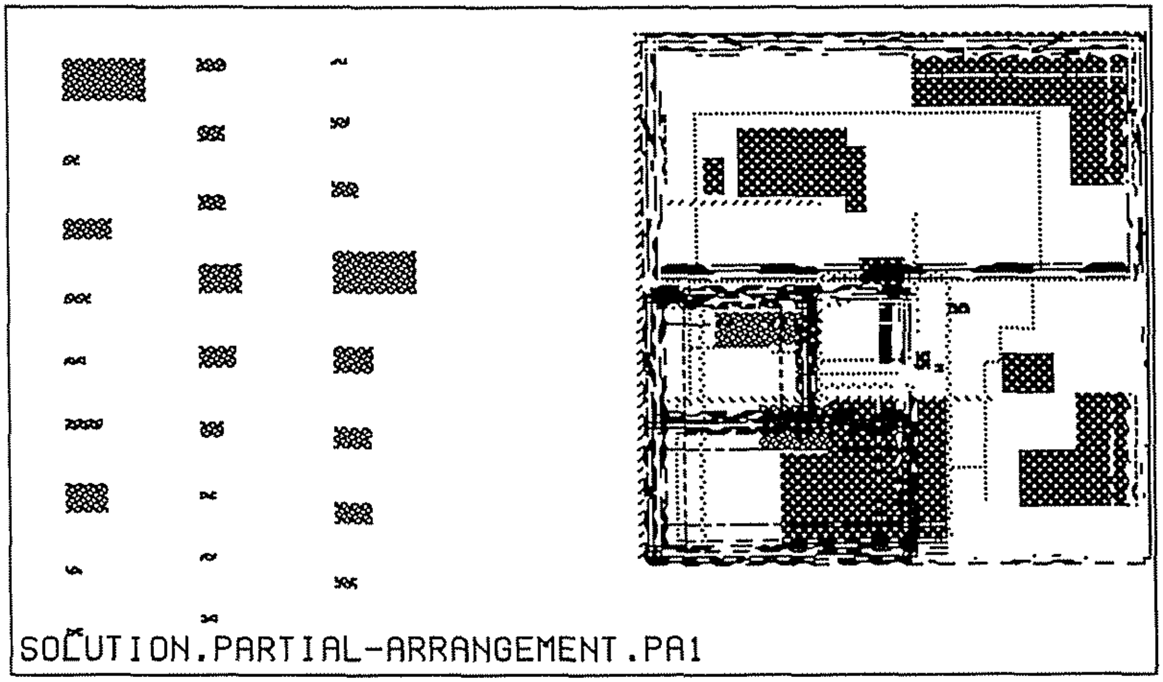


Figure 6.22: All LAYDOWNs meet their Zoning constraints.

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 6.23 depicts SightPlan's CM solution layout.

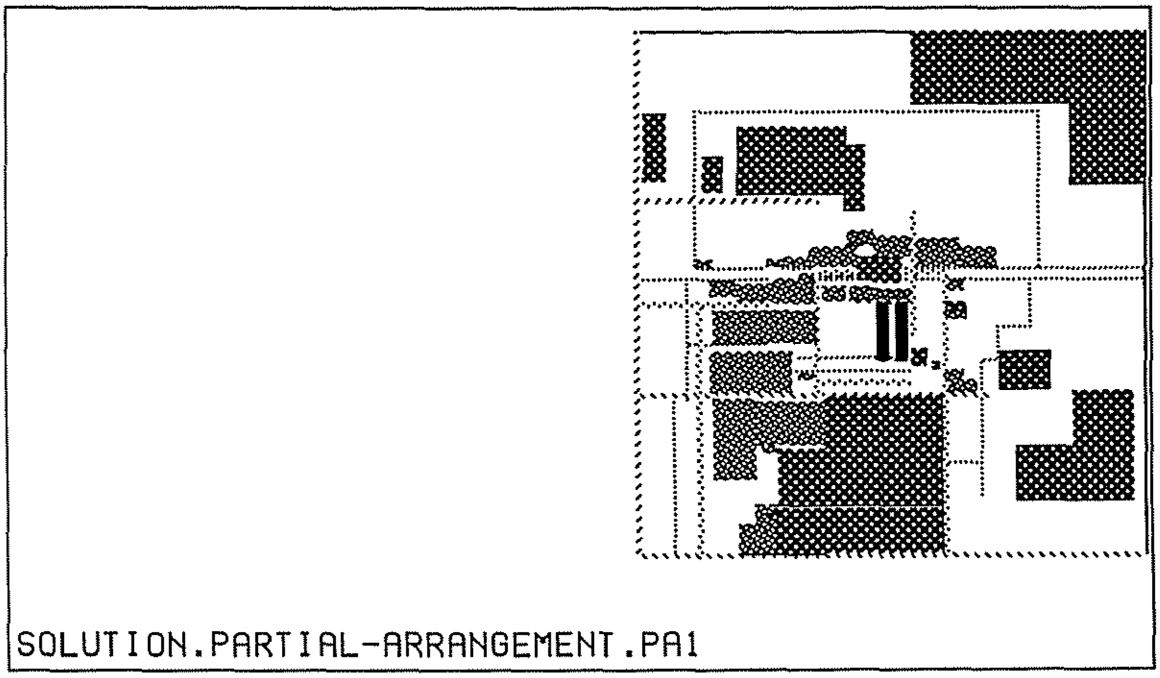


Figure 6.23: *SightPlan's Solution to the* CM's Layout Task on IPP

6.1.3 Discussion of the Expert Strategy on IPP

6.1.3.1 Layout Method

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. 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 suitable for articulating layout knowledge and managers' 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 and foci guide the arrangement's construction. 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 system obtains this flexibility by making use of an explicit control strategy that details for one or more steps at a time what the best actions are and that changes during problem solving.

I stress that the Expert Strategy implemented here reflects my interpretation of what I learned from the interviews I conducted; I cannot verify that the AE or the CM laid out the two partial arrangements in exactly the same way as SightPlan does. However, the *ex post facto* interviews led me to believe that the method SightPlan follows (developing partial arrangements and integrating those into the overall arrangement, and finding space for a grouping of objects in the overall arrangement and detailing it later) reflects the way in which the AE or the CM designed their layout. An example at IPP of how a laydown area was first allocated and then detailed was shown in Figure 5.3. One could confirm or refute my assumption by observing a person who is designing a layout and inquiring at that time about specific groupings of objects. This was not possible at IPP, since the project was near completion at the time of the interviews.

The Expert Strategy of SightPlan is the product of modeling how people lay out sites. Therefore, it also reflects the fact that people adjust their strategy of searching for a solution to cope with **human cognitive limitations**. I first became aware of those limitations when I set out to define the site layout problem (described in Section 2.1). I restricted the problem scope, because including all the variables that might play a role in layout generation would have led to an overly complex definition. Similarly, I defined site layouts as snapshots in time, yet they take place over spans of time.

A second manifestation of human cognitive limitations affecting problem-solving occurs *during problem-solving*. People can focus attention only 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 repeating the cycle with the next object. Once an object is positioned, it remains in that location and no further objects can be positioned in that location.

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 in space, partitioning over time, and aggregating in space, aggregating over time.

1 PARTITION IN SPACE

The CM partitions the site into four sub-areas, and decides for each laydown in which sub-area it should be located. In that way, the problem is split into several sub-problems, each involving fewer objects and fewer constraints. This method is acceptable in this situation, where objects in one sub-area do not have constraints with objects in other sub-areas, and thus sub-problems are independent of each other.

2 PARTITION OVER TIME

In this example, the layout of temporary facilities is divided in two phases. On phase consists of laying out the "long-term temporary facilities" and is the responsibility of the AE, the other phase consists of laying out the "long-term contractor laydown areas" and is the responsibility of the CM. Such a division over time is common practice in industry (although both tasks are usually performed by the CM). The rationale behind it is probably that, because the facilities laid out by the AE are on site for most of the duration of project construction, and remain on site after project completion, they are considered part of the permanent facilities, which the AE designs and lays out. Normally, permanent facilities are laid out before construction starts, so they can occupy the "best" space on site. SightPlan makes this division explicit, and encourages the reader to consider *merging* these two phases. This would result in the desirable situation in which layout considerations from the CM are taken into account at the same time the AE lays out the site. One could take this one step further and argue that time should be made an explicit variable in the model. For example, if a facility had an attribute specifying for what time period it is on site, SightPlan could deduce from this information whether or not the facility is "temporary" or "permanent."

The SightPlan model also makes explicit that the AE and the CM both have to identify site boundaries and objects with fixed location on site. This seems like a duplication of tasks, which could be avoided if both phases were merged. Note however that, although SightPlan represents these actions identically, the AE may perform them differently from the CM. Moreover, people need to process this kind of information in order to acquire background and context for problem-solving, so it may not be feasible to take the actions away from either the AE or the CM in order to eliminate this duplication.

3 AGGREGATE IN SPACE

Objects are aggregated in a partial arrangement before the partial arrangement is merged with or included in another one. An example of this is the layout of the ENTRANCE to determine the aggregate's size and shape before SightPlan places the aggregate in the DELTA-SITE arrangement.

Whether or not partitioning or aggregating space is worth doing depends on the constraints that objects have with respect to each other. For instance, constraints on parts can become constraints on the aggregate. The warehouses in the FACILITIES aggregate need to be adjacent to a railroad, therefore the FACILITIES need to be adjacent to a railroad. If an aggregate object contains many objects with many constraints to objects that are not part of the aggregate, then it may be ill advised to lay out this aggregate.

4 AGGREGATE OVER TIME

The CM uses a single site arrangement drawing for laying out the temporary facilities. This entails asynchrony, in that, when construction starts, obviously none of the buildings to be constructed are on site, and by the time construction ends, temporary facilities may have been removed from site! Yet, implicit temporal reasoning about changes of the site over time, with the use of only a single drawing representing space, is common construction practice.

The early-commitment approach is pervasive in the Expert Strategy, and it appears to work well on IPP, since SightPlan succeeds in generating a solution. In general, however, the construction method 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.* That is, multiple arrangements exist in which all the constraints on the objects are met. A problem is underconstrained when it has inherently few constraints that need to be satisfied; but one can also force a problem to be underconstrained by appropriately phrasing the knowledge about the problem's objects and constraints. The managers at IPP perceived the site as a spacious one, so 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 difficult to determine whether or not the IPP problem is inherently underconstrained. The information I gathered and the way in which I used it for representation, however, makes the IPP problem appear underconstrained. Two main aspects of this problem that affected my approach were:

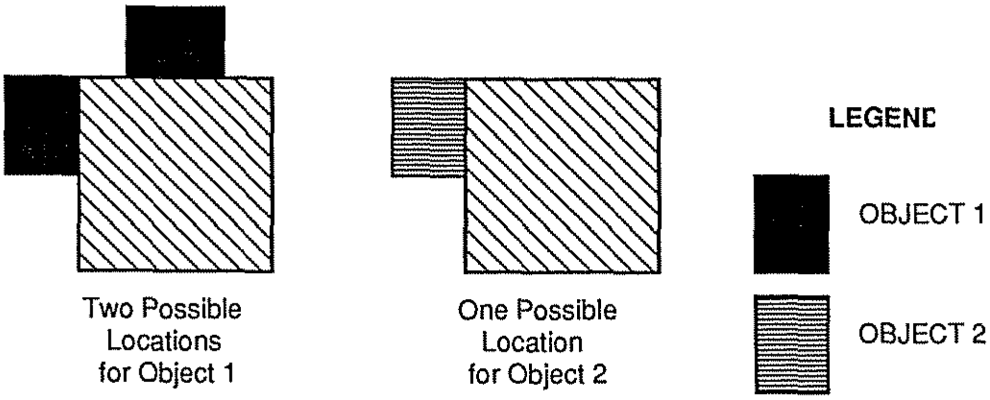
1 IT IS KNOWN AHEAD OF TIME THAT ALL OBJECTS WILL FIT ON SITE, AND ADDITIONALLY, THAT A SOLUTION LAYOUT EXISTS, BECAUSE A LAYOUT ARRANGEMENT FOR TEMPORARY FACILITIES WAS AVAILABLE (HINDSIGHT IS 20/20!)

The phrasing of the layout problem is biased by the fact that I acquired the knowledge after the site had been laid out. Moreover, because I defined objects' 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. Because of this pre-shaping "hack," I knew ahead of time that SightPlan would have the option to generate the expert's solution layout, and I could circumvent the problem of sizing and shaping facilities in this occasion. Flexibly reasoning about size and shape is one of the more interesting facets of site layout and requires more research.

2 IT IS KNOWN AHEAD OF TIME THAT ALL CONSTRAINTS CAN BE MET

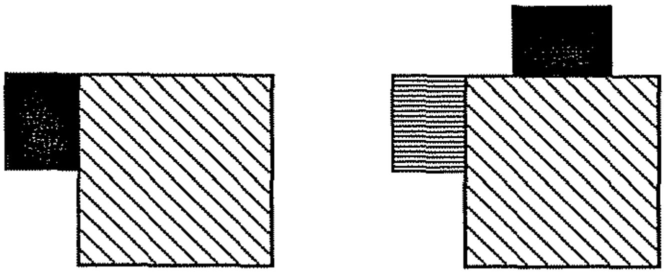
As the knowledge engineer who defined the constraints, I made sure that all constraints I specified could be met, and I defined constraints loosely so that there would be some alternative positions for objects. Also, at this time, the SightPlan model can "understand" only a limited vocabulary of spatial 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. I think that people have the tendency to add additional constraints in order to be able to differentiate among alternative solutions as problem solving progresses and many solutions appear to be possible. These constraints may not be known ahead of time; instead, they are defined opportunistically.

An example of how the early-commitment strategy might fail to find the *best* solution illustrates the **Instantiation Problem** (Figure 6.24). This problem arises, for example, when two objects have several possible locations, when the best position (Position 1) for the first object (Object 1) also is the single best position for the second object (Object 2), and when the first object has an alternative position (Position 2) that is as good as its best position . If Object 1 is positioned in Position 1, then Object 2 cannot be in its best position anymore. However, if Object 1 is positioned in Position 2, then both objects 1 and 2 can be at their best positions. This problem almost arose on IPP, where FACILITIES might have taken the position of ENTRANCE, before ENTRANCE was positioned on site, except that FACILITIES are just marginally closer to power unit 1 when located at their solution position.



Two Possible Locations for Object 1

One Possible Location for Object 2



Object 2 has NO Location

Both Objects have a Location

Figure 6.24: An Example of the Instantiation Problem Due to Early Commitment

AI researchers have identified these types of problems and have proposed several ways for 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. I will return to postponed commitment and investigate how that strategy affects problem-solving in Section 6.3. Other methods for improving the least-commitment strategy have not been investigated.

6.1.3.2 System Performance

System performance in terms of absolute speed has not been a concern in this research, so I did not spend time trying to optimize SightPlan's run-time. As a matter of interest, the AE and the CM Model each take a few hours to construct a solution. The BB1 architecture keeps information about each cycle's actions, events, and scheduler decisions around for bookkeeping purposes and to facilitate explanation and learning from looking at a system's run. The SightPlan Expert Strategy was designed only to capture the layout actions people take, not to optimize those actions. 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 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 I increased SightPlan's execution speed was by creating an object named OCCUPIED-SPACE to keep track of the space occupied by facilities with a single location on site. In this way I reduced expressing that no two facilities could overlap (factorially many constraints) to a single constraint between each facility and OCCUPIED-SPACE.

6.1.3.3 Validation

In this discussion of validation, remember that the primary goal of the SightPlan project is to explore AI architectures for solving the class of spatial arrangement problems to which site layout belongs. We are examining the *approach* of using carefully selected domain knowledge and a flexible reasoning mechanism to gain advantage over generic and more rigid heuristic construction methods. This goal opposes that of developing a narrow and deep expert system that could match and possibly exceed the performance of a human expert in site layout. The latter has never been our ambition.

If the validity of the Expert Strategy were gauged based on its *capability to generate a solution layout* that resembles the layout generated by the field expert, then the strategy has succeeded quite well. The reader can compare Figure 5.2 and Figure 6.23. The two layouts are not identical, probably because the SightPlan model failed to represent all the factors that the expert used to generate his layout and because SightPlan fairly quickly decides to pick positions that meet the preference constraints. Also, it is hard to tell which of the arrangements is the better one, or in general, how *good* a satisfying arrangement is unless additional objective criteria are brought in. I asked the expert several times how he would evaluate a site layout, but he had no guidelines to offer for such an evaluation.

If the validity of the Expert Strategy were gauged based on *how well it represents the actions taken by a field expert*, then that validity is difficult to assess, because I do not have records on exactly how the expert designed the layout.

If the validity of the Expert Strategy were gauged based on how well its *layout process can be understood by practitioners*, then SightPlan has definitely succeeded. I showed SightPlan to the Field Construction Manager at IPP—whom I consider a novice to AI programming techniques—and found it easy to make clear why SightPlan would take each step, and to get his involvement for validating the model. I asked the Field Construction Manager to critique SightPlan in operation, and he pointed out several instances of ignorance on the part of SightPlan. In one case, a location that SightPlan had chosen proved to be inaccessible with large trailer-trucks because a railroad would have to be crossed. In another case, a building assigned for one use by the AE was reassigned for another use by the CM. When one considers, however, how little knowledge SightPlan has about construction management it is remarkable that the system can do a fairly good job of modeling an expert's problem-solving strategy and generating a solution layout, using only limited amounts of domain knowledge. The domain knowledge that it contains permits SightPlan to lay out a site in a way more similar to the way people do it than would a generic strategy, but its lack of complete domain knowledge keeps SightPlan from taking the same actions a field expert would.

Clearly, SightPlan would gain more power if it were augmented with additional domain knowledge. For example, SightPlan should acquire knowledge on how to define an aggregate object. The links between an aggregate and its parts as shown in Figure 6.6 are currently predefined. SightPlan might be able to derive the aggregate/part relationship by knowing about typical arrangements on construction sites (by looking at previous layouts or by searching a library of standard arrangements). Alternatively, SightPlan might derive this relationship by reasoning about what objects have in common, such as attributes and links (similar constraints, proximity constraints) or functions (for example, contractor fabrication areas all need access to a power supply). SightPlan should also learn how to size and shape facilities. Further interviews will help determine what the available knowledge is.

Finally, if the validity of the Expert Strategy were gauged based on *how general the Expert Strategy is*, then one can investigate the validity by applying the Expert Strategy to another site. Section 6.2 discussed the application of the Expert Strategy on American 1.

6.1.3.4 Extensions

In order to make it easier to explain the actions of the expert strategy, I augmented SightPlan with a graphical display. This proved to be very useful, especially when I was explaining the system to people not familiar with BB1 or rule-based systems, as was the case with the Field Construction Manager who validated my system. It is easy for a field expert to use visual inspection as a means for assessing the quality of a layout

("This arrangement does not look quite right," or "That looks good") whereas it may be difficult for the expert to make such judgements without visual reference.

In order to encourage visual inspection by a user, Tony Confrey developed a color graphical display, and he took the idea one step further by developing an interactive graphical display, named SightView. Because SightView interacts with SightPlan, a user can now interact in real-time with the system in two ways: One way is by using the interactive graphics. The other way is by overriding scheduler decisions, a feature native to the BB1 architecture. I have not studied user interaction with the system and its impact on problem solving, but would like to suggest it as a promising area for further research.

In order to asses the generality of the Expert Strategy that SightPlan had learned on the Intermountain Power Project, SightPlan is next applied to American 1.

6. 2 Expert and Temporal Strategy on AM1

6.2.1 Description and Scope of the Expert Strategy on AM1

The second SightPlan model lays out the "Temporary Structures" and "Laydown Areas" on the American 1 (AM1) site (these were identified in Table 5.3). The strategy for laying out this site is the Expert Strategy that was developed on IPP. Applying the Expert Strategy to a second project provides a basis for its validation. The implementation of this Expert Strategy according to which SightPlan lays out all facilities on the site considered at one time interval encompassing the entire project duration is given in Section 6.2.2. A discussion of the validity of the Expert Strategy follows in Section 6.2.3.

The Expert Strategy was then extended to reason about the layout of the site at different time intervals. Under this so-called *Temporal Strategy,* SightPlan is not restricted to laying out a single site arrangement that encompasses the entire project. Instead, SightPlan can generate arrangements that cover the layout over a series of time intervals, each covering part of the project duration. This Temporal Strategy extension to the Expert Strategy is described in Section 6.2.4, its implementation is outlined in Section 6.2.5, and the results and possible extensions are discussed in Section 6.2.6.

6.2.2 Implementation of the Expert Strategy on AM1

The skeletal plan used to guide the layout of IPP was pruned down for application to AM1. This simplification was possible thanks to two characteristics of this second project. First, a single company was involved in designing, engineering, and constructing AM1, so the duplication of tasks between the AE and the CM was not necessary. Second, because AM1 is so much smaller in scope than IPP, the need for aggregating facilities and laying them out in separate arrangements before positioning them on the site did not arise. Figure 6.25 shows the reduced skeletal plan. The reader can compare it with the full plan given earlier in Figures 6.1 and 6.2. The corresponding foci in each of the plans are labeled with the same number, so, from the discontinuity in the numbering sequence, it is clear that the foci duplicated in Figure 6.1 were left out in Figure 6.25. Table 6.2 outlines the major actions of SightPlan's reduced Expert Strategy applied to AM1.

