Title
Conceptual Structures for Spatial Reasoning

Authors
Tommelein, Iris D
Gupta, Anil

Publication Date
1987

DOI
10.34942/P2BC7G

Peer reviewed

CS309A - Conceptual Structures
Prof. John F. Sowa
Term Paper - Fall 1987

Conceptual Structures for Spatial Reasoning

by

Iris Tommelein and Anil Gupta

Department of Computer Science
Stanford University

Table of Contents

List of Figures

1 Introduction

Spatial reasoning plays a major role in human cognition. We see and recognize objects and scenes in our environment, so we are able to identify where we are, and we are capable to move around in it without touching obstructions. We can describe spatial objects and have languages to communicate that information, for instance, we may explain to other people how to find the path from one location to another, or we succeed in programming a robot to move through space. We can design and create artifacts which we construct, for instance, by shaping and assembling physical parts. In short, we continuously represent space and reason about spatial properties of our surroundings.

Any representation can only show a model of the world. In this paper we will investigate what means are provided by our language to translate a spatial world into an English description (encoding), and vice versa, to model a set of spatial relationships into a graphical representation using a given vocabulary (decoding). A brief literature review, in Section 2 of this paper, illustrates the breath of application of spatial models.

English --as many other languages-- provides a wealth of words to describe spatial entities and relationships, but its semantics are not always clear or uniquely defined. In Section 3, we will show the results of compiling a list with vocabulary of spatial terms. Later we will pick a few terms from that list to illustrate a simple model of spatial properties in a specific domain: that of designing and graphically displaying the layout of a construction site.

The language in which such models are expressed contain simplifications of the representation of space in that it abstracts geometrical and topological properties of objects. We will discuss abstraction mechanisms in general in Section 4 by using various examples, and we will show how idioms and word usage dictate which abstractions are appropriate in a given situation.

Then we will apply the formalism of conceptual structures to model this vocabulary of spatial terms for site layout. Occasionally we will also incorporate temporal terms, which can be treated in a way analogous to one-dimensional spatial terms, but we did not get into the details of expressing temporal relationships. Conceptual graphs provide a comprehensive representation of knowledge, and we refer the reader who is unfamiliar with the subject to [Sowa 84] for an introduction. After defining conceptual graphs we can further extend our model by adding schemata and prototypes to capture information of the selected domain of application (Section 5). We limited ourselves to an

abstraction level that uses a 2-dimensional orthogonal representation of the world. Recognizing that a representation at this level strongly limits the expressiveness of the language and may be counter-intuitive and/or grammatically incorrect in many instances, and despite encountered complications, we will be able to build a working model that illustrates and stresses important representation issues in spatial reasoning.

Our first goal in this work is to tentatively illustrate that people succeed in creating mental images of a situation when they are given only a partial description of a scene. A person may directly infer the answers to questions about spatial properties, or may need to add prototypical and schematic default information in order to be able to generate plausible answers. In Section 6 we will illustrate a simple model of this process on an example text which is first translated into conceptual graphs.

Our second goal is to develop a system that assists people in designing a layout that represents a set of given objects which have to meet specified spatial constraints. To achieve this, the specification is given to a constraint engine that generates the family of acceptable layouts. We use the blackboard expert system framework, BB1, that provides an environment for reasoning about actions taken in the process of designing a spatial arrangement. In Section 7 we will explain the principles of this model, named SightPlan, without however getting into all of its detail.

Finally, in Section 8 we summarize our work, we formulate conclusions and we point at further extensions for the spatial models that we introduced. The scope of this work is to apply conceptual structures to model space, to scan major issues that arise in modeling spatial reasoning, and to elaborate on difficulties encountered in the conceptualization of space. We illustrate this by means of applications that perform the tasks of recoding and decoding. The provided examples should permit the reader to obtain a reasonable insight in the nature of the problems encountered while dealing with languages to model space. A detailed discussion of this subject, however, is beyond the scope of this paper.

2 Languages and Models for Encoding, Decoding, and Recoding Space

2.1 Representing Space

Let us start with a few tentative definitions. **Spatial language** consists of the words and symbols that describe the geometrical and topological properties of an individual object, or that represent the geometrical and topological relations between several objects in a given context. The

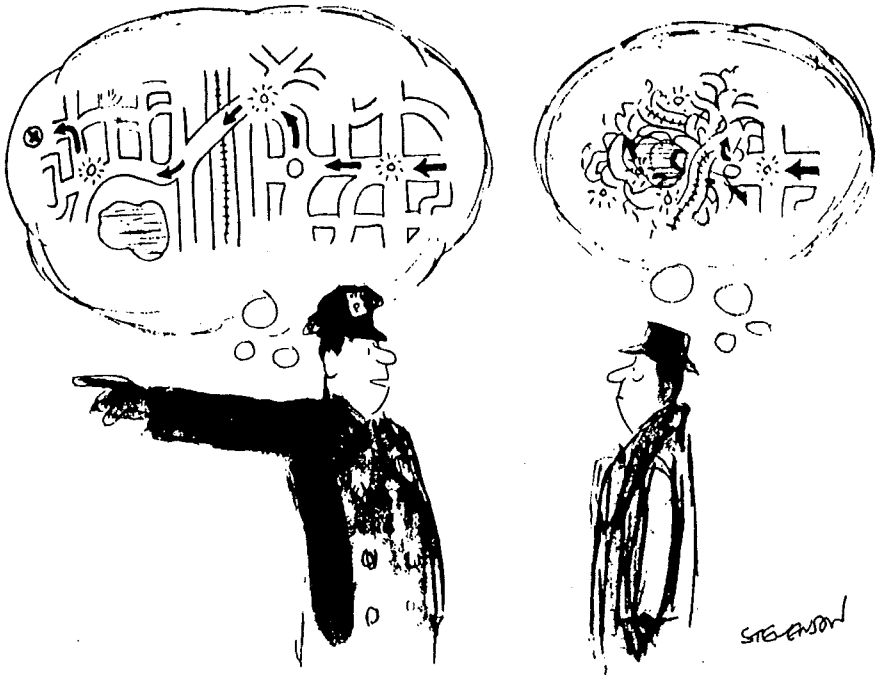
process of *generating a linguistic description* from a situation involving spatial objects or from its graphical representation is called **encoding**; the process of interpreting a locative construction and *generating a graphical representation* for it is called **decoding**; and the process of translating from one description or locative construction to another one --without using any *graphical representation*-- is called **recoding (Figure 1)**.[1]

---

1 We provided these definitions because we find them intuitively clear and they are useful in the further discussion of spatial models. However, they are not as crisp as we would like them to be, and therefore we labelled them as *tentative*. In fact, we could not find a way to clearly distinguish graphical representation languages from other languages.

In the model we propose in this paper we will solely rely on conceptual graphs for the representation of spatial entities. Note however that researchers who work on "visual mental imagery" question that a single system of abstract propositions could suffice to account for all internal representation of space, and suggest that such semantic conceptual store needs to be augmented with an internal spatial representation (see for instance [Kosslyn 77]).

An excellent argument in favor of requiring such augmentation is the following cartoon (Figure 2):



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

Figure 2: Using Language to Express a Mental Model
from [Sowa 83, p. 21]

The man and the police agent use verbal language to communicate, but each individual maps the communicated information to his own internal mental map, and some information is lost in the process.

The irony of the joke is captured perfectly in its pictorial display; a verbal description cannot achieve the same expressive power as an image can.

In these definitions we allow the **graphical representation** to be the "world" itself, or, alternatively, any graphical or pictorial representation thereof (*e.g.*, a picture, a drawing, a projection, a perspective, or a sketch).

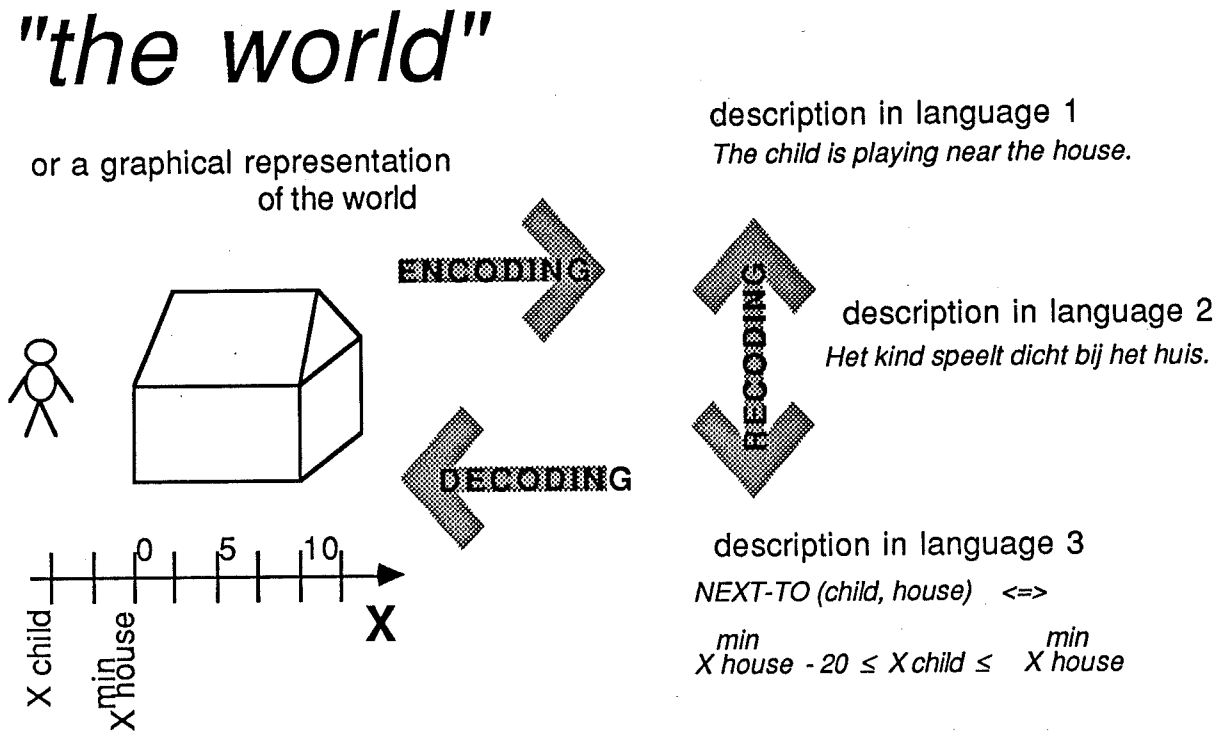


Figure 1: Translation Processes

Typical to these coding mechanisms is that details may get lost in the process, so that the coded language only represents a partial description of what has been coded. This, in fact, can be a desired effect: translation processes may be intentionally invoked to obtain simplification. In Section 4 of this paper we provide more detail on abstraction mechanisms that are at play in language, independently of whether or not the language user is conscious of it.

In the following paragraphs we will provide a brief review of spatial models which elaborate on various issues and difficulties encountered in the coding of space. We classified these models in terms of their decoding, encoding or recoding task.

2.2 Example Models

2.2.1 Encoding

Encoding is the process of generating a linguistic description from a situation involving spatial objects in the world or from a graphical representation thereof. It is perceived as a rather

easy task because many conventions exist on how to represent "maps" so that their interpretation would be easily understood.

❑ Interpreting a Map

A graphical display (for instance, a road map of an area) can easily be used to answer questions that relate to the spatial properties for which representation the map was designed. If such a map was generated based on standards for the true representation (= maintain proportionality of objects) and consistent representation (= show all objects of a same type at the same chosen level of abstraction) of spatial properties such as topology and geometry, then it is known in advance what details are considered irrelevant and were therefore omitted (*e.g.*, a highway map will not show rural roads or back country trails). In addition to these standards, cultural conventions for display are adopted (*e.g.*, "North" points to the top of the page). The interpretation of maps becomes more difficult if they display uncommon local landmarks or if they are taken from an unusual viewpoint (Figure 3).

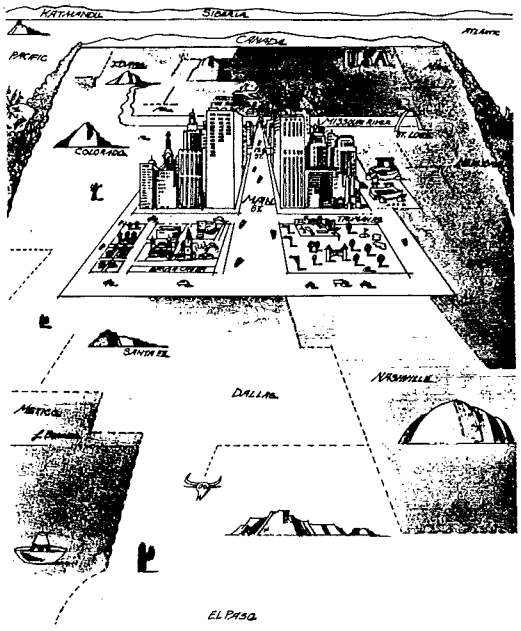


Figure 3: Map of Kansas

Interpreting a map or graphical display could be tricky if its conceiver had something unconventional in mind, for instance, an artist could carefully handcraft a drawing which introduces a paradoxical situation (Figure 4).

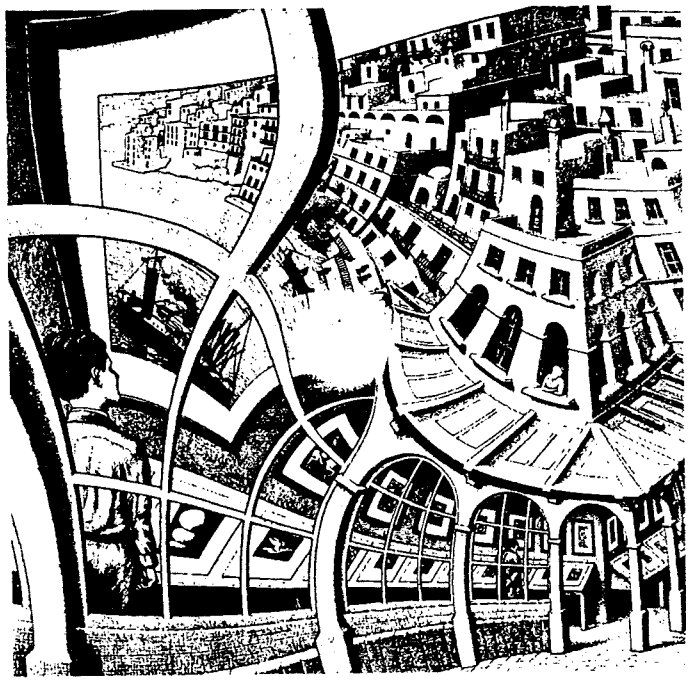


Figure 4: Paradox in Graphical Language
"Print Gallery" by M.C. Escher (Figure taken from [Bool 81])

In particular, a person who is sketching an idea, and creates a graphical display of it, does not necessarily think of the exact dimensions of what is being drawn. The result of the process however is a dimensioned drawing, and will spontaneously be interpreted as such by other viewers. (This is the main reason, we think, for which it is difficult to build computer tools that assist people in sketching out ideas. Sketches always contain unintended geometric information).