CYCLE	ACTION
14	create pa
19	include context
24	include fixed objects
29	include and identify occupied-space
33-35	include areas
40	include facilities
44	orient pa
50-56	position facilities inside and outside of areas
63-64	position largest object
67-75	position second largest object
78-84	position last object
62-84	position facilities
94-100	include laydowns in pa
107-119	position laydowns in area
126-128	position largest aydown
130	update occupied space
131-133	position second largest laydown
135	update occupied space, and so on
146	all laydowns positioned

Table 6.2: Some Cycles from SightPlan's Expert Strategy Applied to AM1

SightPlan creates a partial arrangement (Cycle 14) and includes the context (Cycle 19). This time the context is the AM1 site in King City, California. All objects that have a predefined and fixed location on site are included (Cycle 24). SightPlan then determines the occupied space (Cycle 29), and divides the site up into sub-areas: the LAYDOWN-AREA and the FENCED-AREA (Cycle 33-35). SightPlan includes the construction facilities that will be on site for the entire duration of construction of the project but that have no position yet (Cycles 40). These facilities (FAB-SHOPS, MGT-

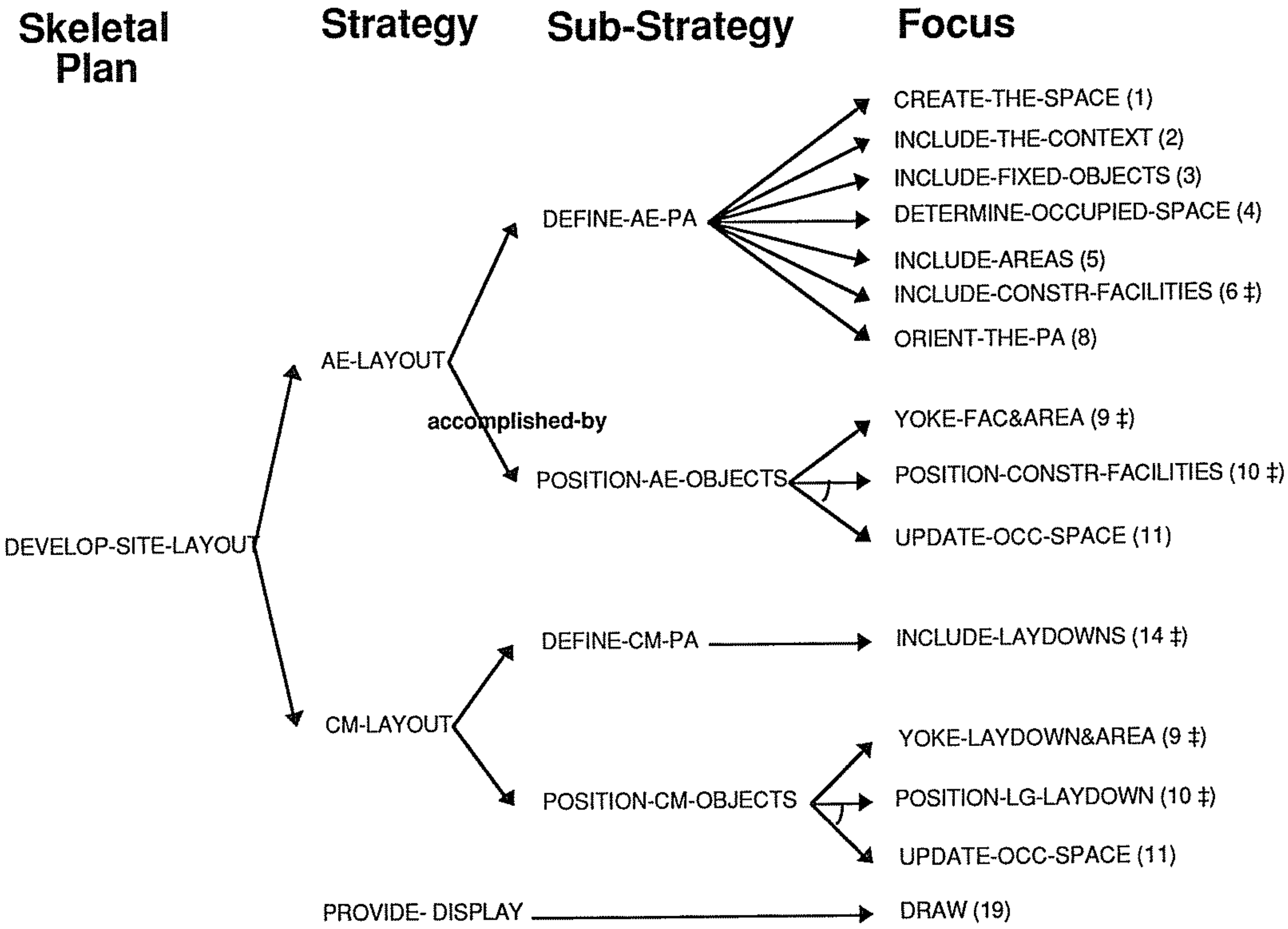


Figure 6.25: Reduced Skeletal Plan of the Expert Strategy Applied to AM1
The numbers in parentheses label the foci so that this skeletal plan can easily be compared to those of Figures 6.1, 6.2, 6.31 and 6.41.

TRAILER, and PARKING) will be referred to as *long-term facilities* in the remainder of this section 6.2. The system ends its definition of this partial arrangement by orienting it (Cycle 44, Figure 6.26).

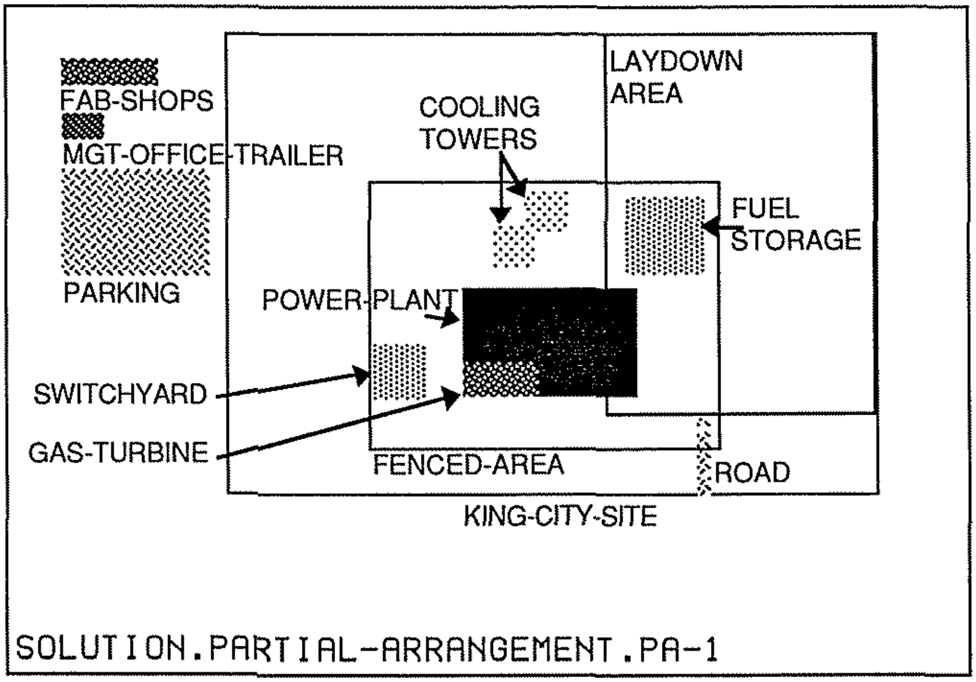


Figure 6.26: Permanent Facilities on AM1

SightPlan zones each of the facilities as needed inside or outside the sub-areas of the site (Cycles 50-56). After all these constraints are met, the system pursues its early-commitment strategy in picking one facility and meeting all constraints between that facility and objects with fixed location on site (Cycle 63). Finally, SightPlan selects a single position by satisfying that facility's preference constraint (pick the position as close as possible to another fixed facility or to a fixed object on site, or pick one position out of the facility's set of possible positions at random) (Cycle 64). The occupied space is updated (Cycle 65) and the process repeats for the following facility, until all facilities have a single position in the layout (Cycles 63-84, Figure 6.27).

When all long-term facilities are uniquely positioned on site, SightPlan includes the laydown areas in the arrangement (Cycles 94-100). These, in turn, are positioned on the site in the same manner that the long-term facilities were positioned (Cycles 107-145). When that is done, SightPlan has completed its task. Figure 6.28 shows SightPlan's solution layout.

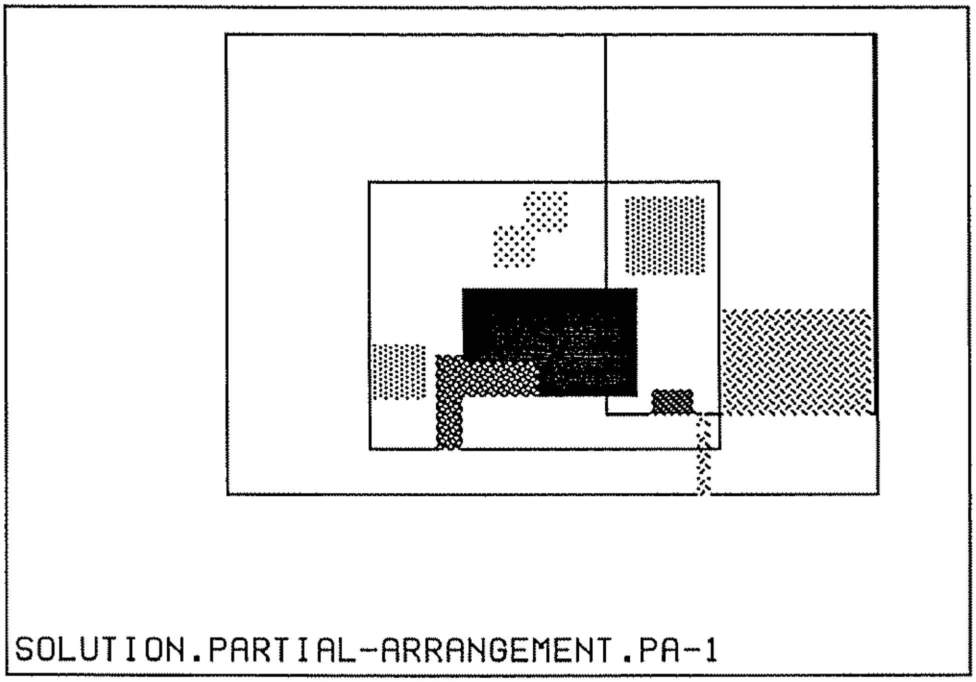


Figure 6.27: AM1 with Permanent and Long-term Temporary Facilities

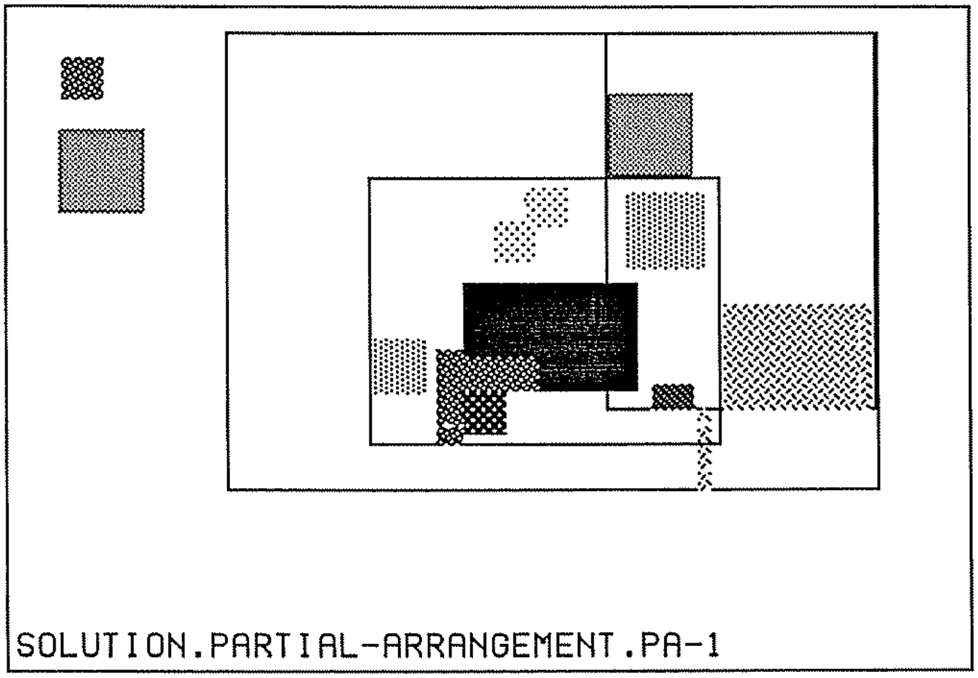


Figure 6.28: Solution Arrangement of SightPlan Laying out AM1

6.2.3 Validation of the Expert Strategy on AM1

The SightPlan system, which initially laid out the IPP site, was easily adapted to tackle the AM1 site:

1 **A NEW PROBLEM-KB REPLACED THE OLD ONE.**

The new problem brought the AMERICAN1-BB and the CONSTRAINTS-BB to replace the INTERMOUNTAIN-BB and its CONSTRAINTS-BB. In SightPlan's representation, Intermountain has about 50 permanent facilities, road and railroad segments, and 40 temporary facilities and long-term laydown areas; American 1 has about 10 and 10 respectively.

2 **THE EXPERT STRATEGY WAS REDUCED TO ENTAIL ONLY A SUBSET OF ITS FOCI.**

Note that the strategy depends on the project-type but not on the specific problem, so that no changes need to be made. This reduction was not necessary for the operation of SightPlan, but it did improve SightPlan's efficiency. If all original foci had been maintained in the AM1 system, some of them would be activated by SightPlan in one cycle and deactivated immediately in the subsequent cycle. This is because their goal is always true in this particular problem. When that is the case, none of the executable domain actions get priority for execution, thus, the control actions do not effectively control other executable actions and may therefore just as well be taken out of the system.

3 **ALL OTHER KBS AND FUNCTION DEFINITIONS IN SIGHTPLAN REMAINED UNTOUCHED.**

That this adaptation went so smoothly demonstrates the generality of the SightPlan approach. A hierarchy of BBs customized with a site-specific layer of project information, combined with a flexible reasoning mechanism fulfills the basic requirements for a construction site layout system, and the constructive assembly method provides the backbone for SightPlan's approach. Moreover, I found that the control strategy developed for the IPP site did transfer and could be applied to AM1 site without major modifications. Thus, I succeeded in formalizing a domain-dependent layout strategy that, despite its simplicity, had the potential to be generally applicable to power plant sites, and possibly to other construction sites. In order to assess this generality, one needs to consider more case studies, and it is quite likely that SightPlan's heuristics will need to be fleshed out for the system to deal with a larger variety of site situations.

The Expert Strategy is, on this second project also, valid in terms of its success in generating a layout that resembles the field expert's layout. This conclusion is similar to the one I arrived at on the IPP project. SightPlan could be extended with more reasoning about constraints and possibly about objective criteria to guide its heuristic pruning of the explored solution space in order to arrive at more satisfying solution layouts.

So far, SightPlan's Expert Strategy has been to:

1 PARTITION IN SPACE

Zoning constraints on facilities bring partitioning in space. On the AM1 project, however, whether a facility is located inside or outside of the fenced area is a matter of site protection against theft more than a means to divide a large set of facilities up in smaller sets of facilities to be grouped in a sub-area of the site.

2 PARTITION OVER TIME

By first locating the more static facilities that are on site for a longer time (AE before CM), one at a time, and to commit such a facility to its position once one had been picked. Long-term facilities, however, were excluded from prime locations (such as inside the WORK-AREA on IPP or enclosed by the POWER-PLANT buildings on AM1). In that way, more important short-term facilities could still get good locations even though they would be positioned later in the layout process. (The layout of short-term facilities is not tackled by SightPlan.)

3 AGGREGATE IN SPACE

Aggregating objects into larger arrangements was not needed on the AM1 site because fewer objects need to be positioned.

4 AGGREGATE OVER TIME

By generating a single arrangement drawing that shows the positions of all objects and facilities as if they were on site for the entire project duration.

No two facilities could overlap in SightPlan's single layout that encompasses the entire project duration. In reality, however, facilities might make use of the same space on a construction site if they do so for time intervals that do not overlap. In that way, prime space on site can be allotted to satisfy more than one user. In order to reason about multiple uses of areas over the duration of project construction, I extended SightPlan's set of control and domain KSs to provide the capability to reason about site layouts that change over time.

6.2.4 Description of SightPlan's Approach for Reasoning about Layouts that Change over Time

Thanks to 1) the limited scope of AM1 in terms of the number of temporary facilities and long-term laydown areas, 2) the centralized control that the AEC firm had, and 3) the short (1.5 years) total project duration, the project manager could address the timeliness of various facilities on site while explaining the layout plan to me. As Tables 5.4 and 5.5 show, different facilities served over time the function of Construction Office, and one laydown area in particular was in high demand by various contractors and thus was reallocated as construction progressed. SightPlan's Temporal Strategy specifically deals with this latter type of flexibility in the reallocation of space over time, although it does not necessarily represent the actions taken by an expert. This section explains SightPlan's reasoning about layouts that change over time, and refers to the AM1 project to illustrate the concepts. The following section describes the actual implementation of this approach as it is encoded in the Temporal Strategy.

The Temporal Strategy is similar to the Expert Strategy in terms of the early commitment it pursues. An important addition is that all objects in the layout as well as the construction project itself have as additional attributes the time period during which they occupy space on site. It is this time period that allows SightPlan to decide when to include an object in an arrangement and in which arrangement to include it. This added complexity opens up a venue of possible methods that SightPlan could pursue for the generation of its layouts. I describe here only one simple implementation of the system, but I plan to pursue this line of research further in the future.

Figure 6.29 shows four frames from the AMERICAN1-BB of the PROBLEM-KB with their attributes referring to time: 1) the AMERICAN1 project itself, with its start- and end-construction dates; 2) the GAS-TURBINE, a *permanent* facility. A permanent object has, by definition, a demobilization date that is NIL; 3) the FAB-SHOPS, a *long-term* facility. A long-term object is, by definition, one that is on site for the entire duration of construction. That is, its mobilization date coincides with the start date of construction of the project, its demobilization date coincides with the end date of construction; 4) the LUMBER, a *medium-term* laydown area. A medium-term object is, by definition, one that is on site for only part of the duration of project construction. That is, either its mobilization date comes after the start date of construction of the project, or its demobilization date comes before the end date of construction, or both dates fall between the start and end of construction. "Attrs: AAA" and "lnks: XXX" symbolize the other attributes and links the objects might have.

When a partial arrangement (PA) is created, it is associated with a time period. By default this period is the duration of project construction. Before an object is included in a PA, the KSAR that was generated from SightPlan's domain KS SITUATE-OBJECT for this object checks whether or not the time period for which the object is on site coincides with the time period covered by the PA.

If it does, this KSAR adds the object to the list in the attribute **situated-objects** of the PA. The resulting new state makes the KSAR that proposes to include the object in a PA executable (this latter KSAR was generated from the INCLUDE-OBJECT domain KS). In the subsequent BB1 cycle, the object can be included in the PA.

AMERICAN1

level:
AMERICAN1.PROBLEM

attributes:
start-construction: 870901
end-construction: 890531
attrs: AAA

links:
lnks: XXX

GAS-TURBINE

level:
AMERICAN1.POWER-PLANT-FACILITIES

attributes:
mobilize: 870901
demobilize: NIL
attrs: AAA

links:
lnks: XXX

FAB-SHOPS

level:
AMERICAN1.CONSTRUCTION-FACILITIES

attributes:
mobilize:870901
demobilize: 890531
attrs: AAA

links:
lnks: XXX

LUMBER

level:
AMERICAN1.CONTRACTOR-LAYDOWN-AREA

attributes:
mobilize: 871201
demobilize: 880430
attrs: AAA

links:
lnks: XXX

Figure 6.29: Four Objects on the AMERICAN1-BB with Attributes Describing their Construction Period (from start-construction to end-construction) or their Time Period on Site (from mobilize to demobilize)

If it does not, the KSAR from SITUATE-OBJECT finds the PA or PAs with a time period that overlaps with the object's time period. If such a PA's time period is an integral part of the object's, then the KSAR adds the object to that PA's **situated-objects** attribute. If such a PA's time period overlaps only partially with the object's, then the KSAR creates new PAs in the following way. First, one new PA obtains the time period of the former PA that did overlap with the object's. This new PA also gets the object in its attribute **situated-objects**. Second, at least one and at most two other new PAs—depending on what is needed—obtain the segments of the time period from the original PA that did not overlap with the object's. Further, these two or three new PAs inherit all the attributes and links from their originating PA. When these new PAs are created and because the object is now part of a **situated-objects** attribute, this new state makes the KSAR that proposes to include the object in a PA executable, and in the subsequent BB1 cycle the object can be included in each of the PAs that list the object in its **situated-objects** attribute.

Figure 6.30 shows a simple example of this mechanism by providing the relevant attributes and links of the PAs that are created in the process. PA1 is the partial arrangement solution that covers AM1's construction duration. LUMBER is a laydown area that is needed for only a few months of construction; Figure 6.25 showed its frame. Starting from PA1, SightPlan creates PA3 to cover the period that this laydown area is on site, and PA2 and PA4 to cover the remaining time periods. "attrs: AAA" and "lnks: XXX" symbolize the attributes and links with their values on PA1 that SightPlan makes part of the other PAs upon their creation.

When all objects are included in their respective partial arrangement(s) with a matching time frame, SightPlan can reason about their layout. SightPlan selects the first object and meets its constraints with the objects in each of the partial arrangements that include it. The constraint that is met last is the one that allows the system to pick a single position for the object in all of its partial arrangements. This is so that the object remains in the *same* position as construction progresses and the time frame moves on from one PA to the next. Future versions of SightPlan may relax this constraint and allow objects to relocate over the project duration. After updating the occupied space in each of the affected PAs, SightPlan proceeds with the next object until all objects have a fixed location, and the set of layouts is thus complete. The following section describes the implementation of this Temporal Strategy for the layout of the AM1 project.