Other encoding applications include systems that generate linguistic descriptions to replace descriptions at the level of physical coordinate space or at the level of a geographical display.

❑ Robot Instruction ([Popplestone 75])

Robot instruction requires an explicit specification of coordinates for the movement of robot arms and the parts it holds. The process would be much simplified if one could use English-like spatial relations to describe the relative position of parts being manipulated in successive stages of the assembly process. In particular, the results of investigating the meaning of "against" and "fits" are presented in this paper.

❑ **Providing Directions (CITYTOUR - [Retz-Schmidt 88])**

Given a scene with true representation of topology and geometry, it may be easy to express relationships in terms of numeric coordinates, yet it is not trivial to do so in natural language. Relative positions of observers with respect to moving objects or oriented objects play a role in determining the prepositions that need to be used; not a simple task!

2.2.2 Decoding

Decoding is the process of interpreting a locative construction and generating a graphical representation for it. It is perceived as a hard problem because a locative construction usually contains only part of the information that is needed to reconstruct its graphical representation.

❑ **Generative Design Expert System (here called GDES - [Flemming 86, 88])**

GDES makes use of a domain independent generator to systematically enumerate all alternative solutions to a design problem, and subsequently sends the results through a tester to evaluate the resulting classes of layouts. Classes of layouts are represented as two-dimensional graphical displays, and the displayed images are tied to their underlying directed graphs that use topological relations "above", "below", "right", and "left" between rectangular objects.

❑ **Layout Design (CONSITE - [Hamiani 87]; SightPlan - [Tommelein 87a, 87b])** Given rectangular objects to be positioned in two-dimensional space and constraints (locative expressions) between them, generate all layouts where those objects can possibly be; the results are to be graphically displayed so that a person who monitors the program is assisted in building his or her internal mental picture of the design, and he/she can manually introduce preferences in order to prune the solution tree. SightPlan is discussed in more detail in Section 7 of this paper.

2.2.3 Recoding

Recoding is the process of translating from one description or locative construction to another one (without generating an entire graphical representation). It is also perceived as a hard problem for the same reason for which decoding is hard: locative constructions contain partial information, so new constructions can be correctly generated only if they make solely use of the information present in the input description. When default information is assumed to complement that input the resulting constructions may be incorrect.

❑ **Route Planning based on Underspecified Descriptions (TOUR - [Kuipers 77])** TOUR is provided with multiple representations of a large-scale space, and it solves route finding and relative position problems. The representation is based on states of partial knowledge that describe either a route (using primitives for "turn" and "go-to"), a topological structure ("place" and "path"), a relative position of two places, a dividing boundary, or a region in space. Inference rules allow the robot --who uses that knowledge-- to put together different states of such underspecified geometrical and topological descriptions, so that it can determine which path to follow.

❑ **Inferring Answers from a Fuzzy Description of Space (MERCATOR - [Davis 81, 84])** MERCATOR takes observations made by a robot that looks around in an environment and builds an internal knowledge base consisting of this incomplete information, so that queries about the spatial properties of the environment could be answered. The focus of this work is on how imprecise and incomplete spatial information could be stored and combined. Davis' representation makes use of fuzz ranges on dimensions of objects, on distances between objects, and on angles between edges.

2.3 Summary

In this section we have differentiated between models for the encoding, decoding and recoding of spatial descriptions, and we used this distinction to classify research work in the field of spatial reasoning. We may conclude that the main result of coding is the loss of information, which is reflected by the underspecified and fuzzy output of the process.

In the following section we will look more closely at the English language and investigate the nature of its words that allow us to express spatial properties. As an illustrative example we then elaborate in more detail on issues of expressiveness, ambiguity and limitation of English to express two-dimensional spatial relationships.

After that, we will focus on how spatial information gets lost in the process of abstraction as dictated by English word-use and idioms.

3 Vocabulary of Spatial Terms

We chose the English language to investigate the nature of words that allow us to express spatial properties. Describing a situation in any language --even languages dedicated to represent space-- leave much ambiguity to be resolved by the interpreter. English, as well as graphical

representation language, cannot be freed from the need for additional contextual information to provide clarification (Figure 5).

Figure 5: Ambiguity in Graphical Language
"The Shadows" by René Maguerite (Figure taken from [Hofstadter 80])

3.1 Spatial Representation using English Vocabulary

Anil and I compiled a vocabulary of English spatial terms by performing a dictionary search in the Longman Dictionary [Longman 78] and in a Synonym Finder [Rodale 78], and by holding a few brainstorming sessions. A sample of the wealth of words we thus obtained is provided in Appendices 1 and 2, but note that it has not been our intent to exhaustively generate all spatial terms, nor do our comments and classifications try to capture all possible cases and expressions on spatial terms. It is in the process of sorting out distinctions in meaning of words that we realized rigorous simplifications would be needed for a representation in which we want to single out crisply defined interpretations for spatial relations. In this paper we will further limit ourselves to discussing terms based their elementary geometric interpretations. This is a sufficient first approximation for the purposes of the selected examples, but this does definitely not capture the expressiveness and flexibility provided by language. Please refer to the studies of other researchers who discuss linguistic issues in use of spatial terms (see for instance [Herskovits 82, 85, 86][Talmy 83][Kautz 85][Retz-Schmidt 88]).

Spatial terms --mainly nouns, adjectives, verbs, and prepositions-- can be divided in: 1) those that describe objects themselves (typically nouns with modifying adjectives and verbs used as adjectives), and 2) those that express relationships between two or more objects (typically prepositions).

3.1.1 Terms Describing Objects' Dimensions and Shape

A *noun* that represents an object can contain information about the spatial properties of that object. Such properties often are essential to the identification of the object and therefore are called natural attributes which will be part of the object's type definition. For instance, a *shape* is determined by words such as "a rectangle", "a swastika", "a cone"; some of the *dimensions* are determined by "a yardstick", a "two-by-four".

Adjectives can add spatial information about additional attributes, often by expressing a *comparison* between the object and another object of the same type that is considered prototypical in a particular context. For instance, the *length to width ratio* of a certain object type is restricted from its normal ratio by talking about "a long rectangle" and "a narrow path". A "long distance" may mean a distance of several miles where a typical distance is only a few hundred yards; "a winding road" is a road quite different from a straight one.

3.1.2 Terms Describing Relationships between Objects in a Context

Terms describing geometrical and topological relationships between objects usually are spatial *prepositions*, also called locatives. They will be modeled as relationships in the formalism of conceptual graphs. Typical word-use and idioms of the language dictate which preposition is the correct one in a given situation, *e.g.*, "she sits **in** the car", but "he is **on** the bus".

Besides prepositions, other terms of the language may also express a spatial relationship between objects. For instance, a *noun* representing an object can describe information about the *distribution* in space of several of such objects: *e.g.*, "a milestone".

In summary: terms can describe the dimensions and shape of individual objects, or relationships between two or more objects.

3.2 Expressiveness, Ambiguity, and Limitation of Language

In selecting words for a language one has to make the trade-off between simplicity (How easy is it to express what needs to be said?), expressiveness (Can one say exactly what needs to be said?), limitation of vocabulary (Is there a single term to express what needs to be said?), and richness (Does a word have multiple/ambiguous meaning?).

To illustrate these issues, we have expressed all possible topological relations between two segments, and we have graphically displayed all those possible between two rectangles, after which we tried to come up with words to label each relation.

When all topological relationships between two intervals in one-dimensional space are drawn, 13 combinations can be distinguished (Figure 6). [Allen 84] lists these topological relations and chooses a set of words in English to uniquely name them: "before", "equal", "meets", "overlaps", "during", "starts", and "finishes". Used in this context, some of the words retain their intuitive meaning (*e.g.*, between, before) while others need to be interpreted in a rather restricted way (*e.g.*, starts, finishes).

Relation	Symbol	Symbol for inverse	Pictoral 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* *YYYYY*
X finishes *Y*	f	fi	*XXX* *YYYYY*

Figure 6: 13 Topological Relations Between Two Intervals
(Figure taken from [Allen 84])

When all topological relationships between two rectangles in two-dimensional space are drawn, 169 combinations can be distinguished (Figure 7). Yet, the English language does not provide 169 *single* words to uniquely express each of these (Give it a try, naming them all! Then imagine how many words you would need to label relations between arbitrary shapes in two-

dimensional space; then try three-dimensional space...). What we can say about two objects with respect to one another is for instance that they (are): "adjacent", "overlap", "touching", "next-to", "inside", "wider", "higher", "above", "below", "to the right", "to the left", "separated", "distant", and so on. Some of these simple terms however may capture several of the combinations of positions of two rectangles; sometimes more than one term is available to indicate the same configuration of rectangles.

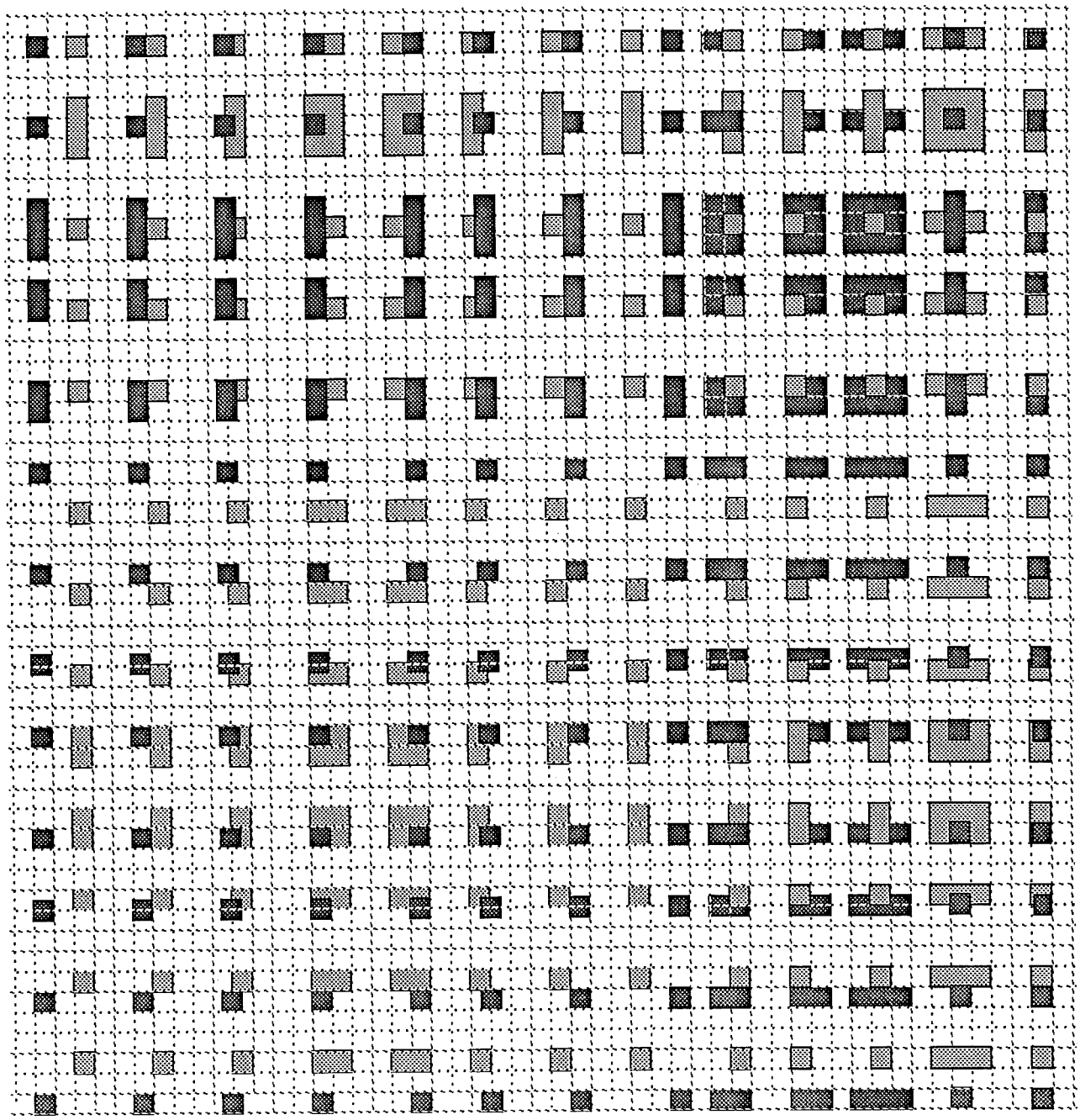


Figure 7: 169 Topological Relations Between Two Rectangular Shapes

For instance, focus on a gray and a black rectangle in Figure 7. Depending on the proportions of the sides of one with respect to those of the other, expressing that the gray rectangle is *adjacent* to the black rectangle, [RECTANGLE: *x] -> (ADJACENT) -> [RECTANGLE: *y], can be realized by the topological configurations illustrated by any of these in which the rectangles *touch* each other but do *not overlap*. So, we could conclude that being adjacent means the same as touching: [*x] -> (TOUCH) -> [*y]. Depending on the grain-size at which the model is represented, we could possibly also state that objects *next-to* each other are also adjacent. But note how the context of the situation comes into play here. Two whales swimming next-to each other may be tens of yards apart. Two ants that follow each other at a great distance may be several feet apart. Thus the distance between the whales is much greater than the distance between the ants, and yet we talk about the first as being "next-to" each other, about the second as being "at a great distance".

We conclude from this, that in order to disambiguate words we need to put them in a context and restrict their meaning to be very tight compared to their meaning in every-day use. On the other hand, in every-day use there may not be sufficient single words to express all desired relationships, in which case we need to select a set of basic words and combine those to provide sufficient expressiveness.

3.3 Conclusions

With the preceding examples we hope to have made clear that there exist many limitations on vocabulary in a given language --the English language for expressing spatial relationships between rectangles was only chosen as an example--, and yet, we get by in expressing what is needed in our day-by-day communication. Before we then proceed in proposing our abstraction and conceptualization of a very simplified vocabulary to represent rectangular objects in two-dimensional space we will, in the following section, first look at which abstraction mechanisms naturally occur in English. After doing so, the choices we made for our models should be more comprehensible.

4 Mechanisms for Spatial Abstraction

wisdom is knowing what to ignore

4.1 Motivations for Using Abstraction

Building a model requires choosing primitives or initial building blocks for its description or construction; selecting appropriate variables, parameters and attributes for its operation; and establishing its goals. As mentioned earlier, language naturally assists in making such choices by introducing conventions on word usage, expressions, and idioms. In addition to that, the application for which the model is designed also influences how these choices are made by referring to standardized representations in models already completed (*e.g.*, grain-size at which other models described the problem); and finally, the person who makes the choices and selects an appropriate abstraction for the model to be construed is guided by the following considerations:

1 abstract away all details irrelevant for problem-solving;

2 add desired properties to the object's description; and,

3 keep details that are "natural" for explanation.

Keeping irrelevant details will increase the amount of data to be maintained, yet some details may be worthwhile to keep, even if they don't directly contribute during the problem solving: one could understand more about the context in which the solution process took place by knowing additional facts. This is particularly useful, for instance, if the system is used by a person unfamiliar with it, or if it is applied at the verge of its intended use.

An example from the construction layout domain: a truck can be abstracted by its vertical projection on the ground for the purpose of finding its possible locations on a site. While desired properties in the description could be "geometrical symmetry" or "simple shape", one could simplify by abstracting the shape to be a rectangle that encloses this projection. Yet, one may want to explicitly keep track of which of the two short sides of the rectangle represents the front of the truck in case a study of the truck's movement on the site needs to be made and one needs to find out whether that piece of equipment can indeed reach the position where it will be used. Also, storing information on what the truck looks like would allow the system to generate a self-explaining icon for its display.

In summary, abstraction allows for different representations of the same object depending on what problem needs to be solved; abstraction must be well-adapted to fit the tools that are available to the solve the problem, and it must be comprehensive to the individuals who build the model, who observe the problem solving process and who look at intermediate and final solutions.

4.2 Mechanisms for Spatial Abstraction

The mechanisms described in the following paragraphs are spatial simplifications as they naturally occur in English expressions and idioms. In these expressions, physical objects of the "real" three-dimensional world (four-dimensional if time is added) are represented as two- or one-dimensional, or without dimensions at all, their shape is ignored, their geometry is reduced to a topological description, and so on; similarly, relations between objects are made symmetrical, or the influence of the viewer in the scene in omitted, and so on. Our purpose is to illustrate abstraction mechanisms at work. When we then propose our model of the spatial world in Section 5, we hope it will not appear to be as far-fetched or arbitrary as it may otherwise seem.

4.2.1 Reduce Dimensions

❑ **omit time** (4D -> 3D)

Study the problem as a static one rather than as a dynamic one. For instance, "Jim lives *in* a big house." -> does not take into account that Jim leaves "the house" every morning to go to work.

❑ **ignore the object shape** (= any reduction -> 0D)

This type of abstraction occurs when no explicit mention is made of the space or volume taken by objects. For instance, in location problems of operations research the distances between objects are so large that taking the dimensions of the individual objects into account upon its measurement will barely affect the results. Topological relations such as adjacency between objects are then irrelevant (see also [Stiny 82a, 82b][Kuipers 78]). Note however that one cannot omit volume entirely if a display of the object in needed.

"How *far* is San Francisco *from* Los Angeles?" -> "cities" are abstracted to points between which the distance in measured.

❑ **reduce a volume to a point** (3D -> 0D), **an area to a point** (2D -> 0D)

"He lives *in* that village." -> "village" is abstracted as a volume, but, in "the village *on* the road" [Herskovits 82 p. 82] -> "village" is abstracted as a point on the line represented by "road". Similarly, "Is Lima *on* the equator?" [Herskovits 82 p. 196] -> an entire "country" is abstracted to be a point.

❑ **reduce an interval or a segment to a point** (1D -> 0D)

"Last year I spent a *week* in Fuji. *At that time* something really funny happened to me." -> a continuous time interval described by a "week" becomes a (discrete) point.

"The man *at* the entrance gate will ask for identification." -> a "gate", often drawn as a two-dimensional segment is here abstracted to be a point.

❑ **model a volume as a surface** (3D -> 2D), **and vice versa** (2D -> 3D)

"It fell *on* the floor." -> a "floor" is thought of as two-dimensional, but "The ceiling lights *on* the third floor are not working" [Herskovits 82 p. 193] -> now it is abstracted to contain the entire volume delimited by it and the following "floor". A similar pair of examples is "a picture *on* the wall" and "a crack *in* the wall". "Park your car *in* the visitor parking lot." -> the "parking lot" is an area, here abstracted as a volume; "Trucks are allowed to park *on* site." -> the "site" is modeled as a surface; "Don't leave material lying *on* the road; put it down *in* that area over there." -> the "road" is a surface but the "area" is a volume.

❑ **model a volume as a segment** (3D -> 1D)

"that park *along* the lake-side" -> the "park" is abstracted to its longitudinal dimension which dominates over its width in the given relationship (see also "ribbonal" objects in subsection 4.2.2)

4.2.2 Simplify Shape

❑ **omit shape** (already described 4.2.1)

❑ **make shape regular**

"a round apple" -> an "apple" is seldom really round.

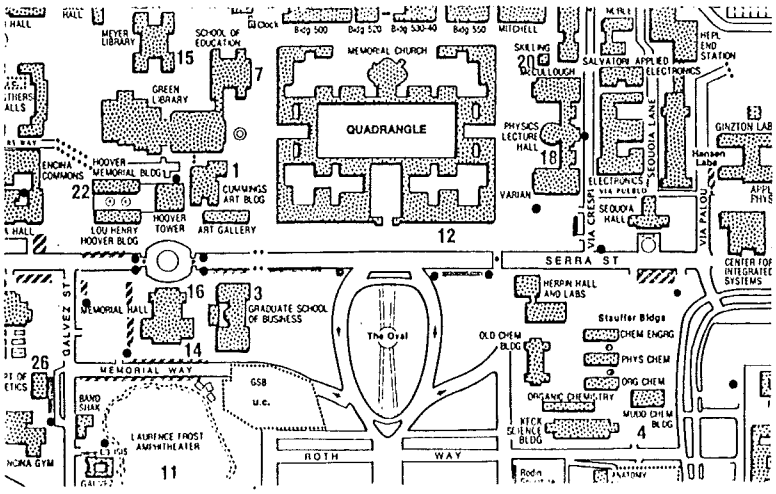


Figure 8: Simplification of Shape:
"The Oval" on the Stanford University campus

❑ **use more simple shape**

"The Oval" on the Stanford University campus (Figure 8) -> is not an oval as it would be defined in geometry. Also, other complex geometric figures are seldom used in textual descriptions, *e.g.* "hexahedron", unless precision is needed in the specification of the shape.

❑ **sharpen boundaries**

"He sat at the edge of the water." -> though water can move with the tides, "edge" is abstracted to be a (straight) line with sharp boundaries. Similarly "She was running along the edge of the road."

❑ **group individuals**

A single word is used to describe a collection of individuals, *e.g.*, mass noun "the cabbage".

❑ **"ribbonize" objects**

This is similar to modeling a volume as a segment. It is useful to define "across" for a rectangular object with one dimension much larger than the other dimension, such as a road to be crossed by a person (not a jaywalker though!). Thinking of an object as a ribbon focuses attention on two major sides of the object and emphasizes the fact that the object is *almost* linear (see [Hobbs 85] [Herskovits 82] [Talmy 83]).

4.2.3 Omit Metric, *i.e.*, move from Geometry to Topology

❑ **replace the distance metric with a qualitative interval**

Note for instance that a topological property such as adjacency depends on the granularity at which you want to round off. Compare "Those two cities are next to each other." -> still a couple of miles apart, with "We sat next to each other." -> maybe several inches, at most a foot apart.

"That is 3 miles from here." could be replaced with "That is a long walk from here.", "That is a short ride from here.", "That is a 5-minute ride from here.", and so on.

4.2.4 Simplify Relations

❑ **do away with subjective relations**

The position of the viewer and/or an interpreter in a scene determine which words are used to describe a particular relation between objects. The meaning of "In front of the car ..." depends on whether the car is moving or not, and where the speaker and the addressee are [Retz-Schmidt 88] .

The position of a third object can also impact the relation used to describe two other objects (Figure 9).

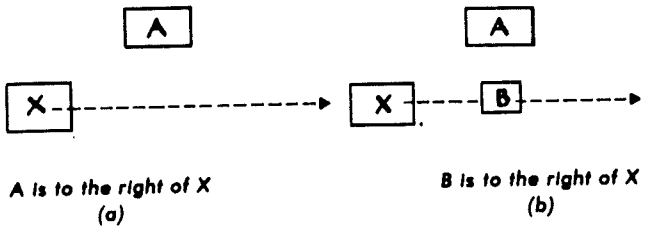


Figure 9: Presence of Objects in the Environment other than the Objects Involved in the Relation (Figure taken from [Herskovits 85 p. 353]

❑ **introduce symmetry**

One could say "The trailer *next to* the turbine building" but it would be quite unusual to talk about "The turbine building *next to* the trailer" because one object is more mobile and smaller than the other (modified from [Talmy 83]).

These are only a few examples of abstraction mechanisms at work in natural language. The interested reader can find more examples in linguistic studies such as [Herskovits 82, 85, 86], [Hobbs 85], [Retz-Schmidt 88], and [Talmy 83].

4.3 Consequences of Sections 2, 3, and 4

Up to now we have looked at models for spatial reasoning proposed by other researchers. Each of them dealt with some of the complexities of representing space. We have investigated how and to which extent language could help us in modeling and we have pointed at the types of simplifications that are idiomatically introduced by language.

Now, in Section 5, is the moment to propose our own model, for which representation we use conceptual graphs. We do so by selecting one specific problem domain and by focusing in on one example application of recoding and one of decoding.

The use and operation requirements of our models will determine the level of abstraction at which objects are represented: we want our models to only require simple computation and to have short processing time, and we will be satisfied with an orthogonal display of its results. In Sections 6 and 7 we will consecutively elaborate in more detail on each of the two models.

5 Conceptual Structures for a Selected Vocabulary

We have selected a narrow domain of application --that of construction site layout-- and two substantially different models --one is Question-Answering (recoding), the other is Design-Generation (decoding)-- in order to investigate some of the complexities encountered in spatial reasoning and to demonstrate the generality of the representation that makes use of the formalism of conceptual structures. An advantage of narrowing the field is that we can make simplifying assumptions in order to eliminate some of the vagueness and ambiguity in meaning of words, and that we can create simple models which however deal quite elegantly with spatial reasoning.

5.1 Selection of a Level of Abstraction to Reason about Construction Site Layout

Construction site layout involves the identification, selection, and spatial allocation on site of temporary construction facilities such as trailers, equipment, laydown areas, and roads. In the models we will describe next, we assume that the facilities are identified and we will focus on their spatial layout.

As the purpose of both our systems is to interpret or generate a two-dimensional display of the site arrangement, we will abstract all objects to be two-dimensional, and more specifically, to be rectangular with fixed size. All locations are approximated by positions on a two-dimensional orthogonal grid with mesh-size of one yard. Locative expressions that relate objects constitute symmetrical and mostly binary topological and geometrical relationships.

In subsection 3.2 we suggested that there already exists a serious shortage of single English words to describe each possible topological combination at the abstraction level we propose here, so that words would need to be compounded in order to express given relations; therefore we anticipate enough complexity in our models. Furthermore, in subsection 4.3 we showed that in English objects as well as areas are often abstracted to be two-dimensional; therefore we hope our models would be sufficiently close to every-day models of space so that they will be practical and easily understood. Though the simplifications sound very limiting, we found them necessary to keep our queries and processing time computationally tractable. The tight interpretations of meaning may make our results look somewhat arbitrary, yet we only claim that the models here discussed are first prototypes that could initiate more extensive research work.

In the first application the description of a particular layout is given and the goal of the model is to infer answers to questions about the spatial relationship between objects. In the second application objects and constraints are provided and the goal is to generate a spatial layout where objects meet those constraints.

5.2 Application 1: Answering Questions about a Description

Given a written description of a layout, would you have some idea of what that layout might look like, and would you be able to answer questions about spatial relationships between objects described in it? Test your pictorial imagination on the following text.

5.2.1 Input State Description

The building is located in the middle of the site. It is 100 yard long and 60 yard wide, and its dimensions are small compared to those of the site. There are two administration buildings on site, each of which is adjacent to the building and to the east thereof; and at the opposite side of the building laydown areas 1, 2, 3, and 4 are grouped. Each laydown area is allocated to a single contractor. The respective contractors on site are Joe-the-carpenter, Bill-the-piping-supplier, Ann-the-welder, and Sue-the-concrete-supplier. Each contractor has his/her own office trailer and tool trailer close to each other on his/her laydown area.

An outside road runs parallel to the north edge of the site, and is adjacent to the site. Along this road are the main entrance gate to the site and a secondary entrance, itself 250 yard to the east of the main gate. The construction parking lot is located at the perimeter of the site, close to the main entrance gate. A road leads from each entrance gate to the building. Another road surrounds the building and the two administration offices; it intersects with the roads that lead off-site. All laydown areas can be accessed from that road. Across the road from the administration offices are three warehouses. They are aligned, parallel and adjacent to the road.

5.2.2 Purpose and Formalization of the Model

The goal of the system is to answer to questions of different type, such as:

Is there an ADMINISTRATION close to the BUILDING ?
Is there a TOOLTRAILER adjacent to the BUILDING?
Is there a WAREHOUSE close to the ADMINISTRATION?
Where is the ConstructionParkinglot located with respect to the BUILDING?

Is there access from Laydown3 to the MainEntrance?
Are the TOOLTRAILERS and the OFFICETRAILERS grouped?

and an answer could be "yes" or "no", it could specify particular objects or relations that meet what is asked for, and so on.