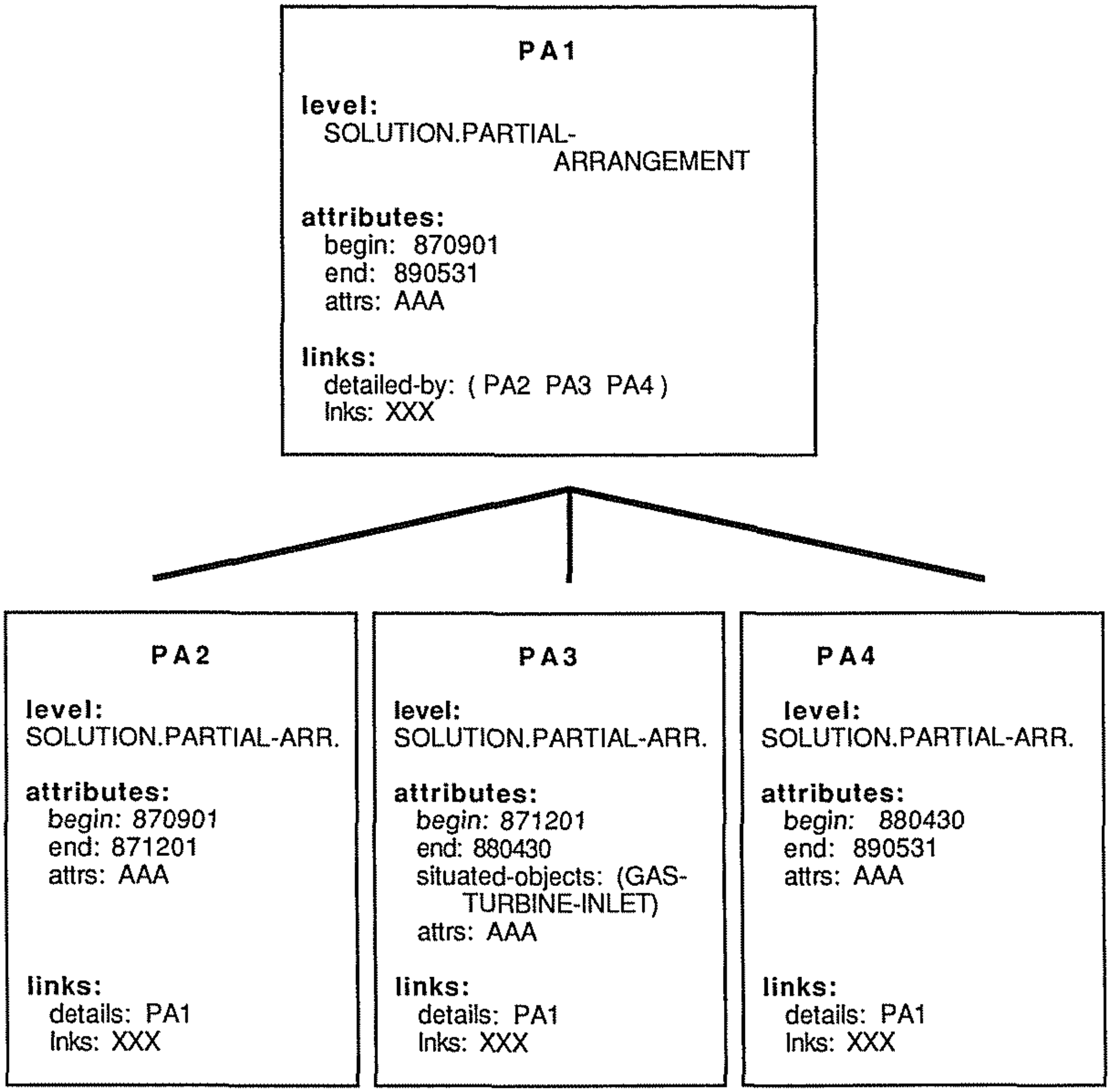


Figure 6.30: Division of a Partial Arrangement
into Partial Arrangements with Shorter Time Intervals

6.2.5 Implementation of the Temporal Strategy on AM1

The implementation of the Temporal Strategy consisted of attributing objects on the AMERICAN1-BB a time interval (see Figure 6.25), modifying KSs to take such time intervals explicitly into account, and adding a few domain and control KSs to the SightPlan system that was described in Section 6.2.3. I only included a few of the site objects in the BBs so that SightPlan would not bog down in computation.

In the SightPlan systems that I have discussed in Sections 6.1.2 and 6.2.2, 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 implementing the Temporal Strategy, such references to objects at a level are restated as conditions on the time interval, 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 exceeding 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.

Figure 6.31 shows the Skeletal Plan of the Temporal Strategy followed by SightPlan. The main differences with the Expert Strategy of Figure 6.22 is the SITUATE-OBJECT focus. This focus assigns a high rating to actions that correlate objects with PAs based on their time-related attributes, and runs in parallel with an INCLUDE focus. (This parallellism is marked by the connection between the arrows pointing at foci in Figure 6.31). Objects in the layout can now be distinguished by their time period on site, so knowledge sources can refer to time attributes to select objects rather than refer to object types. Thus SightPlan can distinguish short-term from long-term objects more sensibly, which therefore SightPlan is capable to lay out permanent as well as temporary facilities.

Table 6.3 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).

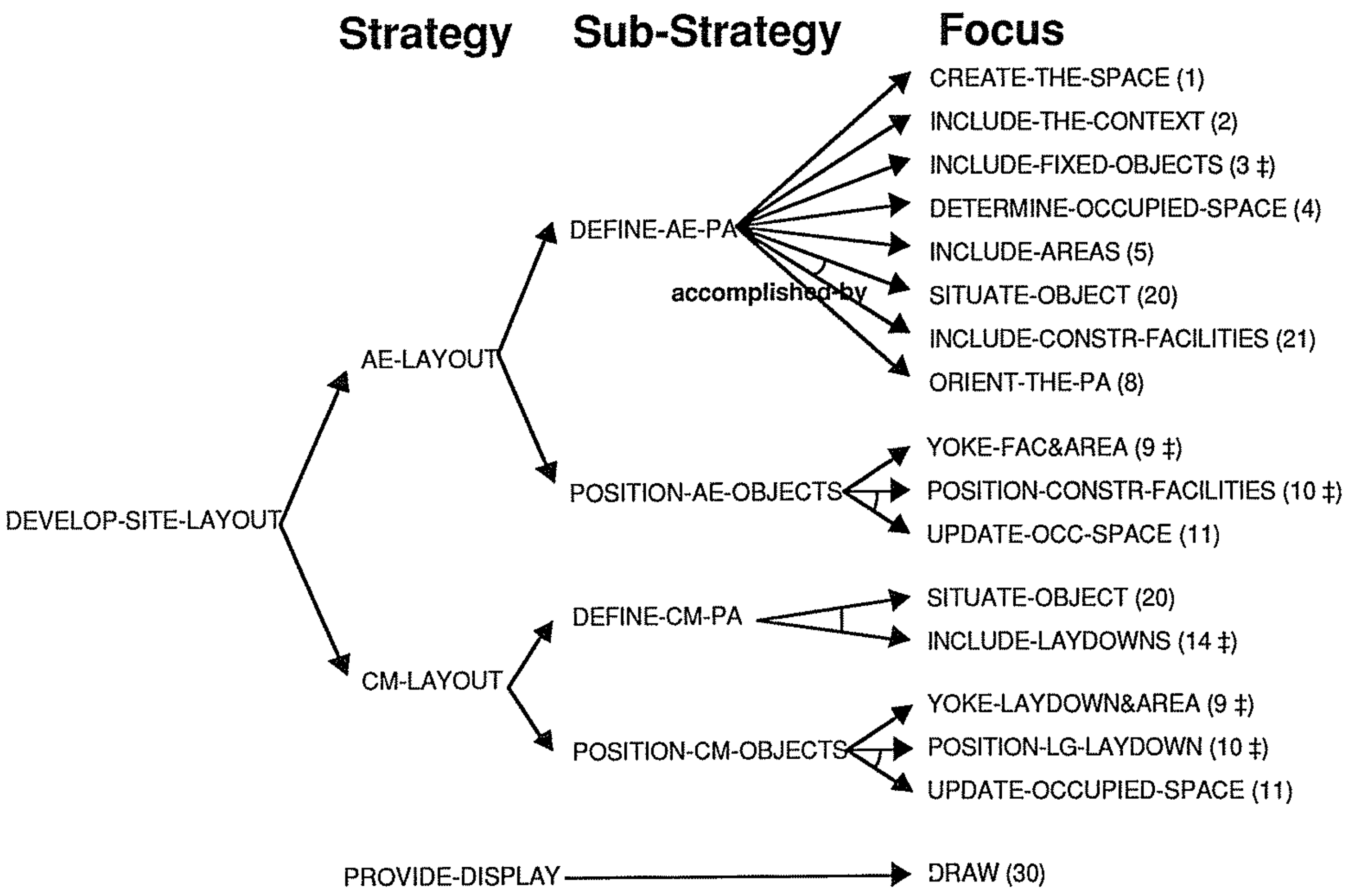


Figure 6.31: Skeletal Plan of the Temporal Strategy applied to AM1. The numbers in parentheses label the foci so that this skeletal plan can easily be compared to those of Figures 6.1, 6.2, 6.25, and 6.41. A focus with number labeled "‡" is similar to the one with corresponding number in the other figures, except for some small changes.

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 6.3: Some Cycles from SightPlan's Temporal Strategy Applied to AM1

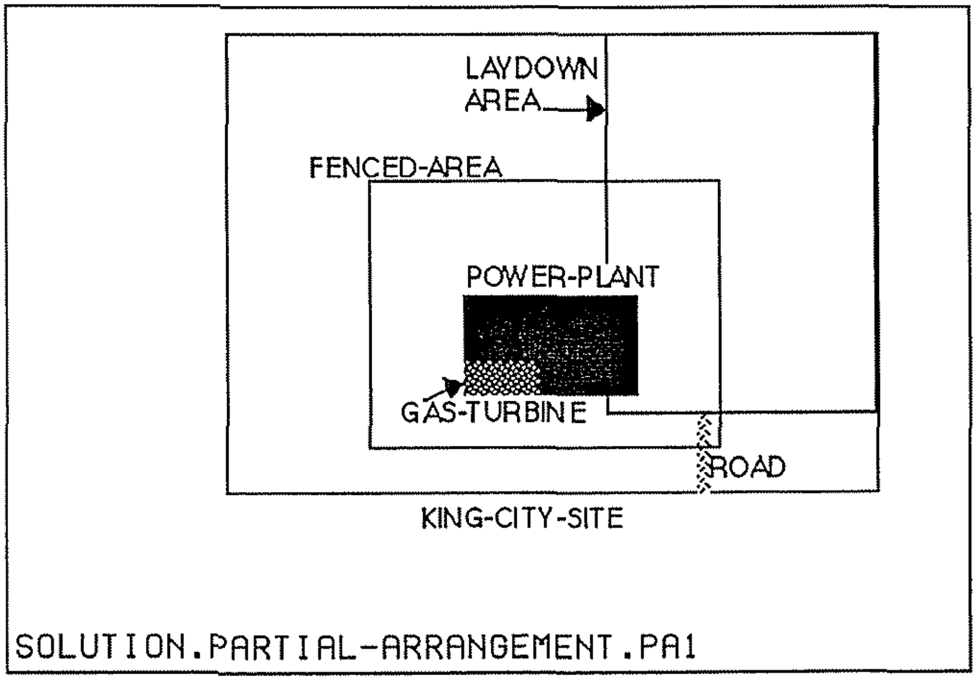


Figure 6.32: SightPlan defined Site Boundaries and Positioned Facilities on AM1

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).

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 6.33 through 6.39. These PAs show different site layout stages over the duration of construction of the AM1 project. Figure 6.40 outlines the time frame of each layout with respect to the entire construction duration.

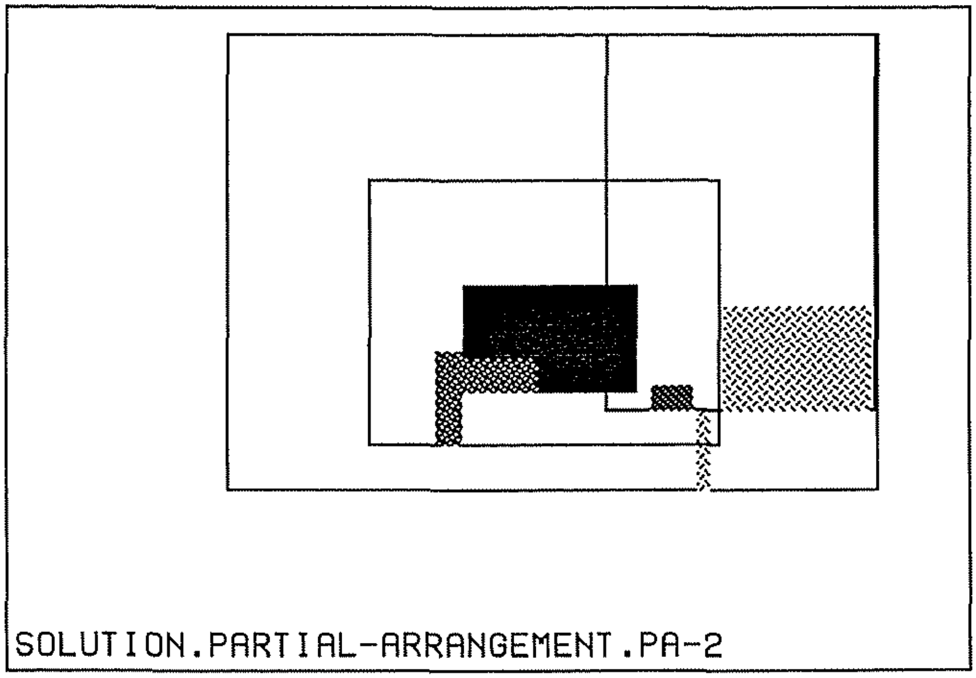


Figure 6.33: Construction Time Sequence of Solution Layouts
Generated by the Temporal Strategy Applied to AM1
One area is used by three contractors over time.
PA2 with Time Frame 870901–871201
No short-term laydown areas are on site.

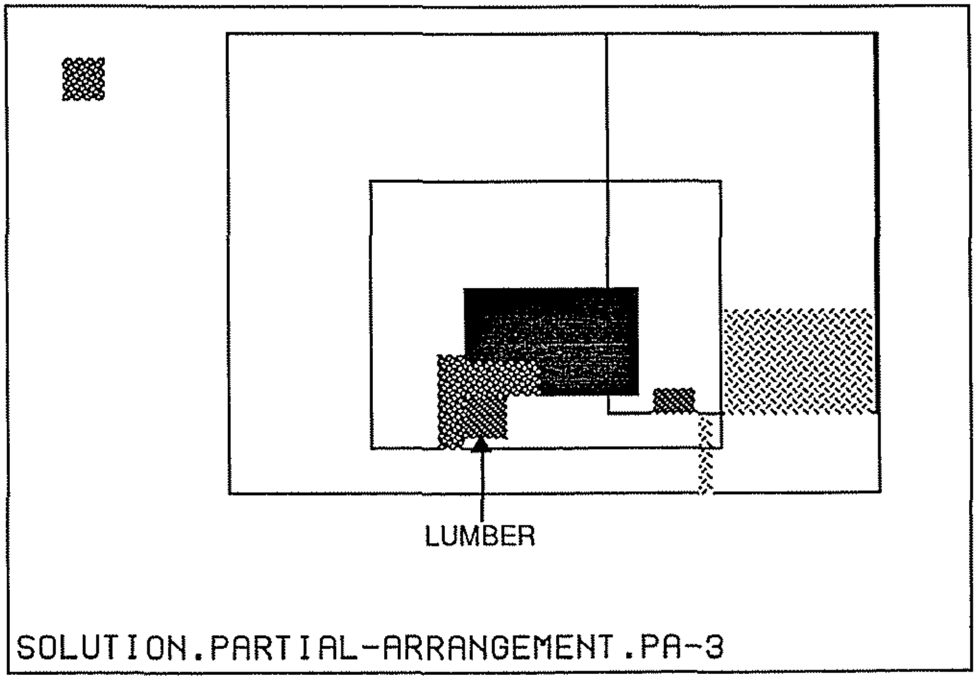


Figure 6.34: PA3 with Time Frame 871201–880430
including the LUMBER short-term laydown area.

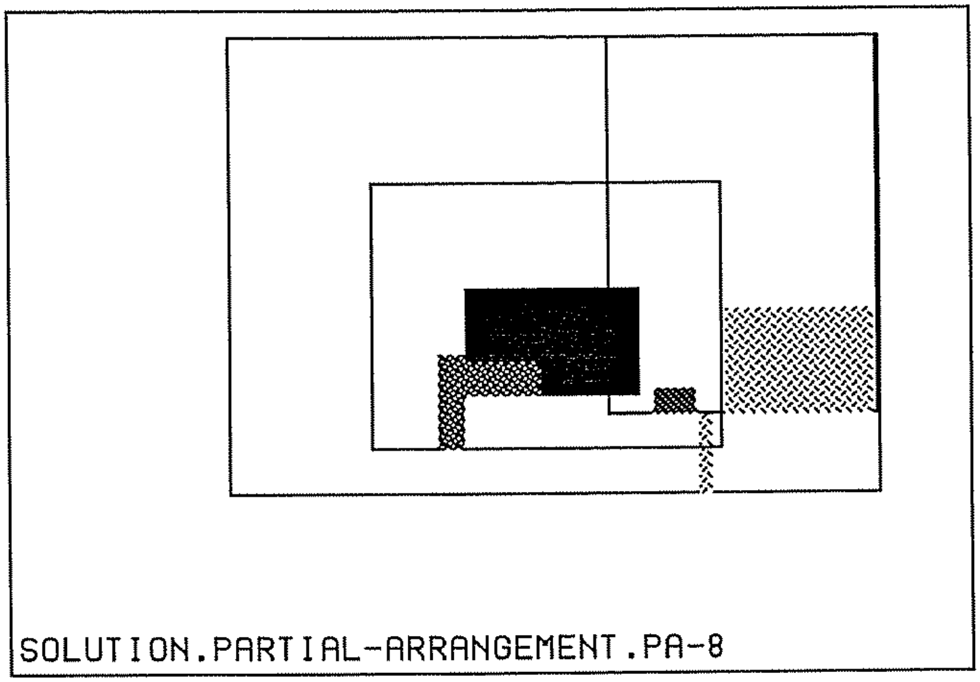


Figure 6.35: PA8 with Time Frame 880430–880501
No short-term laydown areas are on site.

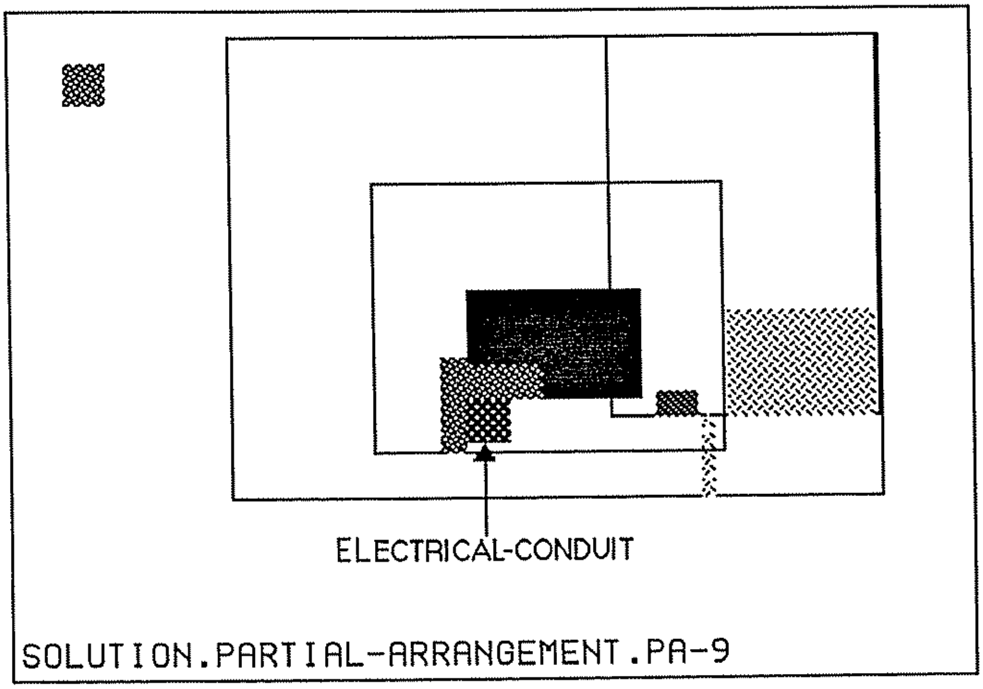


Figure 6.36: PA9 with Time Frame 880501–880630
including the ELECTRICAL-CONDUIT short-term laydown area.

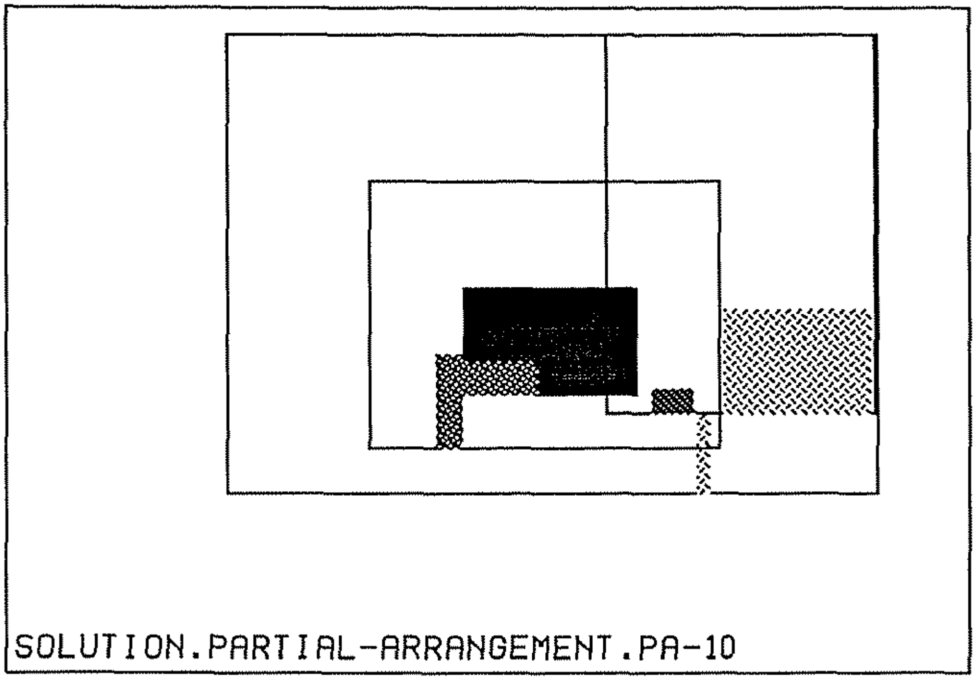


Figure 6.37: PA10 with Time Frame 880630–880701
No short-term laydown areas are on site.

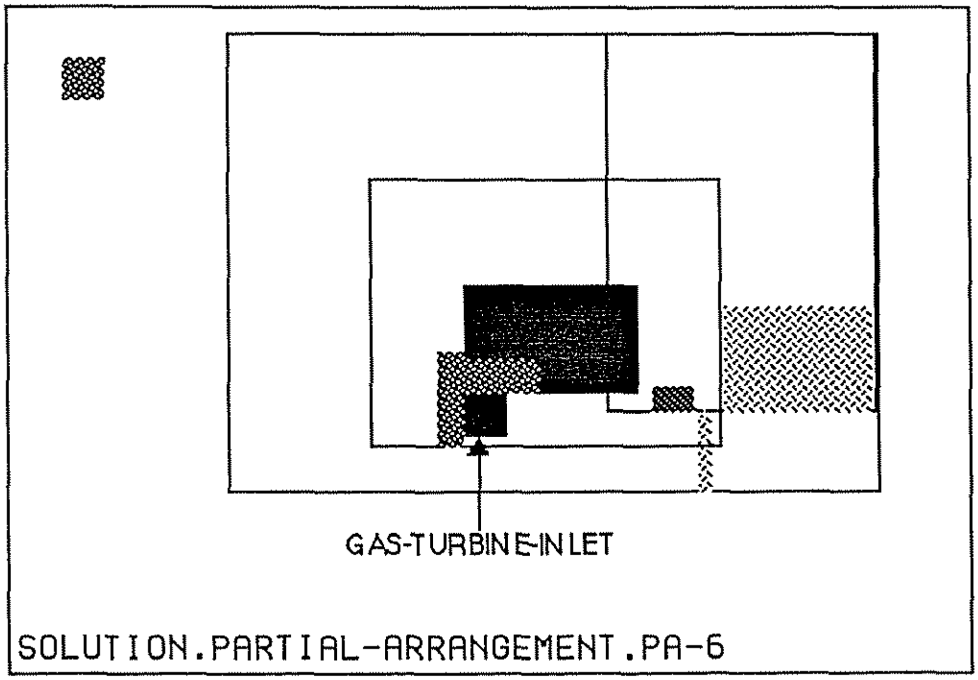


Figure 6.38: PA6 with Time Frame 880701–881231
including the GAS-TURBINE-INLET short-term laydown area.

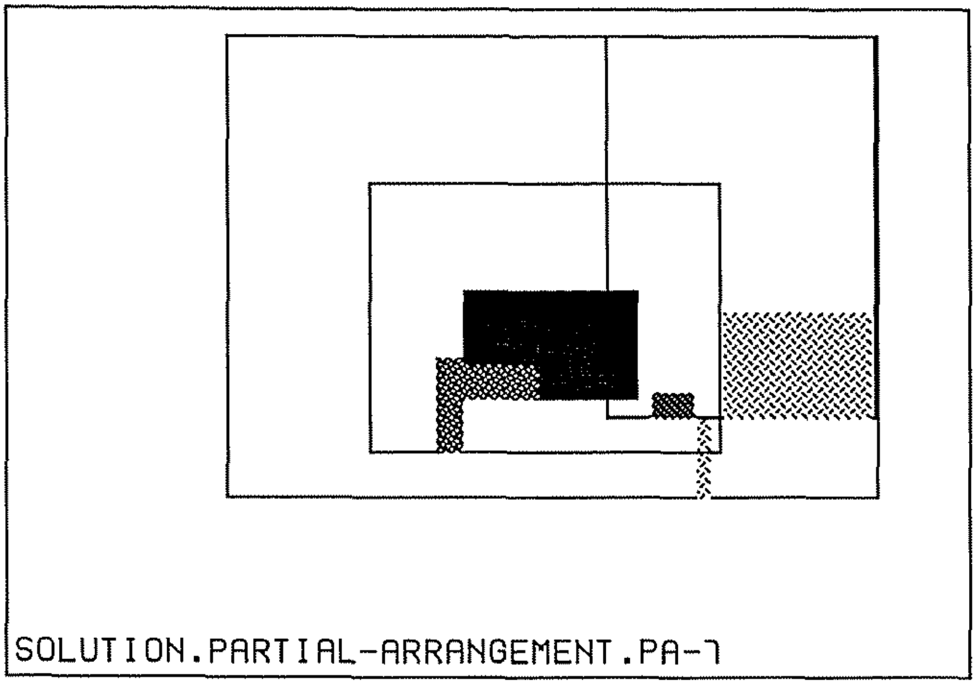


Figure 6.39: PA7 with Time Frame 881231–890531
No short-term laydown areas are on site.

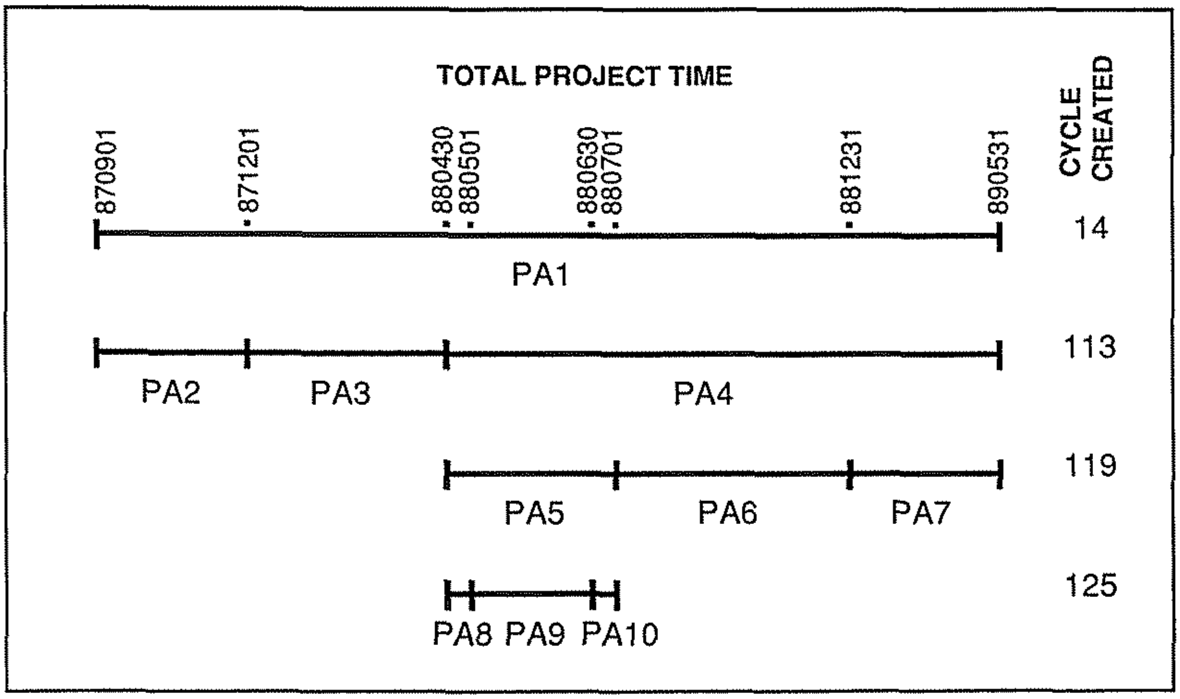


Figure 6.40: Partial Arrangements Laying Out the Site
dividing Total Project Time into Time Intervals.

6.2.6 Discussion of the Temporal Strategy on AM1

6.2.6.1 Layout Method

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 resources other than space and time. In the current system I have barely touched upon temporal reasoning. My future research will explore this area further.

This model was not intended to—and does not—model step-by-step the way people reason about layout changes over time. People tend to design layouts *in a time sequence* parallelling construction time. Because there is more precise information available about the near future than there is about the distant future, and the near future is of more immediate concern, early layouts are likely to get more attention and are probably developed in greater detail than are later layouts. This is not to say that major and critical factors that affect the layout in the long-term future would be ignored early on. Typically, they are taken into account early on, or provisions are made to avoid anticipated problems.

SightPlan could—but need not—follow this approach. Instead, when given sufficient knowledge, it can help identify most constrained layout phases at any stage of construction, generate layouts for those, and back up from there to generate layouts at (often less constrained) preceding phases. Thus, SightPlan provides for greater flexibility in approaching the layout problem. Moreover, as opposed to people drawing the sequential layouts over one another on a single site arrangement drawing, SightPlan easily duplicates arrangements and displays the layout at the various phases. It plays them back like a time-lapse recording, showing the layout frames at a fraction of their duration and in sequence, simulating the evolution of the site over time. This representation greatly reduces the perceived complexity of layout drawings.

The Temporal Strategy 1) requires that objects be associated with a single time interval for which they are on site, 2) operates on discrete time intervals, and 3) inherently assumes that objects are static. That is, if an object appears in one location in one PA and it is included in other PAs, then it appears in that same location there. Furthermore, the Temporal Strategy pursues a kind of early commitment; as soon as an object is included in a PA, it is positioned and fixed in a single location. Albeit that the current implementation is very simple, the resulting layouts show the reuse of prime space and, therefore, the solution generated by the Temporal Strategy is an improvement over that of the Expert Strategy. I expect that adding flexibility in the temporal reasoning will result in even better layouts.

The Temporal Strategy succeeds on the small problem that I implemented. However, computation may soon blow up when more objects and time intervals come into play. SightPlan would then need knowledge about critical phases and milestones in the construction schedule to identify what intervals are critical for layout, and it may resort to another method to deal with problems of different scale.

6.2.6.2 Extensions

SightPlan's approach to temporal reasoning is very simple. There are many possible alternatives to it, several 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 for 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, the space they occupy could be included gradually in the PAs. Finally, the total length of time that a facility is on site and the facility's size may not solely determine its criticality. When other factors are more important than time and space, they should be introduced in the model. The reader may get excited about these challenging, yet doable, areas for future research.

The Expert Strategy of SightPlan, developed on the Intermountain case study described in Section 6.1, and validated on the American 1 case study described in Section 6.2, was the product of modeling how people lay out sites, and, therefore, it reflected how people adjust their solution strategy to cope with human cognitive limitations. Our question was: What would the strategy be like if these human cognitive limitations were removed? Or, stated differently: What strategy would a computer use to construct a layout given *its* strengths and limitations? This strategy is described next, and it is called the Computational Strategy of SightPlan.

6.3 Computational Strategy on IPP

6.3.1 Objectives and Scope of the Computational Strategy on IPP

The third SightPlan model 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 count on gaining insight into the type of cognitive support computers may give to their users.

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 model. The two models make use of the constructive assembly method to generate solutions. I further tailored 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 the Expert Strategy would assign a low rating to them so that the new possible actions would not affect its problem-solving steps. Because I am not concerned with the system's absolute execution time, the additional cost incurred for having these constraints and KSs in the system is irrelevant. Also, in this comparative study, I will not account for the cost of rating, executing, and scheduling actions in this comparison, because these actions are part of the native BB1 architecture; optimizing their efficiency should not be the task of an application developer.

How could one strategy be *better* than another? First, one strategy could *generate the same results as another, but more efficiently*, such as in a shorter computation time. For example, SightPlan might improve its efficiency by performing control reasoning about constraints. The factors that might 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.

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. Moreover, 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 6.3.2, which gives the detailed formulation of the Computational Strategy.

6.3.2 Formulation of the Computational Strategy on IPP

6.3.2.1 Least Commitment

My first idea was to craft the Computational Strategy so that it would follow a *least-commitment strategy*. A least-commitment strategy is one where commitments are postponed for as long as possible. The commitments that can be postponed are those related to satisfying preference constraints, so I removed the AS-CLOSE-AS-POSSIBLE constraints from the Expert Strategy. Upon further inspection of the Expert Strategy, it became obvious that ZONING constraints express a kind of preference as well, so I removed them from the Expert Strategy. I 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 SightPlan generates them from a layout with multiple positions for objects by picking a unique position for each object. Furthermore, I changed the Expert Strategy to reorder the ranking of constraints. The new ranking was based on the characteristics of SightPlan's constraint engine. The reader interested in assessing computational costs that characterize the GS2D constraint engine, and concerned about the generation costs of coherent instances and the evaluation costs of arrangements, may consult Appendix C.

As a result of the least-commitment strategy, the problem became so underconstrained that the Computational Strategy was limited to 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 inspecting some of the solutions obtained, it appears that least commitment creates "chaotic" layouts. Even though preference constraints must not necessarily be met (that is, they are not *hard* constraints in the same way that physical or safety constraints are hard constraints), they help restrict underconstrained layout problems. As a result, people are tempted to add in constraints to the problem formulation during solution generation while solutions are being generated so that the resulting layouts will be *fewer* and *better organized.*

6.3.2.2 Postponed Commitment

The insights I gained studying least commitment, gave me the second idea: to craft the Computational Strategy so that it would follow a *postponed-commitment strategy.* A postponed-commitment strategy postpones commitments up until it is opportune to make them. That is, it does not make decisions as early as an early-commitment strategy would, and not as late as a least-commitment strategy would; it strikes a balance between both 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 reduce the total area of an essential area, and in that they partition the layout space into 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 would be able to narrow the set of satisfying layouts, but it would still need to resort to generating coherent instances to obtain a solution. These coherent instances are then shown to SightPlan's user, who can select the preferred one.

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

6.3.3 Implementation of the Computational Strategy on IPP

Figure 6.41 shows the Computational Strategy applied only to the CM's task on IPP, and Table 6.4 charts the important steps during execution.

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 6.4: Some Cycles from SightPlan's Computational Strategy Applied to the CM's Layout Task on IPP

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). Figure 6.42 shows this intermediate layout.

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),

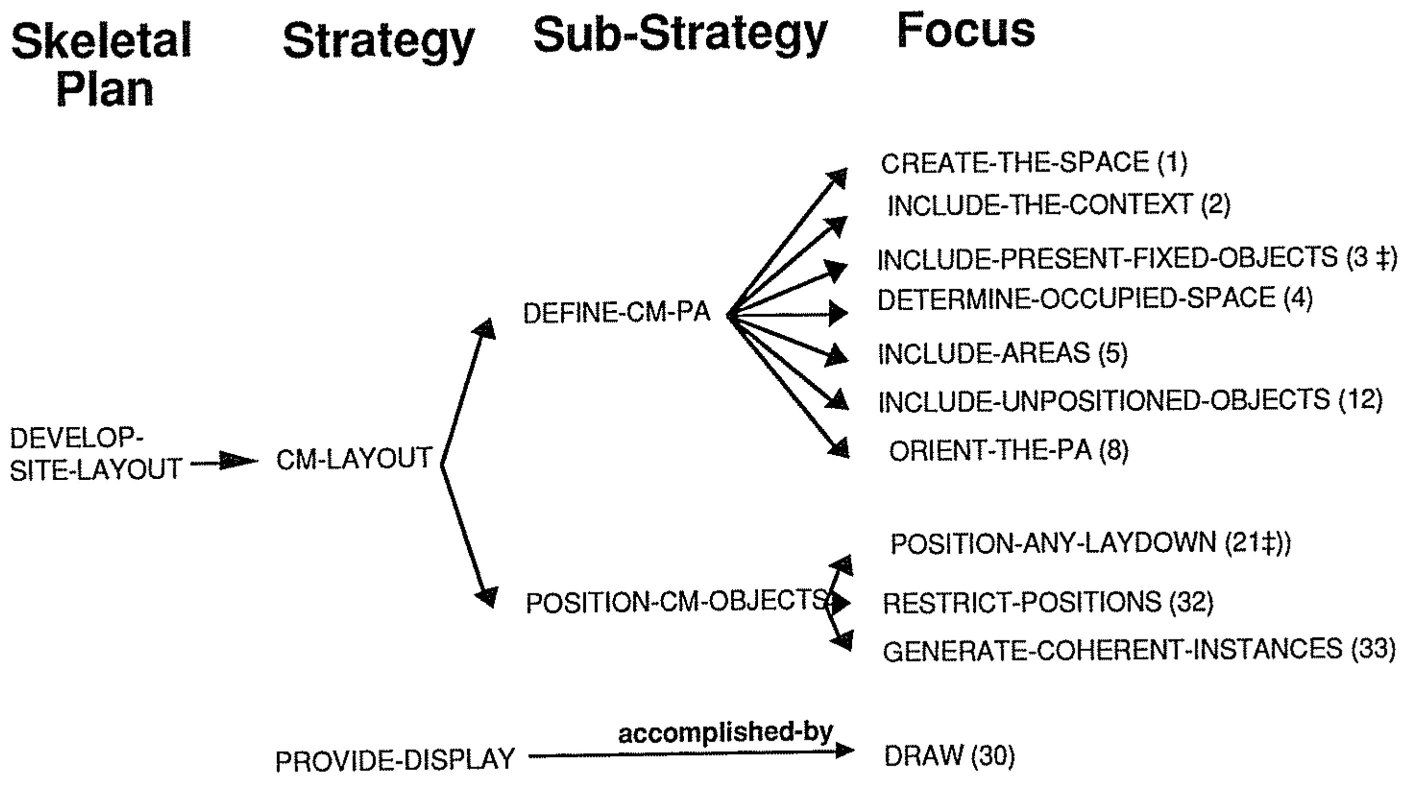


Figure 6.41: Skeletal Plan of the Computational Strategy Applied to IPP
The numbers in parentheses label the foci so that this skeletal plan can easily be compared to those of Figures 6.1, 6.2, 6.25, and 6.31. A focus with number labeled "‡" is similar to the one with corresponding number in the other figures, except for some small changes.

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.

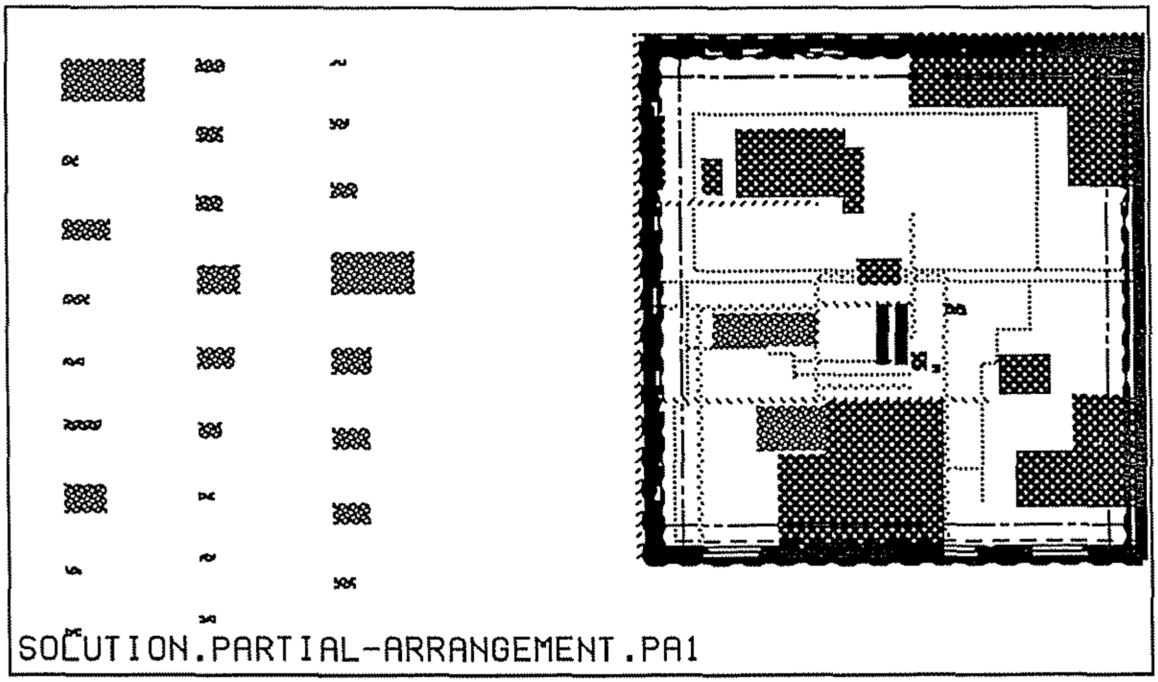
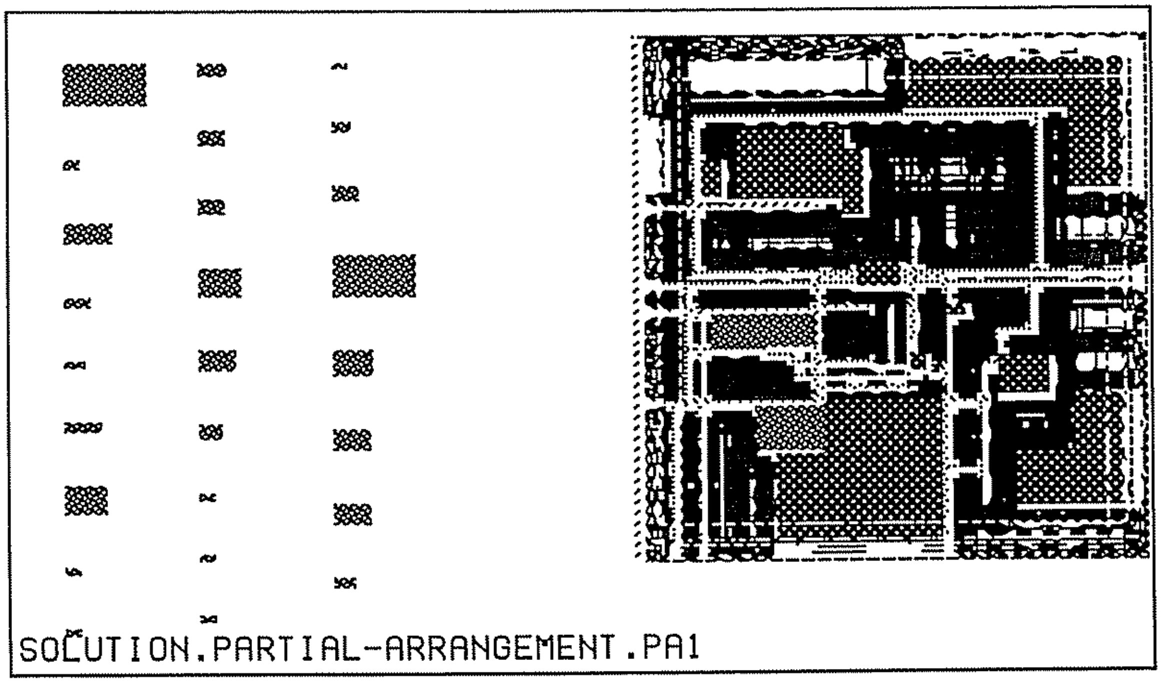