The words of the language we will use are modeled as concepts which are listed in the Conceptual Catalog, given in Appendix 3 of this paper. In Appendix 4 we added Conceptual Relations, and we complemented our knowledge of the domain of construction site layout with a few Prototypes and Schemata in Appendix 5. Finally, in Appendix 6, we wrote down some of the Inference Rules that will allow us to deduce spatial properties from other ones. For the reader who is eager to see our model in use we refer to Section 6 of this paper.

5.3 Application 2: Designing a Construction Site Layout

5.3.1 Input Knowledge and Design Task

The design of a construction site layout exemplifies one problem in the class of spatial arrangement problems. We will here discuss a blackboard expert system, named SightPlan, that, in its current implementation, lays out the temporary facilities on site of a two-unit coal-fired power plant (Figure 10). The project is named the Intermountain Power Project and is located near Delta in Utah.

SightPlan knows the domain objects it has to locate on site. Binary constraints between objects are also specified. Constraints express geometrical and topological bounds on the spatial location of objects that do not yet have a fixed position on site. Given this information, SightPlan's task consists of generating a two-dimensional site layout.

5.3.2 Type Hierarchy for Construction Site Description

The domain objects that SightPlan has to locate on site could be, *e.g.*, contractor trailers, fabrication yards, and temporary buildings. A type hierarchy of an early prototype of SightPlan is shown in Figure 11. Objects are specified by means of the spatial attributes: geometry (each object is *rectangular*) and dimensions (each rectangle has a fixed *length* and *width*). Constraints are also shown on this type hierarchy.

Furthermore, SightPlan starts off with strategic knowledge and domain-specific knowledge, contained in independent knowledge sources, that permit the program to reason about actions it

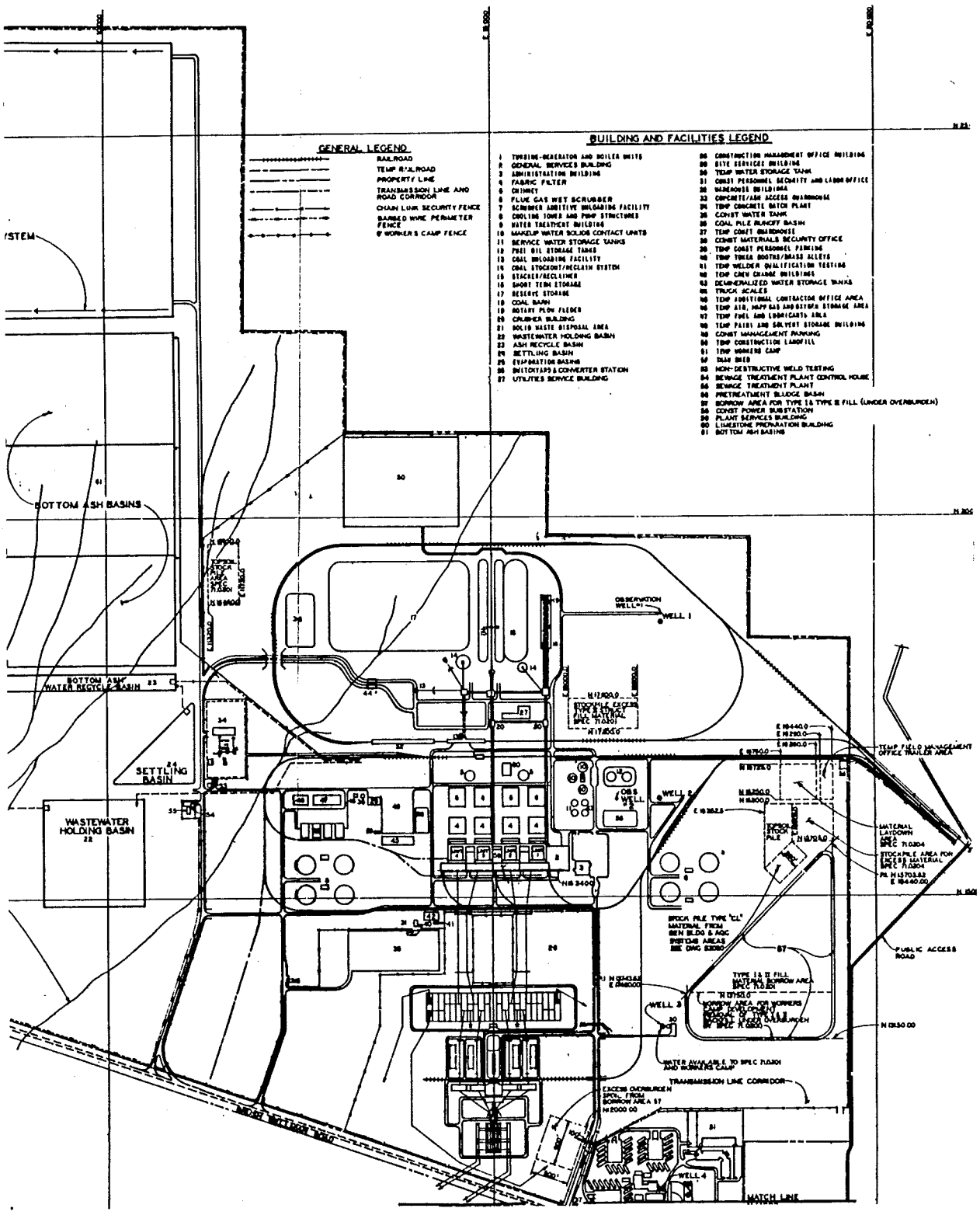


Figure 10: Site Arrangement of the Intermountain Power Project

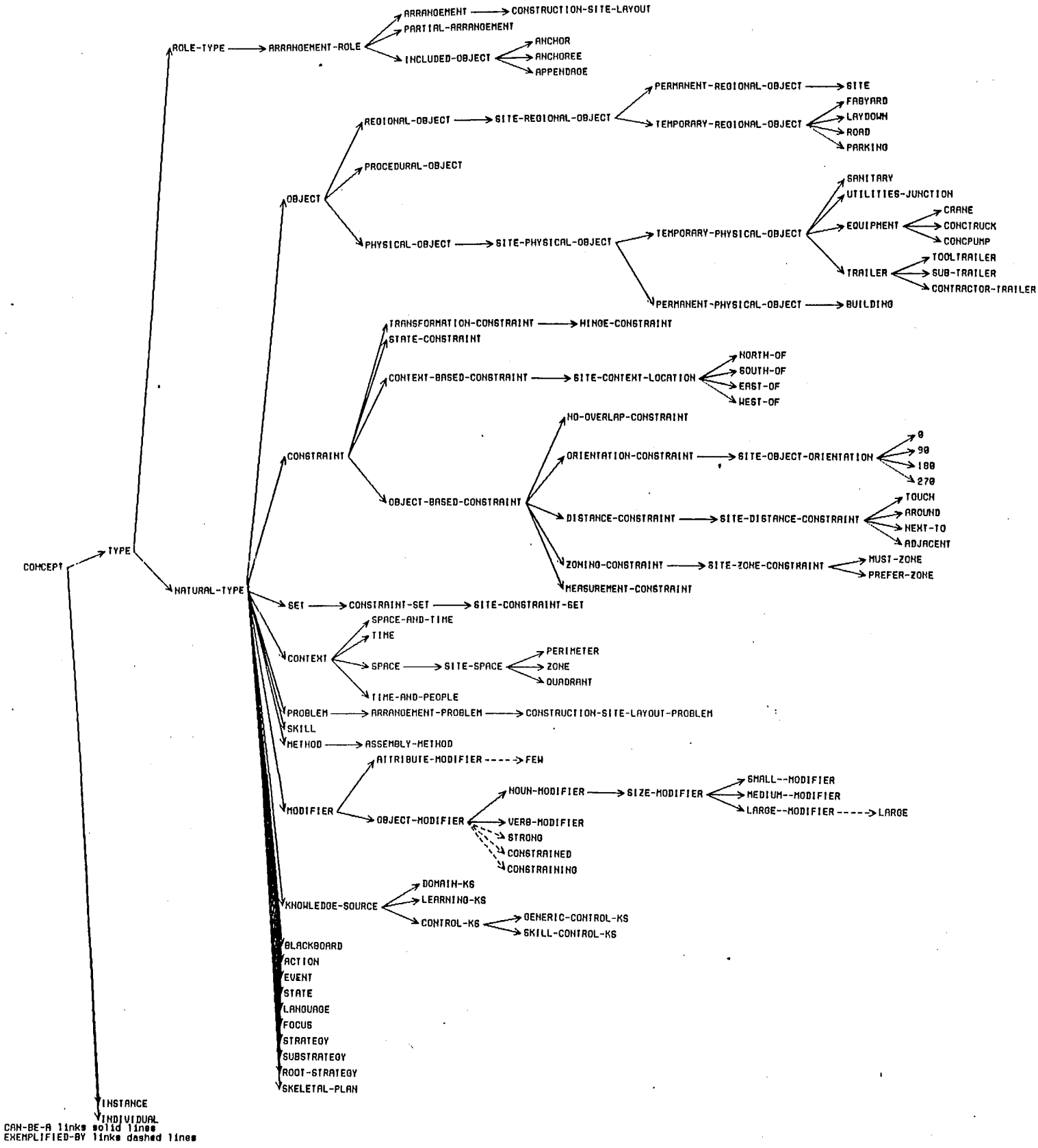


Figure 11: Type Hierarchy in SightPlan

could take to generate the layout. An underlying constraint engine and display engine support the execution of SightPlan's actions. SightPlan is described in Section 7 of this paper.

5.4 Summary

We have thus provided the foundations for our models consisting of a vocabulary of spatial objects and relations. Thanks to the simplifications we made we could easily capture that vocabulary in conceptual graphs. In the following two sections we describe in more detail the two models we here introduced.

6 Inferences from a State Description

In Section 5 we provided a short text describing a layout of some construction site. Reading this description you were probably, at least to some extent, able to create a mental picture of what this layout might look like. You were able do so despite the imprecision with which the location of the objects was specified, by making assumptions on the values of certain missing variables, for instance by guessing or by assuming a typical situation. The text was only a partial description of the real situation, that is, a number of variables were left unspecified. Given that such a text does not contain any contradictory or inconsistent information, it is possible to generate several -- possibly even infinitely many-- configurations that meet the provided description. When asked specific questions about the spatial arrangement of certain objects with respect to one another, answers can sometimes, yet not always be generated.

In this section now, we will show that some types of questions can be answered from the given text by using a type hierarchy for the domain of site layout, and conceptual graphs for the spatial prepositions, extended with schemas, prototypes and inference rules. However, since the given text was under-specified, we will not always be able to generate a precise "yes" or "no": answers may be "it is possible in some instances", or may remain unknown.

6.1 Translation from a State Description in English to Conceptual Graphs

We here repeat the sentences from section 5.2.1 and translate them into conceptual graphs.

The building is located in the middle of the site. It is 100 yard long and 60 yard wide, and its dimensions are small compared to those of the site.

[BUILDING: # *x1] - (1a)
-> (LOC) -> [(λy) [SITE: # *x2] -> (ATTR) -> [MIDDLE: *y]]
-> (ATTR) -> [WIDTH: @60 yard]
-> (ATTR) -> [LENGTH: @100 yard]

[(λy) [*x1] -> (ATTR) -> [DIMENSIONS: *y]] -> (SMALL) (1b)
-> [(λz) [*x2] -> (ATTR) -> [DIMENSIONS: *z]]

There are two administration buildings on site, each of which is <u>adjacent</u> to the building and <u>to the east</u> thereof;

[ADMINISTRATION: Dist { * } @2] (2)
-> (ADJACENT) -> [*x1]
-> (LOC) -> [(λy) [SIDE: East *y] <- (ATTR) <- [*x1]: * s1]

(Note that I could label and enumerate the two administration buildings [ADMINISTRATION: { Administration1, Administration2}], rather than by just saying there are two of them.)

and <u>at the opposite side</u> of the building laydown areas 1, 2, 3, and 4 are <u>grouped</u>.

[LAYDOWN-AREA: { Laydown-1, Laydown-2, Laydown-3, Laydown-4 }] ----- [*w1] (3)

[(λz) [Col *w1] -> (GROUPED) -> [RECTANGLE: *z]] (4)
-> (LOC) -> [(λs2) [*s1] -> (OPPOSITE) -> [(λy) [SIDE: { * } @4 *y] <- (ATTR) <- [*x1]: *s2]]

Since a rectangle is known to have four sides

[RECTANGLE] -> (ATTR) -> [SIDE: { East, South, West, North}]

and it is known which ones are opposite to each other

[(λy) [RECTANGLE] -> (ATTR) -> [SIDE: East]: *y] -> (OPPOSITE)
-> [(λy) [RECTANGLE] -> (ATTR) -> [SIDE: West]: *y]
we could restate (4) as
[(λz) [Col *w1] -> (GROUPED) -> [RECTANGLE: *z]] (4b)
-> (LOC) -> [(λy) [SIDE: West *y] <- (ATTR) <- [*x1]]

Each laydown area is allocated to a single contractor.

(Note that a contractor could have more than one laydown area)

[ALLOCATE] - (5)
-> (PTNT) -> [[(λy) [LAYDOWN-AREA: *y] <- (MEMB) <- [*w1] : ∀]
-> (RCPT) -> [CONTRACTOR: @1]

The respective contractors on site are Joe-the-carpenter, Bill-the-piping-supplier, Ann-the-welder, and Sue-the-concrete-supplier.