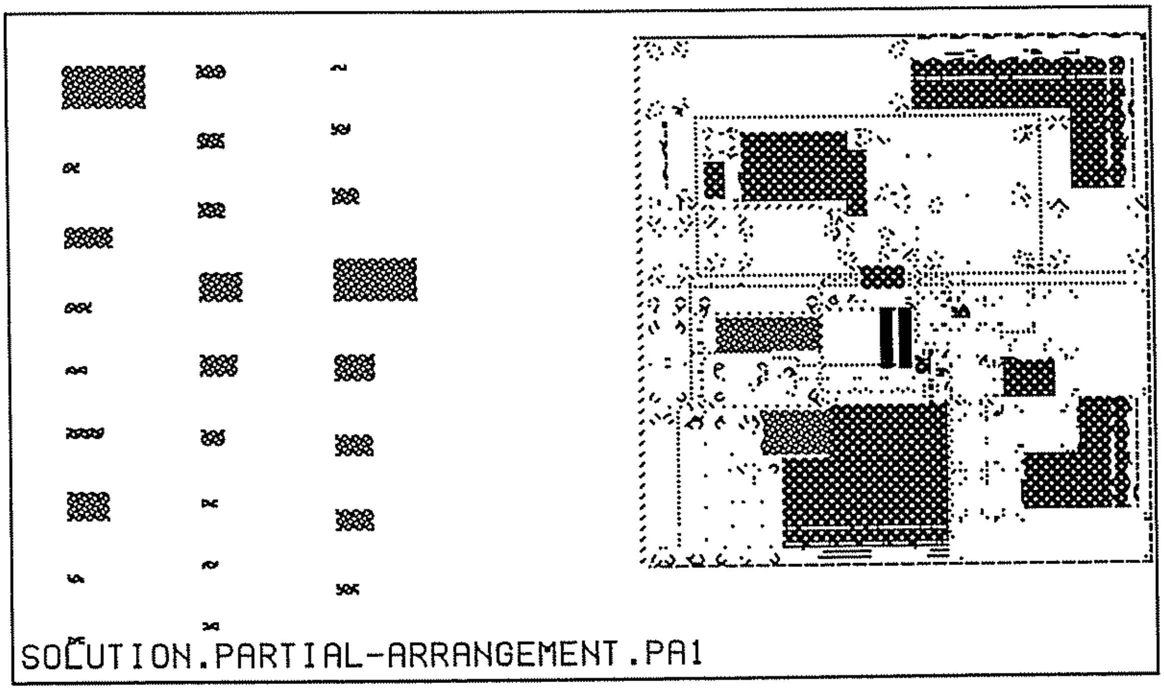
Figure 6.42: SightPlan Included all Permanent and Temporary Objects on IPP

When all constraints on all objects are met, and objects remain with sets of possible locations (Figure 6.43), SightPlan introduces additional constraints into the problem to heuristically reduce those sets. 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, Figure 6.44).



SOLUTION.PARTIAL-ARRANGEMENT.PA1

Figure 6.43: All Objects met their Non-Overlap Constraint with the Permanent Facilities

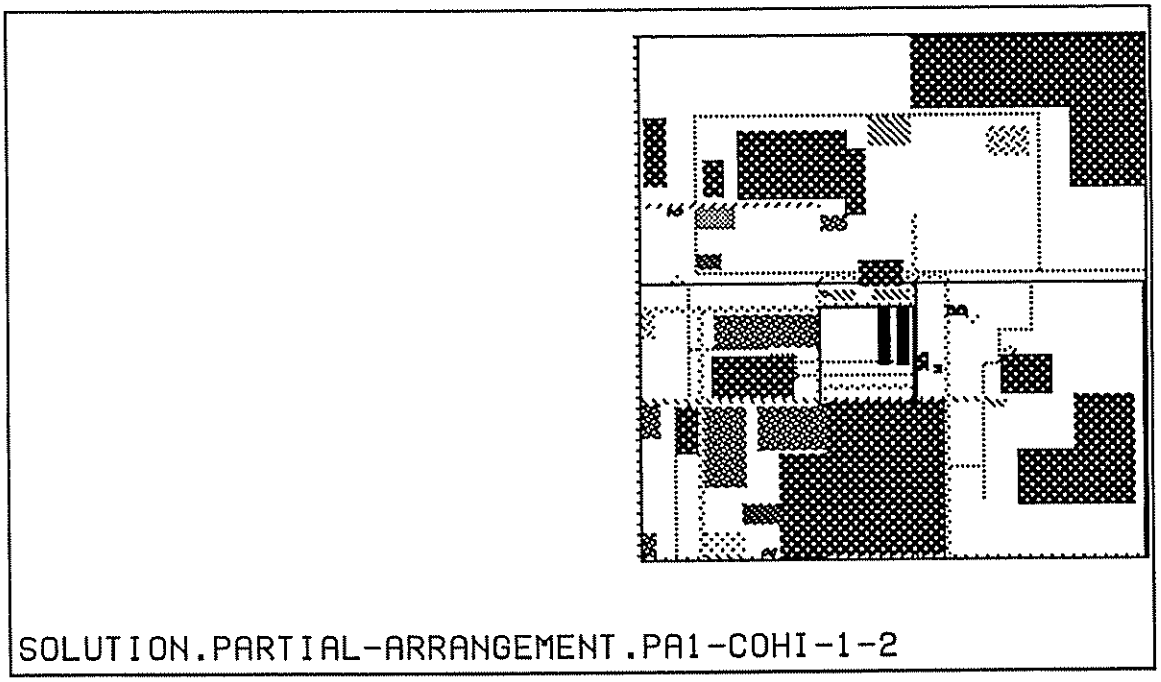


SOLUTION.PARTIAL-ARRANGEMENT.PA1

Figure 6.44: All Objects' Possible Locations Are Restricted to Sets of Points

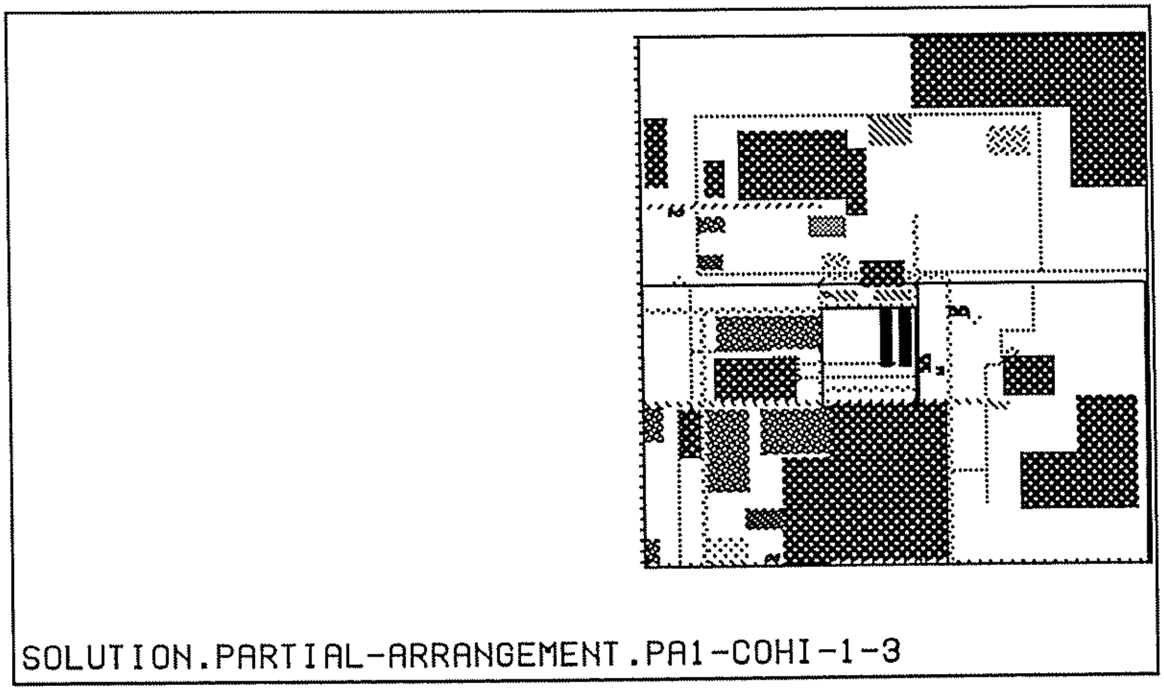
Subsequently, SightPlan uses these limited locations for generating coherent instances (Cycles 183-234). SightPlan returns all coherent instances to the user;

it considers them to be alternative solution arrangements (Cycle 239, Figures 6.45, 6.46, and 6.47).



SOLUTION.PARTIAL-ARRANGEMENT.PA1-COHI-1-2

Figure 6.45: One Coherent Instance of a Solution Layout
Generated by the Computational Model Applied to IPP



SOLUTION.PARTIAL-ARRANGEMENT.PA1-COHI-1-3

Figure 6.46: A Second Coherent Instance of a Solution Layout
Generated by the Computational Model Applied to IPP

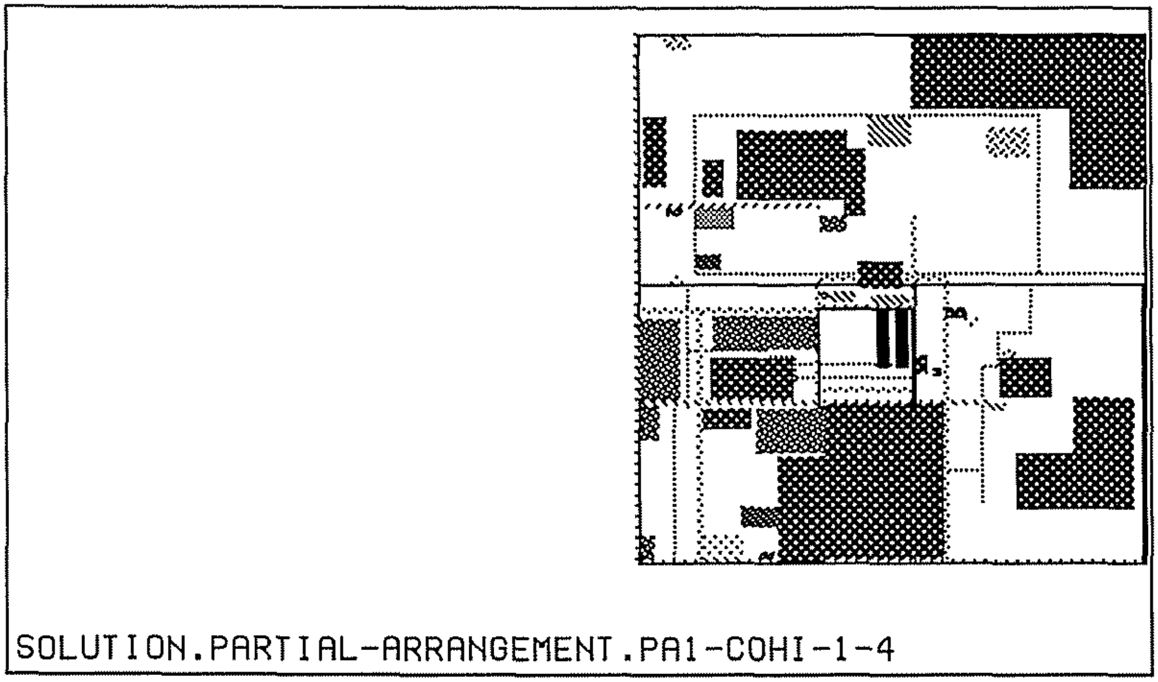


Figure 6.47: A Third Coherent Instance of a Solution Layout Generated by the Computational Model Applied to IPP

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. Although these values provide a comparative measure, the reader will agree that this evaluation value does not assess the value of each layout for construction. I only programmed this feature in SightPlan to illustrate the concept. Significantly better measures for site layout evaluation do not exist to my knowledge but should be developed.

6.3.4 Comparison between the Expert Strategy on IPP and the Computational Strategy on IPP

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 6.3.2), and one of postponed commitment (implemented as the Computational Strategy and described in Sections 6.3.2 and 6.3.3). The question is: Which of these three strategies ought SightPlan to pursue to be most useful as a tool? The following paragraphs discuss the advantages and disadvantages of each. Computer tools can help people overcome some of their human cognitive limitations, so people need not resort to early commitment anymore. Conversely, it is conceivable that computers follow a least-commitment strategy, but many of the resulting solutions may be uninteresting to people. My conclusion is that an intelligently postponed-commitment strategy is the most desirable one for SightPlan.

6.3.4.1 Early Commitment

The Expert Model demonstrated that an early-commitment strategy can succeed for laying out construction sites. Yet, as we have seen, such a strategy may not always result in a solution, even if one exists. The success of the strategy relates mainly to the fact that the IPP problem was defined as being highly underconstrained. Insofar as all constraints defined in the problem are met in the solution layout, SightPlan performed satisfactorily. In that respect, no other strategy could have performed better. However, other strategies might have found solutions faster. For example, the Expert Strategy could be modified to assign other priorities to constraints, while essentially remaining a least-commitment strategy. Such a change in strategy becomes feasible when sufficient knowledge is available ahead of time or at run-time. To a degree, such knowledge was available to characterize the constraint engine. Alternative solutions are excluded if this strategy is followed, yet, it is by providing alternatives to a user or evaluation program that *good* from *better* solutions can differentiated. Thus, SightPlan had to learn strategies that would allow it to generate alternative solutions.

6.3.4.2 Least Commitment

The least-commitment strategy—SightPlan's preliminary Computational Strategy—models a brute-force approach. Brute force has the advantage that it guarantees to 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, may choose to 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 for differentiating between the results may be prohibitive as well. So, it would be worthwhile to balance the costs incurred for generating alternatives, and the benefits gained by finding an arrangement assessed at a higher value.

In rationalizing the Computational Strategy, I suggested that problems seldom are stated in highly underconstrained terms, but instead people opportunistically add constraints before or during problem solving in order to tighten the problem specification and to narrow the set of potential solutions. To prove this concept, I crafted a postponed-commitment strategy for SightPlan, which generates a reasonable subset of all possible solution layouts.

6.3.4.3 Postponed Commitment

The postponed-commitment strategy—SightPlan's 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 return 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.

I chose not to apply this new strategy to the overall layout problem that encompasses the tasks of the AE and the CM simultaneously. Although technically possible, this experiment would not have revealed any more interesting observations than those that I made so far, and it would have been computationally very expensive. For example, because the computional strategy does not know about abstraction or specialization of small aggregations of objects, and lays out all objects in one large arrangement, SightPlan would have needed subtle sampling heuristics to prevent it from bogging down in computation. The potential advantages of incorporating both tasks were discussed in Section 6.1.3. Refinements of the Computational Strategy presented here would need to be developed for elegant incorporation of these two tasks.

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 Summary of the Experimental Approach

The three SightPlan models described in this chapter served as a basis for an experimental study on construction site layout. This study went through the following stages:

1 MODEL FIELD MANAGERS' PRACTICE

Building the models resulting in the Expert Strategy applied to IPP allowed me to formalize, represent, and model field managers' practice for laying out construction sites.

2 CRITIQUE MODEL

The implementations (in particular those of the Expert Strategy applied to IPP and American 1) provided a model of field practice to inspect. I learned about the scope of site layout, the variables that are explicit and implicit in the reasoning behind layout design, and about managers' early-commitment strategies. Also, the implementations demonstrated that the blackboard architecture is appropriate for modeling human design strategies, and that the constructive assembly method is generally applicable to site layout design.

3 IMPROVE MODEL

The models articulated features by means of which I could critique field managers' practice. I proposed alternative solution strategies, the Temporal and the Computational Strategy, and the models allowed me to conduct experiments to demonstrate how layout strategies can be improved upon.

4 *IMPLICATIONS OF IMPROVED MODEL ON MANAGERS' PRACTICE*

I can now project how models like SightPlan might affect current field practice. SightPlan's strength is that it provides cognitive support to its users by making available memory capacity, computation power, and display and representation capabilities. Memory capacity makes it conceivable to build large knowledge bases, which encompass information about entire projects and cover projects' lifetimes. Computation power permits postponed-commitment strategies. Display and representation capabilities allow the system to communicate with other systems so that all data and strategic decisions are always readily available. 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 step for effective collaboration. I envision that a fleshed-out version of a system like SightPlan will take over the role of the physical layout model that is presently the focus of attention for all parties involved in the design, construction, and operation of a facility (see Figure 2.4).

Chapter 7

Conclusions

7.1 Contributions to Knowledge

The SightPlan project has successfully met the research objectives presented in Section 1.2. In doing so, it contributes to knowledge in the following ways:

1 COLLECTED AND FORMALIZED DATA ON CONSTRUCTION FIELD PRACTICE FOR SITE LAYOUT

Chapter 2 contrasted field practice with mathematical models and pointed out the large discrepancy between them. I argued that mathematical models have little appeal to field practitioners, but that one might be able to build AI models to which field managers would closely relate to by incorporating knowledge about field construction practice. I elaborated on what can be learned from industry guidelines and field practice, and demonstrated that at least part of this kind of knowledge can be articulated and communicated, categorized in layers by degree of generality, and integrated into a single system. I pointed out that a system for modeling field practice would necessarily include a layer of project-specific information. In order to acquire such project-specific information, I focused on two case studies; these were described in Chapter 6. This formal representation of construction field practice was then ready for use in a descriptive model of the layout process.

2 MODELED EXPERT'S LAYOUT PROCESS

SightPlan models the layout process by mimicking the actions a person would take in designing a site layout. Its solution method is based on constructively assembling arrangements. The computer program's knowledge is embedded in layers in the BB1 blackboard architecture, and consists of 1) the high-level strategy a person pursues for deciding what action to take next, 2) a set of possible actions that a person can take to lay out arrangements, and 3) general concepts that define the problem type and project-specific examples. This program was then used for experimentation, the aim being to acquire a better understanding of the human layout process and to learn how AI might improve modeling and design practice. Chapter 4 described the architecture SightPlan builds on. Chapter 6 reported on the experiments and the conclusions drawn from them.

I conducted three experiments on alternate input specifications and solution strategies. The first two were needed to meet the objective of modeling the expert's layout process and to validate the model. Instead of further validating SightPlan with additional case studies, I set up another experiment to demonstrate how the model might augment human performance. This last model provided insights that may guide future research.

The first SightPlan model mimicked the "Expert Strategy" of a field expert. It delivered the proofs of concept that it is possible to represent an Expert Strategy for site layout, that a model can apply such a strategy to decide upon its actions, and that the resulting problem-solving process can closely represent how field experts solve the problem.

The second SightPlan model applied the first model's strategy to another site. It allowed us to assess the generality of the Expert Strategy and the model itself. I was not surprised to find that, although the model proved to be well-suited to represent and solve another site's layout problem, the Expert Strategy fell short on dealing with the layout of the second site; some adjustments needed to be made. My conclusion is that the two sites may have been too dissimilar in size for the strategy to apply to both. In particular, whereas the strategy for the first project had to detail separate partial arrangements, the strategy for the much smaller second project needed no such detail.

The third SightPlan model followed the Computational Strategy crafted to better use the power available in a computer, and stemmed from critiqueing the Expert Strategy. It provided us with insight into the relationship between the strategy followed by a person and the cognitive capabilities of that person. Because a computer's limitations are different from those of a person, a computer strategy differs from a human strategy. The Expert Strategy is better than the Computational Strategy in many ways. However, the Computational Strategy proved that, thanks to memory capacity, great computational power, and meticulous thoroughness of a computer, it could propose an alternative strategy to augment human decision-making capabilities.

3 TESTED THE BB1 BLACKBOARD ARCHITECTURE

The BB1 blackboard architecture proved well-suited for the implementation of SightPlan. The two main reasons for this are that the architecture is capable of succinctly representing the available domain knowledge and that the representation used in the model is easy to understand by novice users.

BB1 permitted me to mold the knowledge of site layout in a comprehensive layered representation that distinguishes between generic and domain concepts as well as between strategic and domain actions. In particular, I found the representation for strategic action knowledge extremely useful for portraying the strategies learned from field practitioners.

Because SightPlan models a human design problem, it was important for validation of the system that I could show it to field practitioners. I showed SightPlan using the Expert Strategy to solve the IPP layout to the construction manager of that project, whom I consider a novice to AI modeling techniques. I had no problem conveying to him what the system was doing and why at each problem-solving step. Aided by the graphical display of the layout, he was able to provide me with feedback on the performance of the model. Obviously, field practitioners who know site layout methods as well as AI modeling techniques could critique SightPlan even in more detail.

The speed with which BB1 solved the real-size SightPlan problems—typically in a few hours—is somewhat slow for practical interactive use. In fact, for debugging purposes, I often resorted to developing the models on small, toy-size problems (4 or 5 objects to be positioned) before I would extend them to full-scale. Then again, thanks to the declarative nature of the knowledge representation in BB1, it was really to extend or reduce the SightPlan models in order to include more or fewer objects.

Note that I did not analyze the performance of BB1 in terms of its absolute speed for generating a solution layout. While speed is a major issue for field-operational systems, it is of less importance as a performance criterion in research. Where I have mentioned speed of execution (in Section 6.3.2) I have used it only as a relative measure for comparison. A newer version of BB1 (version 3-0 in CLOS) has been designed and implemented recently with the intent, among other things, of increasing the speed of the system.

As I mentioned in my objectives and scope, I did not set out to compare the utility of BB1 to the utilities of other AI knowledge representation schemes. I do not claim BB1 to be the only environment in which SightPlan could have succeeded. But BB1 provided a flexible declarative control environment, a feature I took advantage of for the implementation of SightPlan's strategies. Another feature of BB1, that it provides an environment in which multiple cooperating experts can collaborate, has not been fully exploited by SightPlan. However, I think that the advantage of choosing the BB1 architecture over another may become apparent when SightPlan is extended and integrated with other systems to encompass more construction management tasks.

4 ASSESSED INTERACTION BETWEEN HUMAN AND SIGHTPLAN MODEL

By building an Expert Strategy model for SightPlan and by researching how a Computational Strategy might improve on that model I learned that most benefit could be gained by building a model in which user and machine collaborate.

Three summarizing findings on SightPlan's strategies are that: First, the Expert Strategy illustrated that people's limited memory makes them resort to an early-commitment strategy. Second, the Computational Strategy is capable of pursuing a least-commitment strategy by generating many alternative layouts, but lacks knowledge about which one to pick from the alternatives. Third, both computer models lack some crucial human strengths in layout problem-solving, such as assessing when an arrangement looks like a good one to be developed further, or opportunistically adding constraints.

One major finding on SightPlan's knowledge representation, which became clear early on in the development of the implementation, is that graphical input and output would constitute an essential part of a practically useful version of the system. First, although SightPlan could solve a layout problem by reasoning only, graphical *output* was needed for its users. Knowledge engineers and novice users alike are greatly helped by a graphical output because it makes it easy for them to perceive how sets of possible locations of objects in an intermediate layout get reduced, and what the final layout looks like. Second, even though users could interact with the system by suggesting actions to override SightPlan's recommendation, or could use the keyboard to input sets of possible locations, we opted for a graphical *input* to SightPlan. The interactive graphics of SightView were a first step in that direction in that they give a user the option to mouse on objects in order to get more information about them, and they allow a user to shrink sets of possible locations graphically before these shrunk sets are sent back to SightPlan.

Tony Confrey and I built a preliminary system that allows some of the interaction between user and machine that we thought would be desirable. SightPlan follows the Computational Strategy with *postponed* commitment. SightView displays partial layout solutions and allows for user interaction in reducing the range of objects' legal positions. SightPlan and SightView pass information back and forth. This setup suggested a number of possibilities for research on knowledge-based interactive graphics that we feel need to be explored further.

Although our objectives have been met, and the SightPlan work described in this dissertation has achieved its goals, I feel that SightPlan can be further improved. In performing this research I ran into many issues and bottlenecks where decisions had to be made. Often, I opted for a "good first guess" in order to make my SightPlan models work, but it is clear that many of the issues that the research raised ought to be addressed more carefully. Section 7.2 describes the successes and shortcomings of SightPlan in terms of the choices that were made for the implementation of the system. Section 7.3 enumerates the implications that can be drawn from the results of SightPlan. The last section of this dissertation concludes with suggestions to extend SightPlan and with directions for future research.

7.2 Successes and Shortcomings of SightPlan

1 SOLUTION METHOD

SightPlan uses the constructive assembly method for generating layouts. SightPlan's strategic advantage over generic constructive methods in that it uses domain-specific knowledge to guide the construction of a solution layout. But consequently, the SightPlan strategy lacks generality that could improve its success rate. One example is that, when a deadlock is encountered by the program pursuing an early-commitment strategy, SightPlan does not have a mechanism that would allow it to identify an earlier layout from which it could try to generate alternatives. In other words, one cannot "backtrack" in execution cycles of BB1. Cycle numbers keep increasing in BB1 as the system performs new actions, although actions can undo the results of previous actions. Another example is that SightPlan currently does not have a method to deal with an overconstrained layout problem. If a situation is encountered in which a constraint cannot be met, SightPlan returns an empty set of positions for the object in question.

SightPlan needs more flexibility in reasoning about shapes and sizes of objects. I have shown with operations on aggregate objects that the model has the capability of doing this kind of reasoning, but I did not have the chance to acquire the necessary knowledge during my interviewing sessions with field experts to make SightPlan size and shape the objects it lays out. An example of how SightPlan might reason about shape is to estimate an object's area first, find a position in the layout where this area might fit, then determine the shape of the fitting area. An example of how SightPlan might reason about size is to estimate an object's area first, then find a position for an object *about* that size, or find a position for two or three pieces of which the total size is the estimated size. To my knowledge, none of the layout systems developed to date deal with this kind of reasoning in a general way.

More knowledge about opportunistically defining and altering constraints at run-time would also be a desirable feature of SightPlan. In the Computational Model, I demonstrated that SightPlan can dynamically introduce run-time constraints for restricting object positions. This was only a simple demonstration of what might constitute a separate research subject.

2 ARCHITECTURE

In the previous section I discussed why I found the BB1 architecture to be appropriate for implementing SightPlan. My main reservation about this architecture lies in the weighting scheme that is used to rank actions by their desirability during problem solving. This scheme is similar to the ranking schemes that are used in numerical methods, although BB1 implements it in a declarative way that can be adjusted at any time. Any type of qualitative reasoning that would lead to a ranking of actions would be acceptable to BB1 as well. The heuristic weights that I implemented in SightPlan are functional, but I feel uncomfortable about the way in which I had to adjust them at implementation time in order to make the system behave in the way I wanted it to. For example, explicit reasoning about those weights at run-time and dynamically changing them may be an option to further increase the system's flexibility. This would constitute reasoning about control, as opposed to reasoning about action, and would introduce a new level of complexity into the problem-solving method, a challenging task!

3 REPRESENTATION

The representation that SightPlan uses, which is based on essential areas for objects, is appropriate for keeping track of all allowable positions for objects at any time. It might be worthwhile—and it would be relatively easy—to extend this essential area notation to represent non-rectangular objects (for example, by approximating a general-shaped object by a set of component rectangles), or to maintain alternative shapes of an object in multiple positions. The advantages of these types of extensions, however, may not justify the burden they would impose on the computational engine. The costs and benefits of such extensions should therefore be investigated first. A similar argument applies to extending SightPlan's representation to make use of a full three-dimensional model.

SightPlan currently controls the execution of a constraint based upon the constraint's type. Other attributes could be attached to constraints in order for them to be classified differently. For example, one could allow a user to define a measure of hardness to a constraint, that is, a degree to which the user would want that constraint to be given priority over other constraints.

4 MODELED VARIABLES

SightPlan reasons only about the spatial layout of facilities and makes use of simple spatial constraints to generate a layout. In Section 6.2.6 I pointed out that some of the spatial constraints may implicitly represent other parameters that are important to the layout model, but that cannot be represented in the model as it stands now. For example, the time frame according to which temporary construction objects are represented is closely tied to the level of detail of the description of such facilities: facilities that are on site for a long time get priority in being located in prime space on site; facilities need to be adjacent to roads or railroads in order to facilitate material flow. If further knowledge acquisition succeeded in finding these hidden variables, it would be worthwhile for the user if the model would make them explicit as well. In that way, SightPlan might be able to make use of general descriptions of "objects" that are also reasoned about by other construction management programs, such as programs for planning and scheduling. If several programs could indeed make use of a common representation of their "objects" (each program could possibly have standard ways of abstracting the information it needs), then that would be a first step towards effective communication between these programs.

7.3 Implications of SightPlan Work

1 CONSTRUCTION MANAGEMENT AND FACILITY ENGINEERING

By articulating the SightPlan Expert Strategy, I have identified one of the points of divisions of tasks between architect/engineers and construction managers, and I was able to suggest a way for integrating their tasks through SightPlan. Such integration into one computer model is possible today because we have computational tools whose cost of operation is low enough to permit extensive use.

Because such computation is cheap, but not free, we now have to learn to make best use of it. For example, I have demonstrated that, although SightPlan can pursue a least-commitment strategy, this may not be its best strategy. I have also argued that, although SightPlan is capable of pursuing an early-commitment strategy, a strategy people have to resort to due to their cognitive limitations, this is probably not the strategy SightPlan should follow. Instead, an intelligent postponed-commitment strategy may be preferred.

This raises the question: How should commitment be postponed? This question cannot necessarily be answered by observing what people do and learning from their strategies. This is because, up until now, people did not have to face this issue, so they did not have the chance to develop appropriate strategies to postpone commitment. Besides, even if people had developed such strategies, these strategies might still not be the best for SightPlan. So, new ways of thinking are required. Thanks to computer tools that augment human cognitive performance, people can now take a more global viewpoint, can broaden their perspective, and should be able to fill the demands for new methods that solve the more general problems.

2 LAYOUT MODELING AND SPATIAL REASONING IN DESIGN

The SightPlan models have illustrated how artificial intelligence techniques can be applied to solve a real-world problem of layout by providing a flexible environment for reasoning about constructive assembly of arrangements. They demonstrated how one can apply domain knowledge to guide the generation of partial layouts, how one can detail arrangements, and how one can incorporate arrangements into each other. Furthermore, I demonstrated that, if a SightPlan model used a strategy different from an early-commitment strategy, the need arose for evaluation functions of partial and global arrangements.

From inspecting SightPlan, we gained some insights into the process of human design and problem-solving. The SightPlan models allowed us to conduct some comparative experiments on the relationship between human layout strategy and cognitive capabilities. The spatial reasoning that SightPlan applies for designing its layout is restricted to satisfying spatial constraints between rectangular objects in two-dimensional continuous space. Although familiar difficulties of spatial reasoning and representation were encountered in the knowledge acquisition and implementation phases, no breakthroughs were made in this field.

3 ARTIFICIAL INTELLIGENCE AND EXPERT SYSTEMS RESEARCH

Although the SightPlan models have not called for innovative changes in the BB1 architecture, they turned out to be a premier illustration of how a real-size design problem could be implemented in that architecture using the ACCORD language and the constructive assembly method.

7.4 Directions for Future Research

SightPlan's development can be steered in many directions that are worthwhile for future research efforts. First, one could work on *overcoming* some of the *limitations* of SightPlan. I pointed out some limitations in Section 7.2, but the critical reader is likely to find many more in the model. Second, one could work on *extending* SightPlan for other tasks, or one could try to *integrate* SightPlan with other models. This second type of work will probably pose the more exciting research challenges. The following are research directions I would like to suggest, classified according to SightPlan's implementation:

1 SIGHTPLAN'S DOMAIN KNOWLEDGE

SightPlan's knowledge about objects to be laid out can be fleshed out. For example, the layout knowledge on other types of projects can be formalized; and information about the layout of permanent facilities can be collected.

SightPlan's representation can be augmented to make explicit additional variables, such as the third spatial dimension (which would allow the system to reason about underground utilities, equipment layout, lifting of materials, and so on), and time (which would allow the system to reason about change of the layout over time considering material flow, and so on).

The ACCORD language for constructive assembly could be extended to provide a vocabulary to express sizing and shaping of objects.

2 SIGHTPLAN'S STRATEGIC KNOWLEDGE

The introduction of additional variables, which was suggested as an extension of SightPlan's domain knowledge, will probably necessitate alterations to the control strategy of SightPlan. Futhermore, SightPlan could be extended to reason about constraint relaxation, to select alternative objects, or to break objects into parts.

One should validate the SightPlan expert strategy thoroughly and fine-tune the heuristic weights of the model by studying many similar power plants, in order to validate that the model represents good expert practice. While this may be worth doing from the standpoint of modeling human cognition, I already pointed out in Section 7.3 that modeling expert practice is probably not desirable if one seeks a computer-based solution.

Human problem-solving approaches can, however, provide a basis for understanding the design process and for identifying design methods that people apply so flexibly. For example, people can cope with overconstrained situations such as those encountered on sites for downtown high-rise construction. People can easily deal with objects of variable shape and objects that may be broken up into many parts. Much more needs to be learned in this field before we can build models that might fully mimic human design activity. For now, I suggest that one develop a toolbox of design methods that people could choose from.

Some on-going work is addressing automatic learning of control strategies, which turns out to be a challenging task for further research. For example, Gans uses NEWWATCH to learn control strategies by observing a SightPlan user who manually selects domain actions [Gans 89]. Confrey uses METAWATCH to learn control strategies from observing multiple runs of a system [Confrey 89].

3 INTEGRATION OF SIGHTPLAN WITH OTHER SYSTEMS

SightPlan can be extended to communicate with other systems. One example of such an extension that we have started investigating is that SightPlan can rely on SightView for its input-output via interactive graphics. SightPlan could also make use of a general CAD project data-base to retrieve the information it needs to lay out a site. These extensions rely on the addition of mainly domain knowledge sources to the SightPlan model.

SightPlan can be extended to incorporate, or to be incorporated in, other systems. An obvious example is to extend SightPlan to reason about layouts for permanent facilities. During the interviews with the AE on IPP, we already collected informally some of the knowledge needed for that. A second example is that, because SightPlan allocates space, and because space is only one of the resources to be allocated on a construction project, SightPlan could be one component in a more general resource allocation program, such as one for construction scheduling. Yet another example is that SightPlan could be integrated with a simulation model that would capture the dynamism of a job site where people, equipment, and materials move around. These extensions may substantially alter both the control and the domain knowledge sources in SightPlan, but in this way SightPlan may one day become part of a system that models the entire construction process or possibly even the entire life-cycle of a project.

Bibliography / References

Note: **all reports labeled KSL-XX-XX are from the Knowledge Systems Laboratory, Department of Computer Science, Stanford University, Stanford, CA**

[AGC 73] Associated General Contractors: *Associated Specialty Contractors Temporary Job Utilities and Services*, National Joint Guideline, NJG-6, pp. 2, 1973

[Allen 84] Allen, J.F.: Towards a General Theory of Action and Time, *Artificial Intelligence*, Vol. 23, pp. 123-154, 1984

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

[Ambler 75] Ambler, A.P., Popplestone, R.J.: Inferring the Positions of Bodies from Specified Spatial Relations, *Artificial Intelligence*, Vol. 6, No. 2, pp. 157- 174, 1975

[Anderson 78] Anderson, J.R.: Arguments Concerning Representations for Mental Imagery, *Psychological Review*, Vol. 85, pp. 249-277, 1978

[Anderson 79] Anderson, J.R.: Further Arguments Concerning Representations for Mental Imagery: A Response to Hayes-Roth and Pylyshyn, *Psychological Review*, Vol. 86, No. 4, pp. 395-406, 1979

[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 Accomodation, *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

[Bachant 84] Bachant, J., McDermott, J.: R1 Revisited: Four Years in the Trenches, *The AI Magazine*, pp. 21-32, Fall 1984

[Baybars 82] Baybars, I.: The Generation of Floor Plans with Circulation Spaces, *Environment and Planning B*, Vol. 9, pp. 445-456, 1982

[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

[Bennet 78] Bennet, J.S., Creary, L., Englemore, R.S., Melosh R.: *SACON: A Knowledge-Based Consultant for Structural Analysis*, Stanford Heuristic Programming Project Memo HPP-78-23 Computer Science Department, Report No. STAN-CS-78-699, Stanford University, Stanford, CA, September 1978

[Bennet 79] Bennet, J.S., Englemore, R.S.: SACON: A Knowledge-Based Consultant for Structural Analysis, *Proceedings of the Sixth International Joint Conference on Artificial Intelligence*, pp. 47-49, August 1979

[Berliner 81a] Berliner, H. J.: *Search vs. Knowledge: An Analysis from the Domain of Games*, Carnegie-Mellon University, Department of Computer Science, CMU-CS-82-104, 1981

[Berliner 81b] Berliner, H.: An Examination of Brute Force Intelligence, *Proceedings of IJCAI 81*, Vol.1, pp. 581-587, 1981

[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)

[Blair 85] Blair, E.L., Miller, S.: An Interactive Approach to Facilities Design Using Microcomputers, *Comp. and Industrial Engineering*, Vol. 9, No. 1, pp. 91-102, 1985

[Bloch 79] Bloch, C.J.: Catalogue of Small Rectangular Plans, *Environment and Planning B*, Vol. 6, pp. 155-190, 1979

[Boltz 87] Boltz, D.H., Molinski, J.: Computer Scheduling: On Line and On Time, *Civil Engineering*, pp. 56-58, June 1987

[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

[BrownA 88] Brown, A.D.: Automated Placement and Routing, *CAD*, Vol. 20, No. 1, Jan/Feb 1988, pp. 39-44

[BrownH 83] Brown, H., Tong, C., Foyster, G.: Palladio: An Exploratory Environment for Circuit Design, *IEEE Computer*, pp. 41-56, December 1983

[BRT 83] Business Round Table: *More construction for the Money, Construction Industry Cost Effectiveness Project*, Technical Report, Business Roundtable, January 1983

[Buchanan 82] Buchanan, B.G., Duda R.O.: *Principles of Rule-Based Expert Systems*, Stanford University, Computer Science Department, Heuristic Programming Project, Report No. HPP-82-14, Stanford, CA, August 1982

[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*, Report No. KSL-87-03, 41 pages, January 1987, to appear in Synthese

[Budwani 85] Budwani, R.N.: The Data Base for U.S. Power Plants, *Power Engineering*, Vol. 89, pp. S-1 to S-28, January 1985

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

[Chan 86] Chan, W.T.: *Logic Programming for Managing Constraint-Based Engineering Design*, Ph.D. Dissertation, Department of Civil Engineering, Stanford University, 114 pages, Stanford, CA, March 1986

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

[Collins 75] Collins, R.E.: Fire Protection for Construction Projects, *Professional Safety*, pp. 18-21, May 1975

[Confrey 88] Confrey, T., Daube, F.: *GS2D: A 2D Geometry System*, Report No. KSL-88-15, Stanford, CA, 16 pages, March 1988

[Confrey 89] Confrey, T.: *Knowledge-Based Strategy Generalization*, Stanford University, Computer Science Department, Knowledge Systems Laboratory, unpublished, 33 pages, Stanford, CA, 1989

[Daube 88] Daube, F., Hayes-Roth, B.: FIRST: A Case-Based Redesign System in the BB1 Blackboard Architecture, *Proceedings of the AAAI-88 Workshopon Case-Based Reasoning*, also Report No. KSL-88-32, August 1988

[Daube 89] Daube, F., Hayes-Roth, B.: A Case-Based Redesign System in the Mechanical Domain, *Proceedings of the Eleventh International Joint Conference on Artificial Intelligence*, IJCAI 89, Detroit, Michigan, 20-25 August 1989

[Davis 81] Davis, E.: *Organizing Spatial Knowledge*, Yale University, Department of Computer Science, Research Report No. 193, January 1981

[Davis 83] Davis, E.: The MERCATOR Representation of Spatial Knowledge, *Proceedings of the Eighth International Joint Conference on Artificial Intelligence* (IJCAI 83), Morgan Kaufmann, Los Altos, California, pp. 295-301, 1983

[Davis 84] Davis, E.: *Representing and Acquiring Geographic Knowledge*, PhD Dissertation, Yale University, Department of Computer Science, Research Report No. 292, 232 pages, January 1984

[Dixon 84] Dixon, J.R., Simmons, M.K.: An Architecture for the Application of Artificial Intelligence to Design, *Proceedings of the ACM/IEEE 21st Annual Design Automation Conference*, Albuquerque, N.M., pp. 634-640, 1984 (quoted in [Mittal 86])

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

[Dreyfus 86a] Dreyfus, H.L., Dreyfus, S.E.: *Mind over Machine: The Power of Human Intuition and Expertise in the Era of the Computer*, The Free Press, New York, 231 pages, 1986

[Dreyfus 86b] Dreyfus, H.L., Dreyfus, S.E.: Why Computers May Never Think Like People, *Technology Review*, Vol. 89, No. 1, pp. 42-61, January 1986

[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

[Dym 84] Dym, C.L.: *Expert Systems: New Approaches to Computer-Aided Engineering*, Xerox Palo Alto Research Center, ISL-84-2, P84-00005, 35 pages, April 1984

[Dym 88] Dym, C.L., Henchey, R.P., Delis, E.A., Gonick, S.: A Knowledge-Based System for Automated Architectural Code-Checking, *Computer-Aided Design*, March 1988

[Eastman 69] Eastman, C.M.: Cognitive Processes and Ill-Defined Problems: A Case Study from Design, *Proceedings of IJCAI 69*, pp. 669-690, May 1969

[Eastman 70] Eastman, C.M.: Representations for Space Planning, *Communications of the ACM*, Vol. 13, No. 4, pp. 242-250, April 1970

[Eastman 71] Eastman, C.M.: Heuristic Algorithms for Automated Space Planning, *Proceedings of the Second International Joint Conference on Artificial Intelligence*, London, U.K., pp. 27-39, 1971

[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 73] Eastman, C.M.: Automated Space Planning, *Artificial Intelligence*, Vol. 4, pp. 41-64, 1973

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

[Edison 84] Edison Electric Institute: *Power Directory 1984/1985: An Environmental Directory of U.S. Steam Electric Power Plants*, 1984

[Finger 89] Finger, S., Dixon, J.R.: A Review of Research in Mechanical Engineering Design, *Research in Engineering Design*, 2-8222, 32 pages, Vol. 1, No. 1, 32 pages, journal is to appear in 1989

[Finn 86] Finn, G.A., Reinschmidt, K.F.: Expert Systems in an Engineering-Construction Firm, *Expert Systems in Civil Engineering*, Proceedings of a Symposium sponsored by the Technical Council on Computer Practices of the American Society of Civil Engineers, ASCE Convention, Washington, April 8-9, 1986

[Fischer 82] Fischer, G.: Symbiotic, Knowledge-Based Computer Support Systems, pp. 311-318, in [Johannsen 83] Johannsen, G., Rijnsdorp, J.E. (editors): *IFAC Analysis, Design, and Evaluation of Man-Machine Systems*, Baden-Baden, Federal Republic of Germany, Pergamon Press, 424 pages, 27-29 September 1982