[CONTRACTOR: Contractor-1] ----- [CONTRACTOR] -> (ATTR) -> [NAME: Joe] (6)
-> (ATTR) -> [CRAFT: carpenter]
[CONTRACTOR: Contractor-2] ----- [CONTRACTOR] -> (ATTR) -> [NAME: Bill]
-> (ATTR) -> [CRAFT: piping-supplier]

```
[CONTRACTOR: Contractor-3] ----- [CONTRACTOR]    -> (ATTR) -> [NAME: Ann]
                                                 -> (ATTR) -> [CRAFT: welder]
[CONTRACTOR: Contractor-4] ----- [CONTRACTOR]    -> (ATTR) -> [NAME: Sue]
                                                 -> (ATTR) -> [CRAFT: concrete-supplier]

[ALLOCATE]  -> (PTNT) -> [LAYDOWN-AREA: Laydown-1]
            -> (RCPT) -> [CONTRACTOR: Contractor-1
[ALLOCATE]  -> (PTNT) -> [LAYDOWN-AREA: Laydown-2]
            -> (RCPT) -> [CONTRACTOR: Contractor-2]
[ALLOCATE]  -> (PTNT) -> [LAYDOWN-AREA: Laydown-3]
            -> (RCPT) -> [CONTRACTOR: Contractor-3]
[ALLOCATE]  -> (PTNT) -> [LAYDOWN-AREA: Laydown-4]
            -> (RCPT) -> [CONTRACTOR: Contractor-4]
```

Each contractor has his/her own office trailer and tool trailer close to each other on his/her laydown area.

```
[CONTRACTOR: { Contractor-1, Contractor-2, Contractor-3, Contractor-4 } ] ----- [ *w2 ]        (7)
```

Each contractor owns an office trailer and a tool trailer.

```
[ (λ y) [CONTRACTOR: *y ] <- (MEMB) <- [*w2 ]: ∀ ] -> (STAT) -> [ OWN ] -                      (8)
        -> (PTNT) -> [OFFICETRAILER: @1 ]
        -> (PTNT) -> [TOOLTRAILER: @1 ]
```

A contractor's office- and tool trailer are unique (in that they don't belong to another contractor as well).

```
[ [OFFICETRAILER: ¬ ] <- (PTNT) <- [ OWN ] <- (STAT)
                      <- [ (λ y) [CONTRACTOR: *y] <- (MEMB) <- [*w2 ]: Col {*} @2 ] ]        (9)

[ [TOOLTRAILER: ¬ ] <- (PTNT) <- [ OWN ] <- (STAT)
                    <- [ (λ y) [CONTRACTOR: *y] <- (MEMB) <- [*w2 ]: Col {*} @2 ] ]
```

The office trailer and a tool trailer that belong to the same contractor are close together, and are on his laydown area. (note: this is so for every contractor)

```
[ ( λ y ) [CONTRACTOR: *y ] -                                                                  (10)
          <- (MEMB) <- [*w2 ]
          -> (STAT) -> [ OWN ] -> (PTNT) -> [ OFFICETRAILER ] -> (ON) -> [ *p ]
                                                    ↓
                                                (CLOSE-TO)
                                                    ↓
          -> (STAT) -> [ OWN ] -> (PTNT) -> [ TOOLTRAILER ] -> (ON) -> [ *p ]
          <- (RCPT) <- [ALLOCATE] -> (PTNT) -> [LAYDOWN: *p ]          : ∀ ]
```

An outside road runs parallel to the north edge of the site, and is adjacent to the site.

```
[ ROAD: *q1 ] -                                                                                (11)
      -> (PARALLEL) -> [ ( λ y ) [ SIDE: North *y ] <- (ATTR) <- [ *x2 ] ]
      -> (OUTSIDE) -> [ * x2 ]
      -> (ADJACENT) -> [ * x2 ]
```

Along this road are the main entrance gate to the site and a secondary entrance, itself 250 yard to the east of the main gate.

[GATE: MainEntrance] -> (ALONG) -> [*q1] (12)

[GATE: SecondaryEntrance] -> (ALONG) -> [*q1]

[DISTANCE: @250 yard] <- (DISTANCE) -
<-1 [GATE: MainEntrance]
<-2 [GATE: SecondaryEntrance]

[GATE: SecondaryEntrance] -> (EAST-OF) -> [GATE: MainEntrance]

The construction parking lot is located at the perimeter of the site, close to the main entrance gate.

[PARKINGLOT: ConstructionParkinglot]
-> (LOC) -> [(λ y) [*x2] -> (ATTR) -> [PERIMETER: *y]] (13)
-> (CLOSE-TO) -> [GATE: MainEntrance] (14)

A road leads from each entrance gate to the building.

[ROAD: *q2] <- (CONNECT) - (15)
<-1 [GATE: MainEntrance]
<-2 [*x1]

[ROAD: *q3] <- (CONNECT) -
<-1 [GATE: SecondaryEntrance]
<-2 [*x1]

(Note that it is only from the following sentence "... with the roads..." that we know *q3 and *q2 are distinct.)

Another road surrounds the building and the two administration offices; it intersects with the roads that lead off-site.

[SURROUND] (16)
-> (AGNT) -> [ROAD: *q4]
-> (PTNT) -> [OBJECT: { *x1, Administration1, Administration2 }]

[INTERSECT] (17)
-> (AGNT) -> [*q4]
-> (PTNT) -> [ROAD: Dist { *q2, *q3}]

All laydown areas can be accessed from that road.

[ACCESS] (18)
-> (PTNT) -> [[(λ y) [LAYDOWN-AREA: *y] <- (MEMB) <- [*w1] : ∀]
-> (SRCE) -> [*q4]

Across the road from the administration offices are three warehouses.

[*q4] <- (ACROSS) - (19)
<-1 [ADMINISTRATION: Dist { * } @2]
<-2 [WAREHOUSE: Dist { * }@3 *w3]

(Note that I have interpreted this sentence as: "each of the warehouses is across the road from each of the administration offices".)

They are aligned, parallel and adjacent to the road.

[*q3] <- (PARALLEL) <- [(λ y) [LINE: *y] <- (ALIGNED) <- [WAREHOUSE: Col { * }@3 *w3]] (20)

[WAREHOUSE: Dist *w3] -> (ADJACENT) -> [*q3]

6.2 Questions and Answers

The conceptual graphs we defined can now be used to find the answer to questions. The inference mechanism we use consists of the merging of conceptual graphs, applying inference rules, and filling in undetermined variables by using prototypes and schemata.

1 Is the ConstructionParkinglot located on site?

Query: Try to find a match for the following conceptual graph:
[PARKINGLOT: ConstructionParkinglot] -> (LOC) -> [(λ y) [*x2] -> (ATTR) -> [CONCEPT: ? *y]]

Answer: The query matches with line (13)

=> YES [CONCEPT: ? *y] ----- [PERIMETER: *y]

2 Is there a CRANE on site?

Q: [CRANE: ?] -> (LOC) -> [(λ y) [*x2] -> (ATTR) -> [CONCEPT: ? *y]]

A: no match was found

=> NO

3 How many GATES are there?

Q: Search for [GATE: ?] among all the concepts in the text.

A: Results in [GATE: MainEntrance], [GATE: SecondaryEntrance] at line 12

=> TWO GATES, [GATE: MainEntrance] and [GATE: SecondaryEntrance]

4 Is there an ADMINISTRATION close to the BUILDING ?

Q: [ADMINISTRATION: ?] -> (CLOSE-TO) -> [BUILDING: #]

A: 1) find possible instances for ADMINISTRATION => 2 are mentioned in line (2); I will call them Administration1 and Administration2; and for BUILDING => one is mentioned in line (1) => *x1
2) for each combination of the instances try to find a match for the question above:
[ADMINISTRATION: Administration1] -> (CLOSE-TO) -> [BUILDING: #*x1]
[ADMINISTRATION: Administration2] -> (CLOSE-TO) -> [BUILDING: #*x1]
=> no success
3) apply an inference rule (see Appendix 6) to any of the given sentences and which may result in finding the CLOSE-TO relationship:
relation [PHYSOBJ: *x1] -> (ADJACENT) -> [PHYSOBJ: *x2]
implies [*x1] -> (CLOSE-TO) -> [*x2]
So the query now consists of finding a given sentence in which both of the arguments match those generated in 1) but the relationship doesn't => line (2)
[ADMINISTRATION: Dist { * } @2] -> (ADJACENT) -> [*x1]
Apply the rule to it twice, and we find the statement true for both combinations
[ADMINISTRATION: Dist { * } @2] -> (CLOSE-TO) -> [*x1]

=> YES, [ADMINISTRATION: Dist { * } @2] -> (CLOSE-TO) -> [*x1]

5 What are the dimensions of the SITE?

Q: [*x2] -> (ATTR) -> [DIMENSIONS: ?]

A: 1) find line that matches the question: line (1b)
[(λ y) [*x1] -> (ATTR) -> [DIMENSIONS: *y]] -> (SMALL)
-> [(λ z) [*x2] -> (ATTR) -> [DIMENSIONS: *z]]
but we need to interpret (SMALL) and the dimensions of the building in order to obtain a numeric value for the dimensions of *x2
2) infer the dimensions of the building itself by using line (1a) and the equivalence of concepts
[[RECTANGLE]
-> (ATTR) -> [WIDTH: @a]
-> (ATTR) -> [LENGTH: @b]] ----- [[RECTANGLE] -> (ATTR) -> [DIMENSIONS: (@a @b)]]
Thus we find: [*x1] -> (ATTR) -> [DIMENSIONS: (@60 yard @100 yard)]
3) run those dimensions through the comparative measure (SMALL)
relation [PHYSOBJ: *x1] -> (SMALL) -> [PHYSOBJ: *x2]
means [(λ y) [*x1] -> (ATTR) -> [DIMENSIONS: *y]]
≤ 0.3 * [(λ y) [*x2] -> (ATTR) -> [DIMENSIONS: *y]]
i.e., the dimensions of *x1 are at the most 0.3 times the dimensions of *x2
So, 1/0.3 [(λ y) [*x1] -> (ATTR) -> [DIMENSIONS: (@60 yard @100 yard) *y]] gives an estimate of the dimensions of the site: [(λ z) [*x2] -> (ATTR) -> [DIMENSIONS: *z]] => *z = (@200 yard @333 yard)

=> [SITE] -> (ATTR) -> [DIMENSIONS: (@200 yard @333 yard)]

6 Is there a TOOLTRAILER adjacent to the BUILDING?

Q: [TOOLTRAILER : ?] -> (ADJACENT) -> [BUILDING: #]

A: 1) find possible instances for TOOLTRAILER => line (8) mentions that each contractor has exactly one and line (6) mentions there are 4 contractors; I will call them ToolTrailer1 thru 4 ; and for BUILDING => one is mentioned in line (1) => *x1
2) for each combination of the instances try to find a match for the question above:
[TOOLTRAILER : ToolTrailer1] -> (ADJACENT) -> [BUILDING: *x1] etc => no success

3) try to apply an inference rule. We cannot infer adjacency because the distance from a tool trailer to the building is not given => no success

=> UNKNOWN

7 Is there a WAREHOUSE close to the ADMINISTRATION?

Q: [WAREHOUSE: ?] -> (CLOSE-TO) -> [ADMINISTRATION: ?]

A: 1) find possible instances for WAREHOUSE => line (19) mentions there are three of them; I will call them Warehouse1 thru 3 ; and for ADMINISTRATION => line (2) mentions there are two of them; I will call them Administration1 and 2
2) for each combination of the instances try to find a match for the question above [WAREHOUSE : Warehouse1] -> (CLOSE-TO) -> [ADMINISTRATION : Administration1] etc , and stop after the first success => no success
3) apply inference rules from Appendix 6:
relation [PHYSOBJ: *x1] <- (ACROSS) -
<-1 [PHYSOBJ: *x2]
<-2 [PHYSOBJ: *x3]
implies [DISTANCE: *y] <- (DISTANCE) -
<-1 [PHYSOBJ: *x2]
<-2 [PHYSOBJ: *x3]
[(λz) [*x1] -> (ATTR) -> [WIDTH: *z]] ----- [*q]
[*q] ≤ [*y] < (SQRT (* 2 (SQUARE [*q])))
relation [DISTANCE: *w] <- (DISTANCE) -
<-1 [PHYSOBJ: *x2]
<-2 [PHYSOBJ: *x3]
[*w] ≤ 25 yard
implies [PHYSOBJ: *x2] -> (CLOSE-TO) -> [PHYSOBJ: *x3]
So, when we find line (19) to match the first inference rule
[*q4] <- (ACROSS) -
<-1 [ADMINISTRATION: Dist { * } @2]
<-2 [WAREHOUSE: Dist { * }@3 *w3]
and we use default knowledge on the width of road *q4 = 30 feet, we can compute the distance between the two buildings: min 30 feet and max 42 feet which is indeed ≤ 25 yard required to apply the second inference rule, which results in
[ADMINISTRATION: Dist { * } @2] -> (CLOSE-TO) -> [WAREHOUSE: Dist { * }@3 *w3]
So the question can be answered True for each combination of administration and warehouse.