[Fisher 84] Fisher, E.L.: *Knowledge-Based Facilities Design*, PhD Dissertation, School of Industrial Engineering, Purdue University, 1984

[Flemming 78] Flemming, U.: Wall Representations of Rectangular Dissections and their Use in Automated Space Allocation, *Environment and Planning B*, Vol. 5, pp. 215-232, 1978

[Flemming 86a] Flemming, U., Coyne, R., Glavin, T.: A Generative Expert System for the Design of Building Layouts, *Proceeding of the First International Conference of Applications of Artificial Intelligence to Engineering Problems*, April 1986

[Flemming 86b] Flemming, U., Rychener, M.D., Coyne, R.F., Glavin, T.J.: *A Generative Expert System for the Design of Building Layouts - version 1*, Progress Report, Center for Art and Technology, Carnegie-Mellon University, 72 pages, June 1986

[Flemming 88] Flemming, U., Coyne, R., Glavin, T., Rychener, M.: A Generative Expert System for the Design of Building Layouts - version 2, Gero, J.S. (editor), *Artificial Intelligence in Engineering: Design*, Elsevier, pp. 445-464, 1988

[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

[Foulds 85] Foulds, L.R., Gibbons, P.B., Giffin, J.W.: Facilities Layout Adjacency Determination: An Experimental Comparison of Three Graph Theoretic Heuristics, *Operations Research*, Vol. 33, No. 5, pp. 1091-1106, September/October 1985

[Fox 87] Fox, M.S.: *Constraint-Directed Search: A Case Study of Job-Shop Scheduling*, Morgan Kaufmann Publishers, California, 184 pages, 1987

[Francis 67] Francis, R.L.: Sufficient Conditions for Some Optimum Property Facility Designs, *Operations Research*, Vol. 15, pp. 448-466, 1967

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

[Fuchs 83] Fuchs, H.: *Selected Reprints on VLSI Technologies and Computer Graphics*, IEEE Computer Society Press, 490 pages, 1983

[Fullman 79] Fullman, J.: *Construction Safety, Security and Loss Prevention*, Wiley & Sons, 1979

[Furusaka 84] Furusaka, S., Gray, C.: A Model for the Selection of the Optimum Crane for Construction Sites, *Construction Management and Economics*, Vol. 2, E. & F. Spon, London, pp. 157-176, 1984

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

[Galle 86] Galle, P.: Abstraction as a Tool for Automated Floor-Plan Design, *Planning and Design*, Vol. 13, pp. 21-46, 1986

[Gans 89] Gans, A.: *NEWWATCH: Learning Interrupted Strategies by Observing Action*, Stanford University, Knowledge Systems Laboratory, unpublished, 57 pages, 1989

[Garey 79] Garey M.R., Johnson, D.S.: *Computers and Intractability, A Guide to the Theory of NP-Completeness*, W.H. Freeman and Company, N.Y., 338 pages, 1979

[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

[Garvey 87b] Garvey, A., Cornelius, C., Hayes-Roth, B.: Computational Costs versus Benefits of Control Reasoning, *Proceedings of AAAI 87*, pp. 110-115, 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

[Gero 86] Gero, J., Balachandran, M.: Knowledge and Decision Processes, Proceedings of the First International Conference on Applications of Artificial Intelligence in Engineering, Southampton, UK, pp. 343-352, April 1986

[Gero 87] Gero, J.S. (editor): *Expert Systems in Computer-Aided Design*, Proceedings of the IFIP WG 5.2 Working Conference on Expert Systems in Computer-Aided Design, North-Holland, 533 pages, February 1987

[Gero 88a] Gero, J.S. (editor): *Artificial Intelligence in Engineering: Design*, 465 pages, *Robotics and Processes*, 403 pages, *Diagnosis and Learning*, 421 pages, Proceedings of the Third International Conference on Applications of Artificial Intelligence in Engineering, Palo Alto, CA, Elsevier, August 1988

[Gero 88b] Gero, J.S., Maher, M.L., Zhang, W.: Chunking Structural Design Knowledge as Prototypes, in Gero, J.S. (editor), *Artificial Intelligence in Engineering: Design*, pp. 3-21, Proceedings of the Third International Conference on Applications of Artificial Intelligence in Engineering, Palo Alto, CA, Elsevier, August 1988

[Gero 89] Gero, J.S. (editor): *WAID: Workshop on Research Directions for Artificial Intelligence in Design*, Working Paper, The Design Computing Unit, Department of Architectural and Design Science, University of Sydney, NSW, Australia, 133 pages, May 1989

[Gilleard 78] Gilleard, J.: LAYOUT – a Hierarchical Computer Model for the Production of Architectural Floor Plans, *Environment and Planning B*, Vol. 5, pp. 233-241, 1978

[Goel 88] Goel, A., Chandrasekaran, B.: *Integrating Model-based Reasoning and Case-based Reasoning for Design Problem Solving*, Laboratory for Artificial Intelligence Research, Department of Computer and Information Science, The Ohio State University, 17 pages, 15 May 1988

[Gray 85] Gray, C., Little, J.: A Systematic Approach to the Selection of an Appropriate Crane for a Construction Site, Department of Construction Management, University of Reading, Whiteknights, Reading, to appear in *Construction Management and Economics*, No. 3, 1985

[Green 86] Green, S.: SPACES: A System for the Representation of Commonsense Knowledge about Space for Design, *6th Annual Technical Conference on Expert Systems, Brighton*, U.K., 15-18 December 1986

[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

[Harris 81] Harris, F.: *Construction Plant Excavating and Materials Handling, Equipment and Methods*, Garland STPM Press, 225 pages, 1981

[Hart 85] Hart, A.: Knowledge Elicitation: Issues and Methods, *CAD*, Vol. 17, no. 9, pp. 455-462, November 1985

[Hayes-RothB 83] Hayes-Roth, B.: *The Blackboard Architecture: A General Framework for Problem Solving?*, Heuristic Programming Project, Report No. HPP-83-30, Stanford University, 28 pages, May 1983

[Hayes-RothB 84a] Hayes-Roth, B.: *BB1: An Architecture for Blackboard Systems that Control, Explain, and Learn about their own behavior*, Heuristic Programming Project, Report HPP-84-16, Stanford University, 1984

[Hayes-RothB 84b] Hayes-Roth, B.: *A Blackboard Model of Control*, Heuristic Programming Project, Report No. HPP 84-38, Stanford University, 84 pages, August 1984

[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 Protean 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

[Hayes-RothB 88b] Hayes-Roth, B., Hewett, M., Johnson, M.V.Jr., Garvey, A.: ACCORD: A Framework for a Class of Design Tasks, Report No. KSL 88-19, 12 pages, March 1988

[Hayes-RothB 88c] Hayes-Roth, B.: Making Intelligent Systems Adaptive, Report No. KSL 88-64, to appear as a chapter in: Van Lehn, K. (editor): *Architectures for Intelligence*, Lawrence Erlbaum, 1988

[Hayes-RothB 88d] Hayes-Roth, B., Hewett, M., Hewett, R., Washington, R., Hastings, R., Parlante, N., and Seiver, A.: *Monitoring Surgical Intensive Care Patients: A Case-study in Adaptive Intelligent Systems*, Technical Report, Stanford University, Stanford, CA, 1988

[Hayes-RothB 89] Hayes-Roth, B., Washington, R., Hewett, R., Hewett, M., Siever, A.: *Intelligent Real-Time Monitoring and Control*, Report No. KSL 89-05, 7 pages, January 1989

[Hayes-RothF 79] Hayes-Roth, F.: Distinguishing Theories of Representation: A Critique of Anderson's "Arguments Concerning Mental Imagery," *Psychological Review*, Vol. 86, No. 4, pp. 376-382, 1979

[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

[Herskovits 82] Herskovits, A.: *Space and the Prepositions in English: Regularities and Irregularities in a Complex Domain*, PhD Dissertation, Department of Linguistics, Stanford University, Stanford, California, June 1982

[Herskovits 85] Herskovits, A.: Semantics and Pragmatics of Locative Expressions, *Cognitive Science*, Vol. 9, pp. 341-378, 1985

[Herskovits 86] Herskovits, A.: *Language and Spatial Cognition*, Cambridge University Press, 208 pages, 1986

[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

[Hewett 88] Hewett, M., Hayes-Roth, B.: *Real-Time I/O in Knowledge-Based Systems*, Report KSL-88-22, 11 pages, 1985

[Hillier 80] Hillier, F.S., Lieberman, G.J.: *Introduction to Operations Research*, Holden-Day, Inc, Oakland, California, 829 pages, 1980

[Hobbs 85] Hobbs, J.R., Blenko, T., Croft, B., Hager, G., Kautz, H.A., Kube, P., Shoham, Y.: *Commonsense Summer: Final Report*, Center for the Study of Language and Information, Stanford University, Stanford, California, Report No. CSLI-85-35, October 1985

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

[Howe 86] Howe, A. E., Cohen, P. R., Dixon, J. R., and Simmons, M. K.: Dominic: A Domain-Independent Program for Mechanical Engineering Design, *The International Journal for Artificial Intelligence in Engineering*, Vol. 1, No. 1, pp. 23-28, July 1986

[Hutchinson 87] Hutchinson, P.J., Rosenman, M.A., Gero, J.S.: RETWALL: An Expert System for the Selection and Preliminary Design of Earth Retaining Structures, *Knowledge-Based Systems*, Vol. 1, No. 1, December 1987

[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

[JohnsonMV 88] Johnson, M.V.Jr., Hayes-Roth, B.: *Learning to Solve Problems by Analogy*, Report No. KSL-88-01, 15 pages, 1988

[JohnsonT 70] Johnson, T., Weinzapfel, G., et al.: *IMAGE: An Interactive Graphics-Based Computer System for Multi-Constrained Spatial Synthesis*, Report, Department of Architecture, M.I.T., Cambridge, Massachusetts, September 1970

[Johnston 81] Johnston, J.E.: *Site Control of Materials: Handling, Storage and Protection* (Table of Contents) Butterworths, London, 137 pages, 1981

[Kautz 85] Kautz, H.A.: Formalizing Spatial Concepts and Spatial Language, Chapter 2 in [Hobbs 85] Hobbs, J.R., et al.: *Commonsense Summer: Final Report*, Center for the Study of Language and Information, Stanford University, Stanford, California, Report No. CSLI-85-35, October 1985

[Kirkpatrick 83] Kirkpatrick, S., Gelatt, C.D., Vecchi, M.P.: Optimization by Simulated Annealing, *Science*, Vol. 220, No. 4598, pp. 671- 680, 1983

[Kosslyn 77] Kosslyn, S.M., Shwartz, S.P.: A Simulation of Visual Imagery, *Cognitive Science*, Vol. 9, pp. 265-295, 1977

[Kosslyn 80] Kosslyn, S.M.: *Image and Mind*, Harvard University Press, 1980

[Kuipers 78] Kuipers, B.J.: Modeling Spatial Knowledge, *Cognitive Science*, Vol. 2, No. 2, pp. 129-153, 1978

[Kuipers 79] Kuipers, B.J.: Commonsense Knowledge of Space: Learning from Experience, *Proceedings of the Sixth International Joint Conference on Artificial Intelligence*, Morgan Kaufmann, Los Altos, California, pp. 499-501, 1979

[Kuipers 82] Kuipers, B.J.: The "Map in the Head" Metaphor, *Environment and Behavior*, Vol. 14, No. 2, pp. 202-220, 1982

[Kumara 88] Kumara, S. R.T., Kashyap, R.L., Moodie, C.L.: Application of Expert Systems and Pattern Recognition Methodologies to Facilities Layout Planning, *International Journal of Production Research*, Vol. 26, No. 5, pp. 905-930, 1988

[Kunz 87] Kunz, J.C.: *Model Based Reasoning*, working paper, Intellicorp, Mountain View, CA, 15 pages, 1987

[Kunz 88] Kunz, J.C., Stelzner, M.J., Williams, M.D.: From Classic expert Systems to Models: Introduction to a Methodology for Building Knowledge-Based Systems,in *Topics in Expert Systems Design* (in press), North Holland, 1988

[Latombe 89] Latombe, J-C.: *Global path Planning Approaches for Material Movements in a Worksite*, Stanford University, Center for Integrated Facility Engineering, CIFE Technical Report No. 003, 56 pages, 1989

[Lawler 63] Lawler, E.L.: The Quadratic Assignment Problem, *Management Science*, 9, 586, 1963

[Lee 67] Lee, R.C., Moore, J.M.: CORELAP—Computerized Relationship Layout Planning, *Journal of Industrial Engineering*, Vol. 18, No. 3, pp. 195-200, March 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

[LevittR 89] Levitt, R.E., Tommelein, I.D., Hayes-Roth, B., Confrey, T.: *Using the BB1 blackboard control architecture in the SightPlan system for the desing of construction site layouts*, Video tape discussion and software demonstration, Stanford University Center for Integrated Facilities Engineering, 1989.

[Liggett 80] Liggett, R.S.: The quadratic assignment problem: an analysis of applications and solution strategies, *Environment and Planning B*, Vol. 7, pp. 141-162, 1980

[Lozano-Perez 84] Lozano-Perez, T.: Spatial Planning: A Configuration Space Approach, IEEE Transactions on Computers, February 1984

[Lynch 60] Lynch, K.: *The Image of the City*, Cambridge, Massachussetts, MIT Press, 1960

[MacGregor 87] MacGregor, J.N.: Short-Term Memory Capacity: Limitation or Optimization?, *Psychological Review*, Vol. 94, No. 1, pp. 107-108, 1987

[Maggio 88] Maggio, R.: *The Nonsexist Word Finder, A Dictionary of Gender-Free Usage*, Beacon Press, Boston, 210 pages, 1988

[Maher 85] Maher, M.L., Fenves S.J.: *HI-RISE: A Knowledge-Based Expert System for the Preliminary Structural Design of High Rise Buildings*, Department of Civil Engineering, Carnegie-Mellon University, Report No. R-85-146, 125 pages, January 1985

[MaherR 81] Maher, R.H., Stark, R.M.: Earthmoving Logistics, ASCE, *Journal of the Construction Division*, Vol. 107, No. CO2, pp. 197, June 1971

[March 71] March, L., Steadman, P.: *The Geometry of Environment*, RIBA Publications, Ltd., 1971

[Markus 88] Markus, S., Stout, J., McDermott, J.: VT: An Expert Elevator Designer That Uses Knowledge-Based Backtracking, *AI Magazine*, Vol. 9, No. 1, pp. 95-112, 1988

[Mawdesley 88] Mawdesley, M.J., Cullingford, G., Haddadi, A.: Site Layout and Resource Scheduling—An Approach to Modelling Movement around Sites, *Proceedings of the 5th International Symposium on Robotics in Construction*, Tokyo, Japan, pp. 391-399, 6-8 June 1988

[Mayer 83] Mayer, S.: *Quantitative Construction Management*, Wiley & Sons, 1983

[McDermottD 80] McDermott, D.: *Spatial Inferences with Ground, Metric Formulas on Simple Objects*, Yale University, Department of Computer Science, Research Report No. 173, January 1980

[McDermottD 81] McDermott, D.: *Finding Objects with Given Spatial Properties*, Yale University, Department of Computer Science, Research Report No. 195, 1981

[McDermottJ 80] McDermott, J.: R1: An Expert in the Computer Systems Domain, *Proceedings AAAI-80*, Stanford, California, pp. 269-271,1980

[McDermottJ 81c] McDermott, J.: R1's Formative Years, *AI Magazine*, Vol. 2, No. 2, pp. 21-29, 1981

[McDermottJ 82] McDermott, J.: R1: A Rule-Based Configurer of Computer Systems, *Artificial Intelligence*, Vol. 19, No. 1, pp. 39-88, 1982

[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

[MillerR 80] Miller, R.K.: *Noise Control for the Construction Industry*, Fairmont Press, 1980

[MillerP 83] Miller, P.L.: ATTENDING: Critiquing a Physician's Management Plan, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-5, pp. 449-461, 1983

[Minsky 75] Minsky, M.: A Framework for Representing Knowledge, in Winston, P.H. (editor): *The Psychology of Computer Vision*, McGraw-Hill, New York, pp. 221-277, 1975

[Minsky 82] Minsky, M.: Why People Think Computers Can't, *AI Magazine*, pp. 3-15, Fall 1982

[Mitchell 85] Mitchell, T.M., Steinberg, L.I., Shulman, J.S.: A Knowledge-Based Approach to Design, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Col. PAMI-7, No. 5, pp. 502-510, September 1985

[Mittal 84] Mittal, S., Dym, C.L.: Knowledge Acquisition from Multiple Experts, *Proceedings of the IEEE Workshop on Principles of Knowledge-Based Systems*, Denver, Colorado, December 1984

[Mittal 86] Mittal, S., Dym, C.L., Morjaria, M.: PRIDE: An Expert System for the Design of Paper Handling Systems, *IEEE Computer*, pp. 102-114, July 1986

[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

[Mostow 85] Mostow, J.: Toward Better Models of the Design Process, *The AI Magazine*, pp. 44-57, Spring 1985

[Murray 88] Murray, W.R.: *Dynamic Instructional Planning in the BB1 Blackboard Architecture*, Technical Report, FMC Corporation Artificial Intelligence Center, 1988

[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

[NSC 77] National Safety Council: Fire Prevention and Control on Construction Sites, *National Safety News*, Data Sheet 491, 1977 Revision C, pp. 83-89, June 1977

[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

[Newell 72] Newell, A., Simon, H.A.: *Human Problem Solving*, Prentice-Hall, Englewood Cliffs, NJ, 1972

[Nii 86a] Nii, H.P.: Blackboard Systems: The blackboard model of problem solving and the evolution of blackboard architectures, *AI Magazine*, Vol. 7, No. 2, pp. 38-53, 1986

[Nii 86b] Nii, H.P.: Blackboard Systems: Blackboard Application Systems, blackboard systems from a knowledge engineering perspective, *AI Magazine*, Vol. 7, No. 3, pp. 82-106, 1986

]Nilsson 80] Nilsson, N.J.: Principles of Artificial Intelligence, Tioga Pub. Co., Palo Alto, CA, 476 pages, 1980

[Nixon 84] Nixon, I.M.: *I.F. An Idiomatic Floorplanner*, University of Edinburgh, Department of Computer Science, CSR170-84, October 1984

[Nof 83] Nof, S.Y.: An Expert System for Planning/Replanning Programmable Facilities, *Seventh International Conference on Production Research*, 1983

[O'Keefe 85] O'Keefe, R.M.: Expert Systems and Operational Research Mutual Benefits, *Journal Opl. Res. Soc.*, Vol. 36, No.2, pp. 125-129, 1985

[OSHA 85] OSHA Safety and Health Standards: *Construction Industry*, Superintendent of Documents, U.S. Govenrment Printing Office, Washington, D.C., 230 pages, 1985

[Pardee 87] Pardee, W., Hayes-Roth, B.: *An Intelligent Materials Processor*, Research Report 1, Rockwell International, Science Center, Palo Alto Laboratory, 38 pages, February 1987

[Parker 72] Parker, H.W., Oglesby, C.H.: *Methods Improvement for Construction Managers*, McGraw-Hill Book Company, 1972

[Parsons 80] Parsons, R.M., Pachuta, J.D.: System for Material Movement to Work Areas, *ASCE Journal of the Construction Division*, Vol. 106, No. CO1, pp. 55-71, March 1980

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

[Pfeffercorn 75b] Pfeffercorn, C.E.: The Design Problem Solver: A System for Designing Equipment or Furniture Layouts, pp. 98-146 in [Eastman 75] Eastman, C.M. (editor): *Spatial Synthesis in Computer-Aided Building Design*, Halsted Press, New York, 1975

[Pick 83] Pick, H., Acredolo, L. (editors): *Spatial Orientation: Theory, Research and Application*, Conference on Spatial Orientation and Perception, 1980, University of Minnesota, 1983

[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

[Popescu 86] Popescu, C.: Steps and Criteria in Designing Temporary Facilities in Romania, *Proceedings CIB-W65*, Washington, D.C., 19-20 May 1986

[Popplestone 80] Popplestone, R.J., Ambler, A.P., Bellos, I.M.: An Interpreter for a Language for Describing Assemblies, *Artificial Intelligence*, Vol. 14, No. 1, pp. 79-107, 1980

[Popplestone 86] Popplestone, R.J.: *The Edinburgh Designer System as a Framework for Robotics*, working paper, Department of Computer and Information Science, University of Massachusetts at Amherst, Amherst, Massachusetts, 20 pages, 1986

[Power Engineering 87] Power Engineering: *New Generating Plants*, Pennwell Publishing Company, 16 pages, April 1987

[Pylyshyn 73] Pylyshyn, Z.W.: What the Mind's Eye Tells the Mind's Brain: A Critique of Mental Imagery, *Psychological Bulletin*, Vol. 80, pp. 1-14, 1973

[Pylyshyn 79] Pylyshyn, Z.W.: Validating Computational Models: A Critique of Anderson's Indeterminacy of Representation Claim, *Psychological Review*, Vol. 86, No. 4, pp. 383-394, 1979

[Rabbat 88] Rabbat, G. (editor): *Handbook of Advanced Semiconductor Technology and Computer Systems*, Van Nostrand Reinhold Co., 941 pages, 1988

[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

[Reinhardt 87] Reinhardt, W.G.: Powerplant rebuilds confidence: Intermountain's coal-fired units are a technical and financial success, *ENR*, pp. 24-28, January 29, 1987

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

[Retz-Schmidt 88] Retz-Schmidt, G.: Various Views on Spatial Prepositions, *AI Magazine*, Vol. 9, No. 2, pp. 95-105, Summer 1988

[Rieser 80] Rieser, J.J., Lockman, J.J., Pick, H.L.: The Role of Visual Experience in Knowledge of Spatial Layout, *Perception and Psychophysics*, Vol. 28, pp. 185-190, 1980

[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

[Rosenman 86] Rosenman, M.A., Gero, J.S., Oxman, R.: An Expert System for Design Codes and Design Rules, pp. 745-758 in [Sriram 86] Sriram, D., Adey, R. (editors): *Applications of Artificial Intelligence in Engineering Problems*, Southampton University, UK, April 1986

[Roth 85] Roth, J., Hashimshony, R., Wachman, A.: Generating Layouts with Non-Convex Envelopes, *Building and Environment*, Vol. 20, No. 4, pp. 211-219, 1985

[Roth 88] Roth, J., Hashimshony, R.: Algorithms in graph theory and their use for solving problems in architectural design, *CAD*, Vol. 20, No. 7, pp. 373-381, September 1988

[Rowings 84] Rowings, J.E., Walker, R.O.: Construction Energy Use, *ASCE Journal of Construction Engineering and Management*, Vol. 110, No. 4, pp. 447-458, December 1984

[Rychener 88] Rychener, M.D. (editor): *Expert Systems for Engineering Design*, Academic Press, 306 pages, 1988

[Schank 77] Schank, R.C., Abelson, R.P.: *Scripts, Plans, Goals, and Understanding*, Lawrence Erlbaum Associates, Hillside, NJ, 1977

[Schulman 87] Schulman, R., Hayes-Roth, B.: *ExAct: A Module for Explaining Actions*, Report KSL 87-8, 15 pages, January 1987

[Seehof 67] Seehof, J.M., Evans, W.O.: Automated Layout Design Program, *Journal of Industrial Engineering*, Vol. 18, pp. 690-695, 1967

[Sheu 88] Sheu, P. C-Y.: VLSI Design with Object-Oriented Knowledge Bases, *CAD*, Vol. 20, No. 5, June 1988, pp. 272-280

[Shoham 85] Shoham, Y.: Naive Kinematics: Two Aspects of Shape, Chapter 4 in [Hobbs 85], *Commonsense Summer: Final Report*, Center for the Study of Language and Information, Stanford University, Stanford, California, Report No. CSLI-85-35, October 1985

[Shragowitz 88a] Shragowitz, E., Lin, L., Sahni, S.: Models and Algorithms for Structured Layout, *CAD*, Vol. 20, No. 5, June 1988, pp. 263-271

[Shragowitz 88b] Shragowitz, E., Lee, J., Sahni, S.: Algorithms for physical design of "sea-of-gates" chip, *CAD*, Vol. 20, No. 7, September 1988, pp. 382-397

[Simon 69] Simon, H.A.: *The Sciences of the Artificial*, 2nd Edition, MIT Press, 247 pages, 1985 (1st Edition 1969)

[Simon 73] Simon, H.A.: The Structure of Ill-Structured Problems, *Artificial Intelligence*, No.4, pp. 181-201, 1973

[Slade 87] Slade, S.: The Yale Artificial Intelligence Project: A Brief History, *The AI Magazine*, pp. 67-76, Winter 1987

[Slade 88] Slade, S.: *Case-Based Reasoning: A Research Paradigm*, Yale University, Department of Computer Science, New Haven, Conn., YALEU/CSD/RR#644, 38 pages, August 1988

[Smithers 89] Smithers, T.: AI-Based Design Versus Geometry-Based Design or Why Design Cannot be Supported by Geometry Alone, *Computer-Aided Design*, Vol. 21, No. 3, pp. 141-150, April 1989

[Smock 87] Smock, R., Reynolds, M.: New utility generating plants: When, where and by whom, *Power Engineering*, Vol. 91, pp. 15-21, April 1987

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

[Sriram 85] Sriram, D., Joobbani, R. (editors): Special Section on AI in Engineering, ACM, *Sigart Newsletter*, No. 92, pp. 38-127, April 1985

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

[Steadman 83] Steadman, J.P.: *Architectural Morphology*, Pion Limited, London, 276 pages, 1983

[Stefik 81a] Stefik, M.: Planning and Meta-Planning (MOLGEN: Part 1), *Artificial Intelligence*, Vol. 16, pp. 111-140, 1981

[Stefik 81b] Stefik, M.: Planning and Meta-Planning (MOLGEN: Part 2), *Artificial Intelligence*, Vol. 16, pp. 141-170, 1981

[Steinberg 87] Steinberg, L.I.: Design as Refinement Plus Constraint Propagation: The VEXED Experience, *Proceedings of the 6th International Conference on Artificial Intelligence* (AAAI 87), Morgan Kaufmann, Los Altos, CA, Vol. 2, pp. 830-835, 1987

[Stiny 75] Stiny, G.: *Pictorial and Formal Aspects of Shape and Shape Grammars*, Birkhaeuser, Basel, 1975

[Stiny 80] Stiny, G.: Introduction to Shape and Shape Grammars, *Environment and Planning B*, Vol No.7, pp. 343-351, 1980

[Stiny 82a] Stiny, G.: Spatial Relations and Grammars, *Environment and Planning B*, Vol. No.9, pp. 113-114, 1982

[Stiny 82b] Stiny, G.: Shapes are Individuals, *Environment and Planning B*, Vol. No.9, pp. 359-367, 1982

[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

[Sussman 80] Sussman, G.J., Steele, G.L. Jr.: CONSTRAINTS - A Language for Expressing Almost-Hierarchical Descriptions, *Artificial Intelligence*, Vol. 14, pp. 1-39, 1980

[Talmy 83] Talmy, L.: How Language Structures Space, Pick, H., Acredolo, L. (eds.), *Spatial Orientation: Theory, Research and Application*, Plenum Press, pp. 225-282, 1983

[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

[Tatum 85] Tatum, C.B., Vanegas, J.A., Williams, J.M.: *Constructability Improvement during Conceptual Planning*, Technical Report No. 290, Department of Civil Engineering, Stanford University, 119 pages, November 1985

[Thorndyke 81] Thorndyke, P.W.: Distance Estimation from Cognitive Maps, *Cognitive Psychology*, Vol. 13, pp. 526-550, 1981

[Thorndyke 82] Thorndyke, P.W., Hayes-Roth, B.: Differences in Spatial Knowledge Acquired from Maps and Navigation, *Cognitive Psychology*, Vol. 14, pp. 560-589, 1982

[Toshinori 85] Toshinori, W. et al: An Expert System for Computer Room Facility Layouts, *Expert Systems and their Applications, 5th Int. Workshop*, France, 1985

[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 88] Tommelein, I.D., Gupta, A.: *Conceptual Structures for Spatial Reasoning*, Stanford University, Department of Computer Science, 52 pages, 1988, unpublished

[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*, 6-8 June, San Francisco, pp. 340-347, 1989

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

[Tsang 87] Tsang, W.W., Chu, D.P., Ling, T.W., Koh, L.S.: *An Expert System on Site Planning for Computer Equipment*, Department of Information Systems and Computer Science, National University of Singapore, Discs Publication No. TRG 4/87, April 1987

[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

[US Army 87] U.S. Army Corps of Engineers: *Safety and Health Requirements Manual*, 433 pages, Revised Oct 1987

[vandeBrug 86] van de Brug, A., Bachant, J., McDermott, J.: The Taming of R1, *IEEE Expert*, No. 1, pp. 33-38, 1986

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

[Van Hentenryk 88] Van Hentenryk, P., Carillon, J-P.: Generality versus Specificity: an Experience with AI and OR Techniques, *Proceedings AAAI 88*, pp. 660-664, 1988

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

[Warland 70] Warland, D.E. (editor): *Modern Building Construction*, Sir Isaac Pitman & Sons, Ltd. London, U.K.,

[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

[Washington 89] Washington, R., Hayes-Roth, B.: Data Management in Real-Time AI Systems, Report No. KSL 89-06, 10 pages, March 1989

[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

[Weinzapfel 75] Weinzapfel, G., Handel, S.: IMAGE: Computer Assistant for Architectural Design, pp. 61-97 in Eastman 75] Eastman, C. M. (editor): *Spatial Synthesis in Computer-Aided Building Design*, Halsted Press, New York, 333 pages, 1975

[White 81] White, A.G.: *Computers in Architecture–Layout Designs: A Short Sourcelist*, Architecture Series: Bibliography, vance Bibliographies, Monticello, Illinois, 6 pages, September 1981

[Whitehead 65] Whitehead, B., Elders, M.: The Planning of Single-Storey Layouts, *Building Science*, Vol. 1, No. 2, pp. 127-139, 1965

[Whitehead 78] Whitehead, B.: Planning the Layout of a Group of Buildings, *Proceedings CIB W-65*, Haifa, Israel, pp. III-289 to III-296, 31 October-2 November 1986

[Whittington 70] Whittington, R.O.: Organization of a Building Project, Chapter 12 in [Warland 70]

[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, 12 pages, November 1971—iteratively compose arrangements from primitive rectangular objects, can generate solutions that may not be satisfactory for grading, only final configuration is tested for satisfaction.

[Yoshida 86] Yoshida, K., Kobayashi, Y., Ueda, Y., Tanaka, H., Muto, S., Yoshizawa, J.: Knowledge-Based Layout Design System for Industrial Plants, IEEE Computer Society, *Proceedings of the Fall Joint Computer Conference*, Dallas, Texas, pp. 216-222, 2-6 November 1986

[Zabilski 89] Zabilski, R. J.: Presented in 3-D, *Civil Engineering*, pp. 48-50, June 1989

Appendix A

Interviewees

The reader should be aware that if there are any errors in the description of any of the projects, then I am the only person to blame for this. In no way should any of the people interviewed and mentioned in my dissertation be held responsible.

Walter E. Smalley, Design Mechanical Engineer

Rick Scott, Mechanical Engineering Assistant

Mechanical Engineering Section
Design & Construction Division
Los Angeles Department of Water and Power
111 North Hope Street
Los Angeles, CA 90051

Reed W. Nielsen, Field Construction Manager

John Knight, Lead Mechanical Coordinator

Bechtel Construction, Inc.
Rt. 1 Box 1000
Delta, Utah 84624

Gary T. Rose, Site Construction Manager

Intermountain Power Project
Rt. 1 Box 824
Delta, Utah 84624

Burr L. McCorkle, Project Engineer

Jim L. Richter, Design Engineer

Black & Veatch Engineers-Architects
1500 Meadow Lake Parkway
Kansas City, Missouri 64114

George D. Spindle, Field Construction Manager

Dean Bursheim, Electrical Superintendent

Ray Sedey, Piping Superintendent

Bechtel Construction, Inc.
P.O. Box 61
King City, CA 93930

Ken Reinschmidt, President, Advanced Systems Development Division

Gavin Finn, Project Manager for Expert Systems Development

Ronald L. Wagner, Consultant

Hank Pagliarulo, Construction Engineer

Bob Clay, Construction Engineer

Stone & Webster Engineering Corporation
245 Summer Street
Boston, Massachusetts 02107

R.G. Franklin, Construction Manager

Kaiser Engineers, Inc.
Cardinal Cogen Plant
P.O. Box G
Stanford, California 94305

Edward D. Ritchie, Vice President of Operations

Dillingham Construction Corporation
P.O. Box 1089
Pleasanton, California 94566

Appendix B

Temporary Facilities

Temporary Buildings: Buildings on site essential to support construction operations, but not needed for the permanent operation of the project that is constructed. Some may remain on site after completion of the project, for example, to serve as maintenance facilities.

project management office
contractor or subcontractor facilities
personnel office
carpenter shop, sandblasting, paint shop, rebar shop
welder test and training building
material testing laboratory (such as for welds and concrete)
pipe fabrication warehouse
general warehouses (air-conditioned, temperature controlled) for electical, mechanical, instrumentation
storage for paints, lubricants, fuels, solvents, water,...
change houses for different crafts
general classrooms
brass-alleys
timekeeper's office
tool sheds
guard houses
payroll office
public relations center, visitor's facilities
eating facilities
lodging facilities or workers' camp

Fabrication Yards: Places on the site used for partial assembly of components before they are integrated in the permanent structure.

Storage areas: Places on site where materials are left for a short or long time period.

Laydown areas: Places for putting material down between the time of delivery to the site and the time of its use in construction and assembly into larger parts or its final storage.

Work areas: Places where workers assemble or fabricate components and materials.

Staging areas or short-term laydown areas: Areas located immediately adjacent to construction sites and used in support of the construction operations then ongoing [Neil 82].

Long-term laydown areas: Areas more remote than the short-term laydown areas, used for storing items not required for immediate use in the construction (structural, concrete, lumber yard, electrical, mechanical, instrumentation) [Neil 82].

fabrication yards or shops
equipment maintenance area
reinforcing steel mat prefabrication area
concrete batch plant
rock crushing facilities
open secure storage
open laydown areas
fencing, fenced storage
equipment maintenance center
surplus fill and material disposal

Site access and transportation: Access requirements and transportation needs can be provided by land, air, and water.

general roads:
construction road, materials delivery road, haul road (for supply deliveries)
personnel access road (for passenger traffic)
access roads, interior roads
turn-around points

railroads
construction railroad
delivery railroad

parking lots
craft parking
staff parking
subcontractor parking
visitor parking
owner parking

barge access

heliport

utilities
trash disposal
sanitation, toilets
site safety, fire station, 1st aid
water supply and distribution
construction power supply and distribution centers
construction gas
communications system (phone, walky-talkies)
yard lighting
fencing
information signs
electronic surveillance, security system

Appendix C

Assessing Computational Costs

The computation cost for the generation of solution layouts depends on three factors under SightPlan's control: 1) the cost of satisfying constraints, 2) the cost of generating layouts in which each object has a single position from a layout in which objects have several acceptable positions, and 3) the cost of selecting a layout if several alternatives are generated.

1 CALLS TO CONSTRAINT ENGINE

The constraint engine performs operations on essential areas of objects (see Section 4.5 and Figures 4.25 and 4.26 for the convention on the representation of essential areas). Essential areas represent the dimensions of the object, as well as, for the object's orientation at 0 and 90 degrees, the rectangular areas where the center point of the object can be located in a local two-dimensional context. For the remainder of this discussion, I will assume that each object has specified dimensions. Given this representation and my assumption, the factors that affect the efficiency of the constraint engine are: 1) the *type of the function* that is called, and 2) the *number of rectangles* in the essential area at each orientation of the object. The type of function and the number of arguments determine the time it takes for the constraint engine to process the data by calling lower level primitives.

Table C.1 shows some results of a timing study I conducted on the GS2D constraint engine, which deals with binary spatial constraints [Confrey 88]. The two left columns show the first and second argument called by the constraint. These arguments were taken from a simple data set (see below). Constraint names head the columns to the right, and underneath them are the timing results from calling the constraint on the two arguments. GS2D ran on a TI Explorer I™, and computed each constraint 300 times. The CPU time (in microseconds) in the table is the average of those 300 execution times. The measurements varied around 25% of the mean. Such large variations are inherent in Lisp machines, and are accentuated in the present tests because measured durations are very small. They make it difficult to characterize the constraint engine.

1st DATA#	2nd DATA#	NORTH	EAST	WEST	SOUTH
1	0	7329	7277	6624	5540
0	1	5485	5414	5438	5421
2	0	5405	5451	5457	5442
0	2	5493	5475	5444	5485
3	1	6917	6939	10335	5792
1	3	6077	6055	6057	6031
4	1	10305	12535	11487	10306
1	4	6941	6983	6960	6928
5	1	15164	10248	10362	10281
1	5	8777	7993	7865	6954
6	1	63973	82535	86828	88753
1	6	13673	13770	13832	13670
7	1	71380	75916	88788	79877
1	7	13710	13821	13896	13613
8	1	162862	176496	163416	184446
1	8	22548	22762	22877	22425
4	4	15539	48017	11982	11816
5	5	11889	29386	11847	11851
6	6	78482	88859	110622	86017
7	7	84298	98401	83021	84596
8	8	187945	182794	199970	193875

Table C.1: Results of the GS2D Timing Study

Each piece of data from data # 3 upward contains the same rectangle in its state family, but the number of rectangles and whether or not these represent a 0 or 90 degree orientation are different. Providing different data sets and recording information on the essential areas returning after constraint satisfaction might reveal additional properties of the constraint engine.

DATA#	ESSENTIAL AREA	DESCRIPTION
0	((200 200) ((0)(((-1000 1000)(-1000 1000)))) ((90) NIL))	1 rectangle at 0
1	((100 100) ((0)(((0 0)(0 0)))) ((90) NIL))	fixed at 0
2	((100 100) ((0) NIL) ((90) (((0 0)(0 0)))))	fixed at 90
3	((200 200) ((0)(((-1000 1000)(-1000 1000)))) ((90)(((-1000 1000)(-1000 1000))))	1 rectangle at 0 1 rectangle at 90
4	<too long to enumerate>	10 rectangles at 0, none at 90
5		10 rectangles at 90, none at 0
6		50 rectangles at 0, none at 90
7		50 rectangles at 90, none at 0
8		50 rectangles at 0 and 50 at 90

Experimental results and further rationalization about the constraints' representations and effects support the following characteristics of the constraint engine:

- **CONSTRAINT TYPE**

Some constraints execute faster than others that are applied to the same arguments. For example, PARALLEL computes more rapidly than CLOSER-THAN.

- **COMPLEXITY OF ESSENTIAL AREA**

A Constraint generally executes faster when the essential areas of their arguments have a smaller number of rectangles.

- **0/90 DEGREE ORIENTATION**

Constraints such as PARALLEL change their arguments upon constraint satisfaction when one of their arguments only has a single orientation, and have no effect otherwise.

- **AREA OF RECTANGLES IN THE ESSENTIAL AREA**

For some constraints, the *area of a rectangle*, which specifies possible locations of an object's center point, affects the computation. One extreme is that, if both objects' areas are large compared to both objects' dimensions, then the application of constraints like CLOSER-THAN (see figure 4.27 for an application of the CLOSER-THAN constraint) or NON-OVERLAP might not reduce the essential area of either of its arguments. The other extreme is that, if one of the objects has a fixed position, then the application of these same constraints reduces the essential area of the second object as much as is possible.

- **NUMBER OF RECTANGLES IN THE ESSENTIAL AREA**

When SightPlan samples an object's set of possible locations, the result is a finite set of point locations. Although the essential areas is 0 in that case, the number of rectangles could be very high, thus taxing any computation.

Two performance criteria can be set for the constraint engine: 1) require a quick response time for a single constraint's satisfaction, and 2) require simple essential areas to be returned, so that further computation can be fast.

Given these criteria and from the above observations, one may conclude that a strategy that builds on effective use of the constraint engine may rank constraints by relative execution speed in the following order of priority for execution, such as PARALLEL, CLOSER-THAN, ADJACENT-TO. In fact, this linear ranking does not suffice. The ranking should be at least a second-order function that also takes into account the arguments of the functions. These arguments, in turn, are characterized by a number of rectangles in their essential area and by those rectangles' areas. In this research, however, I have not gone so far as to derive that second-order function. My assumption is that it was not worthwhile to do so. Even if I could identify this second-order function, and made it part of the Computational Strategy, BB1 might spend so much time on reasoning about which constraint to execute that the computational benefits of prioritizing a given constraint over another may be lost. Also, using the constraint engine's efficiency to determine control is adopting a local view on optimizing problem solving.

Note how an early-commitment strategy, as implemented in the Expert Model, and adapted to a human constraint engine, applies constraints in an order that is efficient for GS2D as well. For example, early commitment strives towards finding unique positions for objects early on. Objects with unique positions are represented by a very simple essential area, and this makes subsequent constraint satisfaction very efficient. Zoning objects in sub-areas on site reduces the sets of locations originally covering the entire site.

While the efficiency of the constraint engine can be assessed approximately, it is much more difficult to assess the cost of generating coherent instances for objects with sets of possible locations, or to assess the cost of evaluating solution layouts.

2 GENERATION OF COHERENT INSTANCES

A *coherent instance* of a layout consisting of objects that have multiple possible positions is a layout in which each of the objects has one position, which is selected from the object's multiple possible positions, and in which the objects do not overlap each other.

The need for computing coherent instances arises in the following situation: Assume SightPlan solves an underconstrained problem, and pursues a strategy that chooses *not* to apply preference constraints (preference constraints, for example, were the "as-close-as-possible" constraints used in the Expert Strategy). After SightPlan applies all (non-preference) constraints defined in the problem, the result is a layout consisting of objects that have multiple possible positions. The system, however, does not have further knowledge or selection criteria to reduce positions to single instances, and yet, the user of SightPlan is interested in obtaining at least one *solution layout*, that is, a layout in which all objects have a unique position. In that case, SightPlan should be capable of computing coherent instances from its current layout, each of which will constitute a solution layout.

Generating coherent instances consists of 1) choosing one or more single instances of positions for each object that has multiple feasible locations, and 2) verifying that objects do not overlap each other when they are located at their chosen position in a layout. While verifying lack of overlap is straightforward there are many ways to choose positions. The more instances are selected, the more combinations of coherent instances are likely to be generated. The cost for generating such combinations of objects and verifying that no two objects in any combination overlap increases with the number of instances that are picked and the number of objects in the layout. It is therefore crucial that this brute-force approach for generating layouts be applied only at opportune times. For example, a system could intelligently sample positions before attempting to generate coherent instances. The cost of computing coherent instances is orders of magnitudes greater than the cost of satisfying one constraint. Clearly, computing a coherent instance involves, among other things, meeting many non-overlap constraints.

3 EVALUATION OF SOLUTIONS

In the process of generating a solution one might decide that, because any of the coherent instances satisfies all constraints that were given in the problem definition, any coherent instance is a satisfactory solution to the problem. Additional criteria might be added to determine the quality of different coherent instances. That is, one might further constrain the problem, or one might define an evaluation function. In order to assess how expensive it is to reach the final solution, the cost for evaluating alternatives ought to be included in the total generation cost.

Although it may be worthwhile to set up evaluation functions and assess how effective they are, I did not devote much time to this possibility. While I was learning from field experts how they go about laying out sites, I found confirmation for my intuitive idea that field managers opportunistically define additional constraints to guide their search towards a solution. None of the field managers I talked to articulated a way to assess the value of a layout globally, so I did not derive any evaluation functions from them. The reader who is interested in evaluation functions may review related work in operations research (see section 2.3.2). To show that SightPlan is capable of evaluating a layout when given the criterion to do so, I implemented I simple scoring function (see Section 6.3.3).