=> YES

8 Where is the ConstructionParkinglot located with respect to the BUILDING?

Q: [PARKINGLOT: ConstructionParkinglot]
-> (LOC) -> [(λy) [BUILDING: # *x1] -> (ATTR) -> [CONCEPT: ? *y]]

A: 1) since a direct match fails, try to find the location of the parking lot with respect to the site =>
line 13 [PARKINGLOT: ConstructionParkinglot] -> (LOC) -> [(λ y) [*x2] -> (ATTR) -> [PERIMETER: *y]]
and we know from
line 1 [BUILDING: # *x1] -> (LOC) -> [(λy) [SITE: # *x2] -> (ATTR) -> [MIDDLE: *y]]
2) we could infer more, because:
line 14 [PARKINGLOT: ConstructionParkinglot] -> (CLOSE-TO) -> [GATE: MainEntrance]
line 12 [GATE: MainEntrance] -> (ALONG) -> [*q1]
by using **relation** [PHYSOBJ: *x1] -> (ALONG) -> [PHYSOBJ: *x2]
implies [PHYSOBJ: *x1] -> (ADJACENT) -> [PHYSOBJ: *x2]

infer from line 12 [GATE: MainEntrance] -> (ADJACENT) -> [*q1]
relation [PHYSOBJ: *x1] -> (ADJACENT) -> [PHYSOBJ: *x2]
and [PHYSOBJ: *x2] -> (CLOSE-TO) -> [PHYSOBJ: *x3]
and [(λ y) [*x2] -> (ATTR) -> [DIMENSIONS: *y]] -> (SMALL)
-> [(λ y) [*x1] -> (ATTR) -> [DIMENSIONS: *y]]
and [(λ y) [*x2] -> (ATTR) -> [DIMENSIONS: *y]] -> (SMALL)
-> [(λ y) [*x3] -> (ATTR) -> [DIMENSIONS: *y]]
implies [PHYSOBJ: *x1] -> (CLOSE-TO) -> [PHYSOBJ: *x3]
new line 12 and line 14, and we know that the dimensions of [GATE: MainEntrance] are small compared to those of the ConstructionParkinglot and the (length of the) Road
[PARKINGLOT: ConstructionParkinglot] -> (CLOSE-TO) -> [*q1]
Furthermore:
line 11 [ROAD: *q1] -> (PARALLEL) -> [(λ y) [SIDE: North *y] <- (ATTR) <- [*x2]]
Infer [PARKINGLOT: ConstructionParkinglot] -> (CLOSE-TO) -> [(λ y) [SIDE: North *y] <- (ATTR) <- [*x2]]
and line 1 [BUILDING: # *x1] -> (LOC) -> [(λ y) [SITE: # *x2] -> (ATTR) -> [MIDDLE: *y]]
so we can conclude that [PARKINGLOT: ConstructionParkinglot] -> (NORTH-OF) -> [BUILDING: # *x1]

=> [PARKINGLOT: ConstructionParkinglot] -> (NORTH-OF) -> [BUILDING: # *x1]

9 Is there access from Laydown3 to the MainEntrance?

Q: [ROAD: ?] <- (CONNECT) -
<-1 [LAYDOWN-AREA: Laydown3]
<-2 [GATE: MainEntrance]

A: 1) instantiate line 18 for Laydown3:
[ACCESS] -> (PTNT) -> [LAYDOWN-AREA: Laydown3]
-> (SRCE) -> [*q4]
infer [*q4] <- (CONNECT) -
<-1 [LAYDOWN-AREA: Laydown3]
<-2 [PHYSOBJ: ?]
specify line 17 for *q2 only: [INTERSECT] -> (AGNT) -> [*q4]
-> (PTNT) -> [*q2]
infer [*q4] <- (CONNECT) -
<-1 [LAYDOWN-AREA: Laydown3]
<-2 [PHYSOBJ: ?] 1->
[PHYSOBJ: ?] 2->
- (CONNECT) -> [*q2]
and line 15 [ROAD: *q2] <- (CONNECT) -
<-1 [GATE: MainEntrance]
<-2 [*x1]
Finally, apply the transitivity of CONNECT to find the connection between [LAYDOWN-AREA: Laydown3] and [GATE: MainEntrance]

=> YES, there is a connection from Laydown3 to the MainEntrance

10 Are the TOOLTRAILERS and the OFFICETRAILERS grouped?

Q: [OFFICETRAILER: Col { * } @4] -> (GROUPED) -> [RECTANGLE: ? *z]
[TOOLTRAILER : Col { * } @4] -> (GROUPED) -> [*z]

A: 1) find possible instances for TOOLTRAILER => line (8) mentions that each contractor has exactly one and line (6) mentions there are 4 contractors; I will call them ToolTrailer1 thru 4 ; similarly for OFFICETRAILER; call them Officetrailer1 thru 4;
2) line 10 states that for each contractor's laydown area, his office and tooltrailer are on it:

[OFFICETRAILER] -> (ON) -> [LAYDOWN: *p]
|
(CLOSE-TO)
|
[TOOLTRAILER] -> (ON) -> [*p]
Furthermore in line 4, all laydown areas are grouped into a bigger rectangle:
[LAYDOWN-AREA: Col { * } @4] -> (GROUPED) -> [RECTANGLE: *z]
Assuming now that if units are grouped, the parts of those units are grouped as well (this may be too strong an assumption), we may conclude that
[OFFICETRAILER: Col { * } @4] -> (GROUPED) -> [RECTANGLE: *z]
[TOOLTRAILER : Col { * } @4] -> (GROUPED) -> [*z]

=> **YES**, they are grouped

6.3 Conclusions

We have demonstrated that conceptual graphs can be used to clearly state the information provided by a description in English. By making use of a substantial amount of additional domain knowledge captured in prototypes, conceptual relations and inference rules we were most of the time capable of constructing answers to a variety of types of questions.

The approach that we used here is straight forward, but we feel we were only moderately successful in our task: at moments we had to carefully mold the input sentences so that inference rules could apply to them. Our problems may be due to a lack of rigor in the formalization of conceptual relations to represent space and the accompanying inference rules (which one might improve), but worse, as in the modeling of any recoding problem we had to face the incompleteness of the input information which prevented us sometimes from concluding additional statements, that in fact are not implied, but which are obvious at first sight.

7 The SightPlan System

7.1 SightPlan's Design Methodology

SightPlan uses the assembly method to generate its solution: the system incrementally selects and assembles site objects into partial layouts. Objects can initially be anywhere in such a layout but their legal positions are reduced by successively applying constraints. At well-chosen times partial layouts are combined to form the complete arrangement.

SightPlan reasons about its problem-solving actions. At each step, it opportunistically selects what the best action is to take next. The system is equipped both with strategic knowledge on how to go about generating a spatial arrangement, and with specific domain knowledge on how to lay out a construction site. For instance, SightPlan first chooses to create a spatial arrangement

bounded by the edges of the site. Then, it includes objects: all of those that have a known location on site --permanent buildings, roads, fencing around the site--, and some of those that need to be located on site during problem solving. SightPlan chooses one object in the arrangement to be the "anchor", and then locates one of the unpositioned objects by satisfying constraints between it and this "anchor". Next, another unpositioned object is selected, and its constraints either with the anchor, with a fixed object, or with a previously located object are met, and so on.

The strategy is implemented by foci which describe control sentences that contain nouns (= types of objects or constraints in the layout) and modifiers (= adjectives that provide a measure for comparing objects). Actions that match such a control sentence well are prioritized for execution over those that don't match as well (Figure 12).

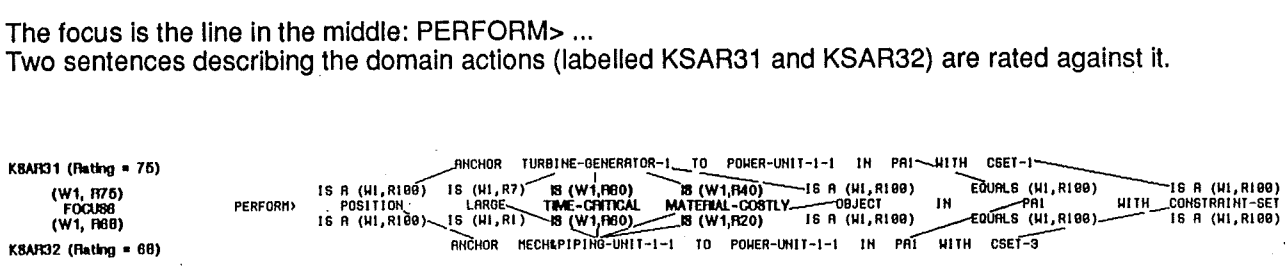


Figure 12: Matching a Domain Action against the Focus of a Strategy

Upon execution of a domain action SightPlan looks up the procedural definition of the constraint and applies it to its arguments. As a result of meeting the constraint the call to the underlying constraint engine returns a reduced set of possible locations for each argument. SightPlan then calls a display engine and depicts the resulting locations to the user. Pursuing this method, the system generates positions for objects where they satisfy the constraints. At the end of the SightPlan run, when all objects have been located and all constraints have been applied, it is possible that objects still don't have unique allowable positions. In that case, SightPlan may sample one or more legal locations from the allowable positions of one object, then verify the constraints between it and other objects, and proceed in this manner until a single layout is found. (We assume that the solution space is underconstrained).

7.2 Applying Modifiers to Strategically Rank Objects

Modifiers allow the control system to differentiate between objects and to rank objects, so that one can strategically specify which of a given type should be preferred over others. Modifiers measure numerically how well one or more attributes of an object compare to those of a selected

prototype. For instance, SightPlan can choose to place "large" objects first. Objects are ranked by measuring their "large"-ness, by computing the area given the dimensions. Based on this each object is then compared to the "large"-ness of a reference object --the site in this case (Figure 13).

```
TURBINE-GENERATOR-1                MECH&PIPING-UNIT-1-1
Attributes:                        Attributes:
        Dimensions: (700 800)              Dimensions: (500 200)
        in feet                            in feet

=> LARGE-RATING:  7                => LARGE-RATING:  1
```

Figure 13: Applying Modifiers to Rank Objects

In subsection 3.1.1 we pointed out that nouns and adjectives are typically used to describe an object's dimensions and shape, so these types of words have the potential to be modifiers. In SightPlan only adjectives are modifiers, and nouns represent objects that need to be located on site.

7.3 Mapping from Conceptual Graphs to Lisp Functions to Represent Constraints

Constraints express spatial relationships between objects that must hold when they are being located. These relationships involve the distance between edges and the angles between main axes of the objects. In the first implementation of SightPlan we decided that sufficient expressiveness would be provided by the following vocabulary for binary constraints: "closer-than", "further-than", "parallel", "perpendicular", "adjacent", "North", "South", "East", "West", "above", "below", "right", "left", "alongside", "ashortside", "no-overlap", "zoned-in", and "zoned-outside". All locatives are thus translated by the knowledge engineer into hard constraints (Figure 14): *e.g.*, the turbine-generator contractor must be closer than 50 yard from the power-unit (We don't know yet how to deal with constraints such as "as close as possible").

```
[ DISTANCE: @50 yard ] <-  (CLOSER-THAN) -
                              <-1 [TURBINE-GENERATOR]
                              <-2 [POWER-UNIT]
```

Figure 14: Conceptual Graph for a Constraint in SightPlan

To use such conceptual graphs in our reasoning system, we provided a procedural attachment to an underlying constraint engine that generates positions. For more information on the constraint engine, named GS2D, and its low-level geometrical primitives (max-distance, min-distance and space-reduction) see [Confrey 88]. The mapping from the conceptual graph of Figure 14 to the

constraints in Figures 15 and 16 should be obvious. (We chose the language LISP to implement these functions because it is the underlying language of the BB* environment. See Section 7.5)

```
(def-bb1-object CLOSER-THAN

  :level SITE.CONSTRAINTS

  :ATTRIBUTES

  ((FUNCTION-DEFINITION
    (FUNCTION

     (LAMBDA (OBJECT1 OBJECT2 CONSTRAINT)

      (LET ((ESS-AREA-1 ($VALUE ($VALUE OBJECT1 'NEWEST-STATE-FAMILY) 'ESSENTIAL-AREA))
            (ESS-AREA-2 ($VALUE ($VALUE OBJECT2 'NEWEST-STATE-FAMILY) 'ESSENTIAL-AREA)))

     (GS2D-MAX-DISTANCE ESS-AREA-1 ESS-AREA-2 (FLOAT ($VALUE CONSTRAINT 'DISTANCE)))))))

   (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 given distance.  This constraint is symmetrical
                 in the arguments object1 and object2."))

  :LINKS

  ((IS-A (SITE.CONSTRAINTS.SITE-DISTANCE-CONSTRAINT))
   (EXEMPLIFIED-BY (INTERMOUNTAIN.CONSTRAINTS.CLOSER-THAN-39-42
                    INTERMOUNTAIN.CONSTRAINTS.CLOSER-THAN-39-41
                    INTERMOUNTAIN.CONSTRAINTS.CLOSER-THAN-C-A))))
```

Figure 15: Definition of the Generic Closer-Than Constraint

```
(def-bb1-object CLOSER-THAN-C-A

  :level INTERMOUNTAIN.CONSTRAINTS

  :ATTRIBUTES

  ((DISTANCE 50)
   (ARG1 INTERMOUNTAIN.CONSTRUCTION-FACILITIES.TURBINE-GENERATOR-1)
   (ARG2 INTERMOUNTAIN.POWER-PLANT-FACILITIES.POWER-UNIT-1))

  :LINKS

  ((INVOLVES (INTERMOUNTAIN.POWER-PLANT-FACILITIES.POWER-UNIT-1
              INTERMOUNTAIN.CONSTRUCTION-FACILITIES.TURBINE-GENERATOR-1))
   (EXEMPLIFIES (SITE.CONSTRAINTS.CLOSER-THAN))))
```

Figure 16: Example of the Closer-Than Constraint

Inheritance is used in the type hierarchy to inherit the function definition from the level at which the generic constraint is defined to the level at which an example of the constraint exists. It is with the arguments defined on the example that the attached procedure is called.

In subsection 3.1.2 we showed how prepositions and nouns could describe relations between objects. In SightPlan only prepositions are used to label constraints.

7.4 A Very Brief Example

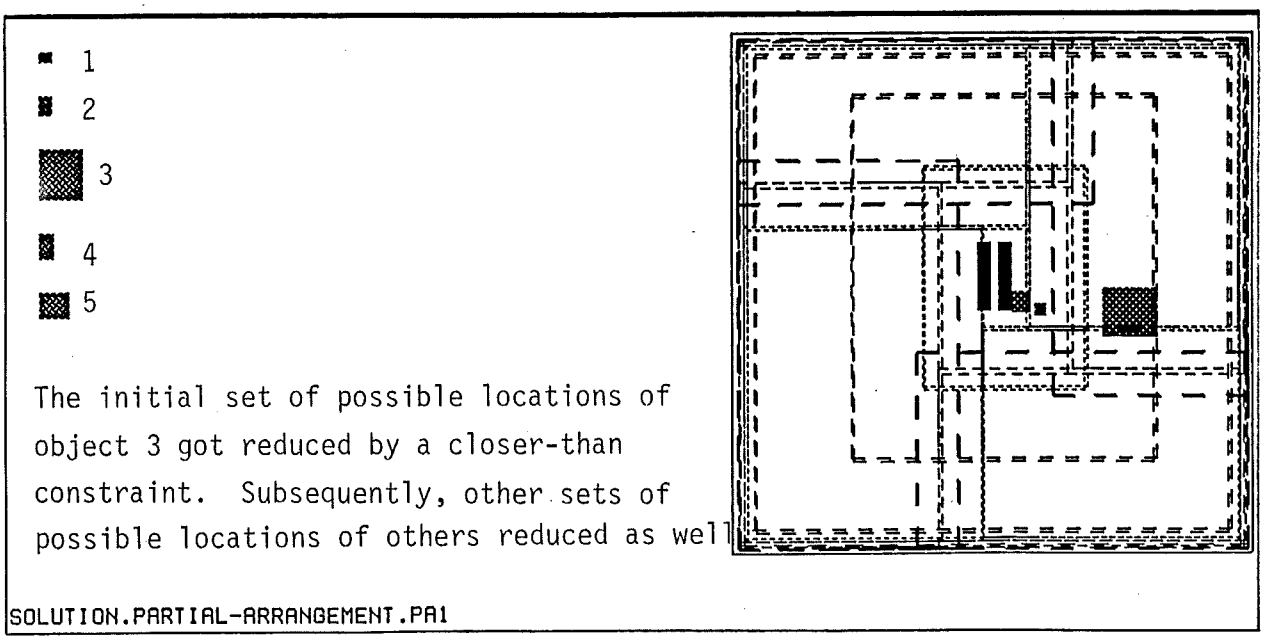


Figure 17: SightPlan during Problem-Solving under a Least-Commitment Strategy

Figure 17 displays how the initial state families of objects included in the partial arrangement that encompasses the entire site are reduced by consecutively applying constraints. It is beyond the scope of this paper to provide full detail on SightPlan however, and we refer the interested reader to [Tommelein 89] for more information.

7.5 SightPlan's System Description

SightPlan is implemented in the BB* layered architecture. BB* consists of BB1, frameworks, and application systems. BB1 is a domain-independent architecture for knowledge-based systems, based on the blackboard model. A framework provides a task-specific problem solving language; in particular, the ACCORD framework provides a language for solving arrangement problems by the assembly method; the DECIDE framework provides a language for expressing control actions. The application system described here, with strategic and domain knowledge for construction site layout, is research work in progress [Tommelein 89]. More information about the BB* layered environment can be found in [Hayes-RothB 85, 86], and prototypes for SightPlan are described in [Tommelein 87a, 87b].

8 Conclusions and Possible Extensions

While doing the research work we discussed here, we have discovered an abundance in spoken language of nouns, adjectives, verbs, and prepositions, and in pictorial language of icons, symbols, gestures, caricatures, projection mechanisms, drawing conventions, and so on, to express spatial relations. We surveyed models from other researchers on spatial reasoning and elaborated on the abstraction mechanisms at work in the English language to discover that a necessary and at the same time inevitable effect of any coding process is the loss of information.

Abstractions must be tailored to suit the needs of a particular model, and yet, they must retain sufficient information to remain clear and flexible for application in a context that encompasses more than what is needed for the one-time use of the model. The framework we chose for the conceptualization of space is formalized by conceptual structures, which have the advantage of being very general and broad. Thanks to this it is easy to maintain consistency among different models.

We showed how conceptual structures could represent spatial predicates, and we applied them in two ways. First, we performed queries by using prototypical concepts and inference rules on a set of sentences in order to answer questions about a particular site description. Second, we mapped conceptual graphs to spatial functions, called their underlying procedures, and thus satisfied constraints on given objects in order to generate a layout design. The results of both our models were satisfying. The problems we ran into were only partially due to a lack of rigor in the formalization of our spatial relations; most problems stemmed from the incompleteness of the input information and arose at the moment our models' processes needed to recode or decode such an input. The latter are tough problems that require further investigation by researchers in linguistics, in artificial intelligence, in robotics, in cognitive science, in computer science, and by others.

Extensions to our models can be made in several ways. First, the vocabulary needs to become richer, for instance to deal with objects in three-dimensional space, objects that are time-dependent, objects of various shape, and so on. Second, different abstraction levels need to be introduced so that objects can be observed from multiple perspectives, and such abstraction levels must be integrated so that an object's representation can easily alter based on changing needs. Third, results from work on visual mental imagery might suggest that other representations are needed to complement the abstract propositions that we have introduced here. From our experience we conclude however that the formalization of concepts into conceptual structures provides a solid basis for additional research in this field.

9 Bibliography

[Allen 83] Allen, J.F.: **Maintaining Knowledge about Temporal Intervals**, *Communications of the ACM*, Vol. 26, No. 11, pp. 832-843, November 1983

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

[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

[Baykan 87] Baykan, C.A., Fox, M.S.: **An Investigation of Opportunistic Constraint Satisfaction in Space Planning**, *Proceedings of the 10th International Joint Conference on Artificial Intelligence*, IJCAI 87, Milan, Italy, pp. 1035-1038, 23-28 August 1987

[Bool 81] Bool, F.H., et al.: ***M.C. Escher - His Life and Complete Graphic Work***, Locher, J.L. (editor), Harry N. Abrams, Inc, NY, 349 pages, 198 (figure from p. 152)

[Confrey 88] Confrey, T., Daube, F.: ***GS2D Constraint Engine***, Knowledge Systems Laboratory, Computer Science Department, Stanford University, Report No. KSL-88-15, March 1988

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

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

[Flemming 86] 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

[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

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

[Hayes-RothB 86] Hayes-Roth, B., Garvey, A., Johnson, M.V.Jr., Hewett, M.: ***A Layered Environment for Reasoning about Action***, Knowledge Systems Laboratory, Computer Science Department, Stanford University, Report No. KSL-86-38, August 1986

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

[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

[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

[Hofstadter 80] Hofstadter, D.R.: ***Gödel, Escher, Bach: An Eternal Golden Braid***, Vintage Books, 777 pages, 1980 (figure from p. 480)

[Kahn 77] Kahn, K., Gorry, G.A.: **Mechanizing Temporal Knowledge**, *Artificial Intelligence*, Vol. 9, pp. 87-107, 1977

[Kautz 85] Kautz, H.A.: **Formalizing Spatial Concepts and Spatial Language**, Chapter 2 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

[Kosslyn 77] Kosslyn, S.M., Shwartz, S.P.: **A Simulation of Visual Imagery**, *Cognitive Science*, Vol. 9, pp. 341-295, 1977

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

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

[Longman 78] Longman Group, Ltd.: ***Longman Dictionary of Contemporary English***, Longman, London, 1978, on-line computer version available from CSLI at Stanford University

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

[Moore 76] Moore, G.T., Golledge, R.G. (editors): ***Environmental Knowing: Theories, Research and Methods,*** Stroudsburg, Pa, Dowden, Hutchinson & Ross, 1976

[Pick 83] Pick, H., Acredolo, L. (eds.): ***Spatial Orientation: Theory, Research and Application***, Plenum Press, 1983

[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

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

[Rodale 78] Rodale, J.I.: ***The Synonym Finder***, Urdang, L. and LaRoche, N. (editors), Warner Books, 1978

[Schubert 83] Schubert, L.K., Papalaskaris, M.A., Taugher, J.: **Determining Type, Part, Color and Time Relationships**, *Computer*, pp. 53-60, October 1983

[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

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

[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

[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

[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

[Tommelein 87a] 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 87b] 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, 1987

[Tommelein 89] Tommelein, I.D.: PhD Dissertation, Department of Civil Engineering, Stanford University, in preparation, 1989

abut on
according as
adjacent
adjoin
after
along
along
anyhow
applicable
apply
bail out
b and b
before
beside
bet
better
big
blot
blow
boost
bridge over
bring down
but
carry
carry forward
certain
chum
clear
clinker-built
close out
coarse
come
come in
come off
come on
come out
come out with
complain
conjecture
contiguous
continuous
correlative
curtail
deal
delay
develop
direct
disorientate
do
drop
due
dull
employ
except
except for
exhaust
expire
extend
extraordinary
follow

Appendix 1

Search in the Longman Dictionary

for
foresee
forward
friendly
further
further to
future
gain
get back
give
go in for
go out
grant
guess
guy
hammer out
hand
have in
hinge on
immediate
incidentally
judge
located
melt
modern
morally
move in
neighbour on, AmE -bor on
next
next
next-door
next door
next to
next to
nip at
no
note
now
nuts
off
other
pass on
pay
payable
possible
present
promise
prompt
propose
proximal
proximo
pto
pulse
put back
put in
put off
put on
put out
put up
qualify
regret
resolve
right
round up
run
run out
second-degree
send on
set
shall
sine die
skip
slate
sleep on
something
squash
stay on
string along
succeed
tell
think
tide over
tip
treat to
twang
unthinkable
up
vice-
vote
wed
what
whatever
when
when
will

Appendix 2 - Vocabulary of Spatial Terms

A above, about, abut, access, accumulate, across, adapt, adjacent, adjoin, adjunct, adjust, after, agglomerate, akin, align, almost, along, alongside, amass, among, analogous, anchor, (at an) angle, anywhere, apart, append, area, (in the same) area, around, arrange, assemble, at, attach, attenuate, average, away

B back, balanced, before, begin, behind, below, bend, eside, between, big, bilateral, border, bottom, bound, box, bracket, broad, buckle

C categorize, center, chain, circle, circuitus, (at the) circumference, circumfluent, circumscribe, close, clump, cluster, cocear, coextensive, cojoining, collateral, colossal, combine, commensurate, compact, comparable, complementary, comprise, concatenate, concave, concentric, concurrent, condens, cone, configure, confine, conglomerate, connect, consecutive, consolidate, constrict, contactual, continuous, convex, couple, cover, crooked, cube, curb

D demarcate, direct, distant, distinct, distribute, divide, down, during

E east, encircle, enclosed, encompass, end, enfold, enormous, enwrap, equal, equidistant, expansive, extensive

F face, far, fasten, fit, fix, following, form, format, frame, (in) front (of)

G gigantic, great, group

H harmonize, heap, hexaeder, high, hitch, hollow, huge

I immense, impose, in, inadequate, inch, include, indent, infinite, inside, insufficient, integrate, intermediate, intermittent, intersect, interval

J join, just, juxtaposed

K knot

L later, large, laydown, layout, lean, left, leftmost, length, lie, like, limit, line-up, link, little, localize, locate, long, low

M matching, mean, measure, medium, meter, merge, middle, moderate, moment, monumental, moore, move

N narrow, near, next (to), normal, north

O octaeder, on, on all sides, order, orient, orientation, organize, out, outside, oval, over, overlapping

P parallel, pattern, perpendicular, place, peripheral, pile, pose, position, precede, prescribe, proportional, proximal, put, put into

Q

R rank, rear, rectangular, relate, restrict, right, roomy, round

S same, separate, set, shape, short, side, (on all) sides, simultaneous, situate, small, smooth, (at a) spacing, spacious, spread out, stack, stage, stagger, straight, stupendous, suit, surround, south, square

T tall, tangential, taper, tie, tiny, top, touch, tower, translate, triangle, turn

U unbounded, under, underneath, up, upon

V vast, volume

W west, while, width, wide, wrap (around)

X

Y yard, yoke

Z

Appendix 3 - Conceptual Catalog

ACCESS < VERB

ADMINISTRATION < BUILDING

AREA < ATTRIBUTE

BUILDING < PHYSOBJ

CONCEPT < T

CONTRACTOR < PERSON < ANIMAL < PHYSOBJ
Joel is the concrete supplier.
[CONTRACTOR: Joel] -> (ATTR) -> [CRAFT: ConcreteSupplier]

CRAFT < ATTRIBUTE

CRANE < EQUIPMENT < MOBILE-ENTITY

DIMENSIONS < ATTRIBUTE
[RECT]
-> (ATTR) -> [WIDTH: @a]
-> (ATTR) -> [LENGTH: @b] —— [RECT] -> (ATTR) -> [DIMENSIONS: (@a @b)]

DISTANCE < MEASURE < CONCEPT

EQUIPMENT < MOBILE-ENTITY

GATE < PHYSOBJ

INTERSECT < VERB

LAYDOWN-AREA < OBJECT

LENGTH < ATTRIBUTE
Length is an attribute of a geometrical object such as a rectangle.
[LENGTH] <- (ATTR) <- [RECTANGLE]

MIDDLE < ATTRIBUTE
A building is located in the middle of the site.
[BUILDING] -> (LOC) -> [(λy) [SITE: #] -> (ATTR) -> [MIDDLE: *y]]

MOBILE-ENTITY < PHYSOBJ

NAME < ATTRIBUTE

OBJECT < CONCEPT

OFFICETRAILER < TRAILER

PARKINGLOT < OBJECT

PERIMETER < ATTRIBUTE

PHYSOBJ < OBJECT

RECTANGLE < SHAPE
[RECTANGLE]
-> (DIMENSIONS) -> [DIMENSIONS: (@a @b)]
-> (ATTR) -> [AREA: (* @a @b)]
-> (ATTR) -> [SIDE: { East, South, West, North }]
-> (ATTR) -> [PERIMETER]
-> (ATTR) -> [MIDDLE]

Note that I used algebraic notations for all numeric computations although I could have spelled them out as a conceptual graph as well:
(* @a @b) ----- [NUMBER] <- (RSLT) <- [TIMES] -> (ARG)
1-> [NUMBER: @a]
2-> [NUMBER: @b]

ROAD < OBJECT

SHAPE < CONCEPT

SIDE < ATTRIBUTE

SITE < OBJECT

SURROUND < VERB

TOOLTRAILER < TRAILER

TRAILER < PHYSOBJ

VERB < CONCEPT

WAREHOUSE < BUILDING

WIDTH < ATTRIBUTE
Width is an attribute of a geometrical object such as a rectangle.
[WIDTH] <- (ATTR) <- [RECTANGLE]

Appendix 4 - Conceptual Spatial Relations

relation ACROSS is not reflexive, is symmetrical, but not transitive; binary relation
Across the road from the city hall is the store.
[ROAD] <- (ACROSS) -
<-1 [BUILDING: City Hall #]
<-2 [BUILDING: Store #]

relation ADJACENT is not reflexive, is symmetrical, but not transitive; binary relation
The house is adjacent to the store.
[HOUSE: #] -> (ADJACENT) -> [STORE: #]
because of symmetry, also: [HOUSE: #] <- (ADJACENT) <- [STORE: #]

relation ALIGNED n-ary relation
The trees are aligned.
[LINE] <- (ALIGNED) <- [TREE: { * } #]

relation ALONG is not reflexive, is not symmetrical, and not transitive; binary relation
The gate is along the road.
[GATE: #] -> (ALONG) -> [ROAD: #]

relation BETWEEN ternary relation
The truck is parked between the crane and the trailer.
[TRUCK: #] <- (BETWEEN) -
<-1 [CRANE: #]
<-2 [TRAILER: #]

relation CLOSE-TO is not reflexive, is symmetrical, and not transitive; binary relation
The parking lot is close to the gate.
[PARKINGLOT: #] -> (CLOSE-TO) -> [GATE: #]

relation CONNECT ternary relation
A road leads from the main entrance gate to the building.
[ROAD] <- (CONNECT) -
<-1 [GATE: MainEntrance]
<-2 [BUILDING: #]

relation DISTANCE ternary relation
The distance between the main entrance gate and the secondary entrance gate is 250 yard.
[DISTANCE: @250 yard] <- (DISTANCE) -
<-1 [GATE: MainEntrance]
<-2 [GATE: SecondaryEntrance]
The relation DISTANCE operates like a function would: it takes two objects as argument and computes the distance between them. The output is a numeric value for the distance.

relation EAST-OF not reflexive, not symmetrical, but transitive; binary relation
The secondary entrance gate is to the Eastof the main entrance gate.
[GATE: SecondaryEntrance] -> (EAST-OF) -> [GATE: MainEntrance]

similarly: relation NORTH-OF, SOUTH-OF, WEST-OF

relation GROUPED n-ary relation
The four laydown areas are grouped.
[LAYDOWN-AREA: { Laydown-1, Laydown-2, Laydown-3, Laydown-4 }] -> (GROUPED) -> [RECTANGLE]

relation INSIDE not reflexive, not symmetrical, but transitive; binary relation
Sue is inside the building.
[PERSON: Sue] -> (INSIDE) -> [BUILDING: #]

relation LARGE not reflexive, not symmetrical, but transitive; binary relation
[PHYSOBJ: *x1] -> (LARGE) -> [PHYSOBJ: *x2]
means [(λ y) [*x1] -> (ATTR) -> [DIMENSIONS: *y]]
> 0.7 * [(λ y) [*x2] -> (ATTR) -> [DIMENSIONS: *y]]

relation LOC binary relation
The building is located in the middle of the site.
[BUILDING: #] -> (LOC) -> [(λ y) [SITE: #] -> (ATTR) -> [MIDDLE: *y]]

relation LONG not reflexive, not symmetrical, but transitive; binary relation
[PHYSOBJ: *x1] -> (LONG) -> [PHYSOBJ: *x2]
means [(λ y) [*x1] -> (ATTR) -> [LENGTH: *y]]
> 0.7 * [(λ y) [*x2] -> (ATTR) -> [LENGTH: *y]]

relation MEDIUM not reflexive, not symmetrical, and not transitive; binary relation
[PHYSOBJ: *x1] -> (MEDIUM) -> [PHYSOBJ: *x2]
means [(λ y) [*x1] -> (ATTR) -> [DIMENSIONS: *y]]
> 0.3 * [(λ y) [*x2] -> (ATTR) -> [DIMENSIONS: *y]]
and [(λ y) [*x1] -> (ATTR) -> [DIMENSIONS: *y]]
≤ 0.7 * [(λ y) [*x2] -> (ATTR) -> [DIMENSIONS: *y]]

relation MEMB binary relation
[PERSON: Paul] <- (MEMB) <- [PERSON: { Ann, Paul, John }]

relation NEAR same meaning as CLOSE-TO

relation OPPOSITE not reflexive, symmetrical, but not transitive
The East and the West side of a rectangle are opposite to each other.
[(λ y) [RECTANGLE] -> (ATTR) -> [SIDE: East]: *y] -> (OPPOSITE)
-> [(λ y) [RECTANGLE] -> (ATTR) -> [SIDE: West]: *y]
alternatively [(λ y) [RECTANGLE] -> (ATTR) -> [SIDE: East]: *y] <- (OPPOSITE)
<- [(λ y) [RECTANGLE] -> (ATTR) -> [SIDE: West]: *y]

relation OUTSIDE not reflexive, not symmetrical, but transitive; binary relation
John is outside the room.
[PERSON: John] -> (OUTSIDE) -> [ROOM: #]

relation PARALLEL reflexive, symmetrical, and transitive; binary relation
The path runs parallel to the road.
[PATH: #] -> (PARALLEL) -> [ROAD: #]

relation SMALL not reflexive, not symmetrical, but transitive; binary relation
[PHYSOBJ: *x1] -> (SMALL) -> [PHYSOBJ: *x2]
means [(λ y) [*x1] -> (ATTR) -> [DIMENSIONS: *y]]
≤ 0.3 * [(λ y) [*x2] -> (ATTR) -> [DIMENSIONS: *y]]
The advantage of defining words that specify a magnitude as a relation is that the relation can compare any two concepts of the same dimensions with one another. By default, an abject can be cmpared with a prototypical one of its type.
The dimensions of the bike are small compared to those of the house.
[(λ y) [BIKE: #] -> (ATTR) -> [DIMENSIONS: *y]] -> (SMALL)
-> [(λ z) [HOUSE: #] -> (ATTR) -> [DIMENSIONS: *z]]

Appendix 5 - Prototypes and Schemata

prototype of ROAD is

```
[ ROAD ] -
        <- (CONNECT) -
                        <-1 [ LOCATION: *x ]
                        <-2 [ LOCATION: *y ]
        -> (ATTR) -> [ LENGTH: @700yard ]
        -> (ATTR) -> [ WIDTH: @30feet ]
        -> (ATTR) -> [LANE: { * } @2 ]
```

A road connects two locations. It provides space for objects to move on, so we can abstract it as a rectangular object with a width and a length. Assume that the width of the road is constant over its length. The width is dependent on the type of road. This prototype represents a typical road on the site of a powerplant: it is a 2-lane road, usually 30 ft wide. Roads can be short or long. Since we consider a road on the site, we can assume a road typically is a fraction of the site dimensions, say between intersection a road may stretch over about 1/4 of the site, or approximately 700 yard.

prototype of TRAILER is

```
[ TRAILER ] -
        -> (ATTR) -> [ LENGTH: @60feet ]
        -> (ATTR) -> [ WIDTH: @12feet ]
```

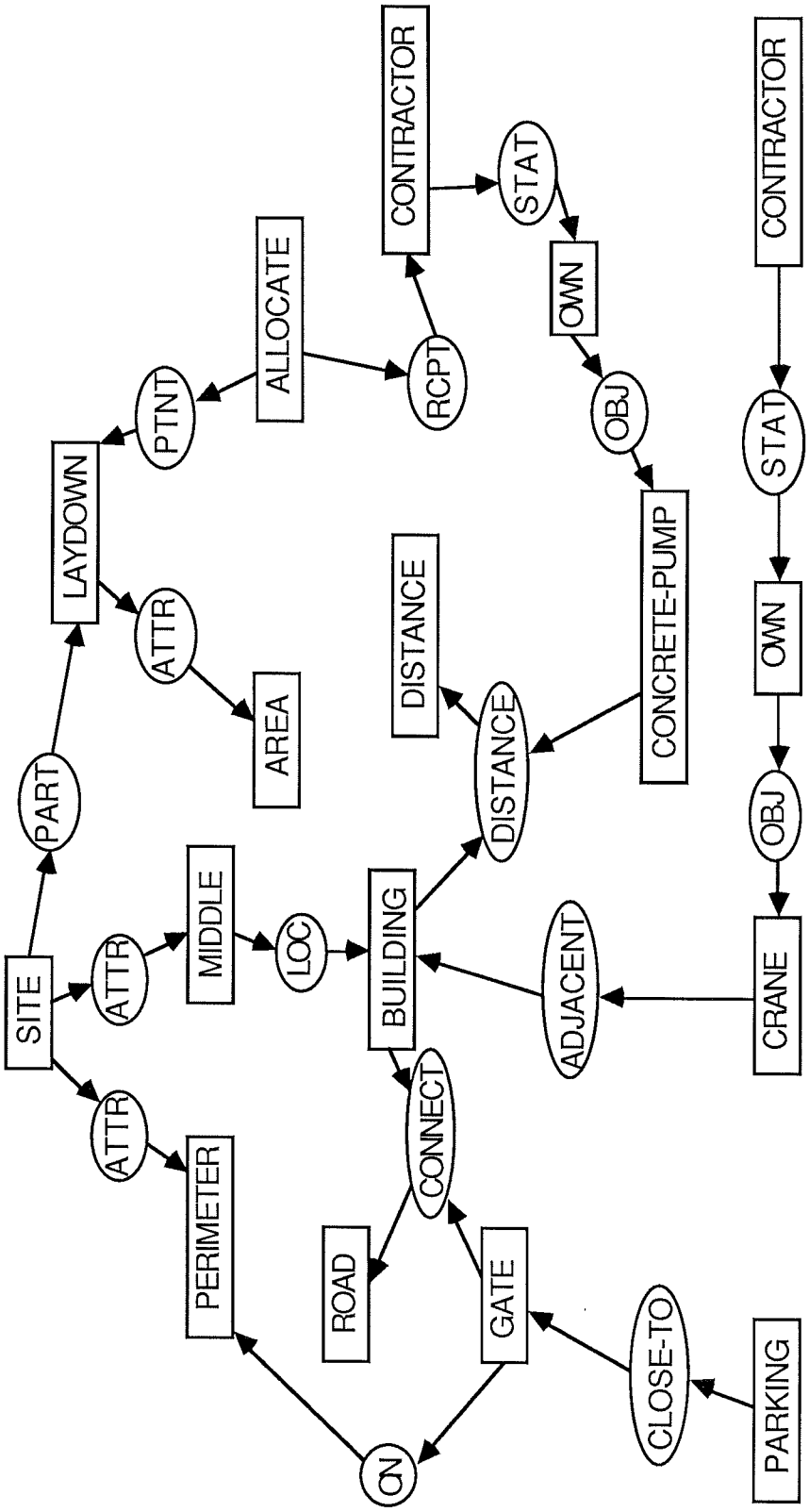
Trailers on an industrial construction site are quite large, but they are still small enough to easily be moved around on site. Several trailers can be put together to form a larger unit.

prototype of WAREHOUSE is

```
[ WAREHOUSE ] -
        -> (ATTR) -> [ LENGTH: @200feet ]
        -> (ATTR) -> [ WIDTH: @60feet ]
```

Warehouses are often standardized so that they can be constructed from prefabricated elements. In this case the length and width are multiples of 20 feet segments.

schema for SITE is



Appendix 6 - Inference Rules

LENGTH and WIDTH represent the DIMENSIONS of a rectangle.

[RECTANGLE]
-> (ATTR) -> [WIDTH: @a]
-> (ATTR) -> [LENGTH: @b]
is equivalent to [RECT] -> (ATTR) -> [DIMENSIONS: (@a @b)]

When two objects are ADJACENT they are CLOSE-TO each other.

relation [PHYSOBJ: *x1] -> (ADJACENT) -> [PHYSOBJ: *x2]
implies [*x1] -> (CLOSE-TO) -> [*x2]

When a road INTERSECTs with another one it means that the two roads CONNECT to the same point.

relation [INTERSECT]
-> (AGNT) -> [ROAD: *x1]
-> (PTNT) -> [ROAD: *x2]
and [ROAD: *x1] <- (CONN) -
<-1 [*u]
<-2 [*w]
implies
¬[[ROAD: *x2] <- (CONN)
<-1 [*u]
<-2 [*z]]

¬[[ROAD: *x2] <- (CONN)
<-1 [*w]
<-2 [*z]]

¬[[ROAD: *x2] <- (CONN)
<-1 [*z]
<-2 [*u]]

¬[[ROAD: *x2] <- (CONN)
<-1 [*z]
<-2 [*w]]

When two objects are at a DISTANCE of at most 25 yard from each other, then they are CLOSE-TO each other.

relation [DISTANCE: *w] <- (DISTANCE) -
<-1 [PHYSOBJ: *x2]
<-2 [PHYSOBJ: *x3]
and [*w] ≤ 25 yard
implies [PHYSOBJ: *x2] -> (CLOSE-TO) -> [PHYSOBJ: *x3]

When two objects are ACROSS a third object from each other, then the distance between them is at least the width of the third object, and at most the distance at 45 degrees over the third object (Figure A).

relation [PHYSOBJ: *x1] <- (ACROSS) -
<-1 [PHYSOBJ: *x2]
<-2 [PHYSOBJ: *x3]
implies [DISTANCE: *y] <- (DISTANCE) -
<-1 [PHYSOBJ: *x2]
<-2 [PHYSOBJ: *x3]
and [(λ z) [*x1] -> (ATTR) -> [WIDTH: *z]] ----- [*q]
and [*q] ≤ [*y] < (SQRT (* 2 (SQUARE [*q])))

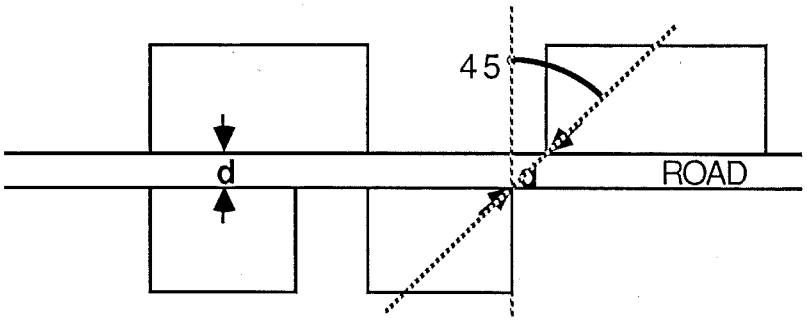


Figure A: Definition of ACROSS

When an object is ADJACENT to a second one, the second is CLOSE-TO a third, and the dimensions of the second are SMALL with respect to both the first and third , then the first object is CLOSE-TO the third object as well.

relation [PHYSOBJ: *x1] -> (ADJACENT) -> [PHYSOBJ: *x2]
and [PHYSOBJ: *x2] -> (CLOSE-TO) -> [PHYSOBJ: *x3]
and [(λ y) [*x2] -> (ATTR) -> [DIMENSIONS: *y]] -> (SMALL)
-> [(λ y) [*x1] -> (ATTR) -> [DIMENSIONS: *y]]
and [(λ y) [*x2] -> (ATTR) -> [DIMENSIONS: *y]] -> (SMALL)
-> [(λ y) [*x3] -> (ATTR) -> [DIMENSIONS: *y]]
implies [PHYSOBJ: *x1] -> (CLOSE-TO) -> [PHYSOBJ: *x3]

When an object is ALONG another object then the two objects are ADJACENT to each other.

relation [PHYSOBJ: *x1] -> (ALONG) -> [PHYSOBJ: *x2]
implies [PHYSOBJ: *x1] -> (ADJACENT) -> [PHYSOBJ: *x2]

When two objects are ALONG a road, then that road CONNECTs both objects.

relation [PHYSOBJ: *x1] -> (ALONG) -> [ROAD: *q1]
and [PHYSOBJ: *x2] -> (ALONG) -> [*q1]
implies [*q1] <- (CONNECT)
<-1 [PHYSOBJ: *x1]
<-2 [PHYSOBJ: *x2]