

# Reasoning With Diagrams Only

**George W. Furnas**  
**Cognitive Science Research Group**  
**Bell Communications Research**  
**gwf@bellcore.com**

## **Abstract**

Traditional deductive systems work with sentences of symbols. Even in newer systems that also reason from diagrams sentential representations still play a major role. The work here explores deductive systems that use only picture-like representations. Machinery functionally equivalent to variables, quantifiers, substitution, unification, and binding are defined based on a model of deductive chaining as the composition of mappings, where pictures themselves are used to specify the mappings.

Appeared in the proceedings of the AAAI Spring Symposium on "Reasoning with Diagrammatic Representations," Stanford CA, 3/25/92-3/27/92.

# Reasoning With Diagrams Only

George W. Furnas

*Cognitive Science Research Group*

*Bell Communications Research*

## Abstract

Traditional deductive systems work with sentences of symbols. Even in newer systems that also reason from diagrams sentential representations still play a major role. The work here explores deductive systems that use only picture-like representations. Machinery functionally equivalent to variables, quantifiers, substitution, unification, and binding are defined based on a model of deductive chaining as the composition of mappings, where pictures themselves are used to specify the mappings.

## 1. The problem

In the last several years there have been increasing efforts to take diagrams seriously in the support of formal reasoning. An excellent example of how they can be put to good use is the work of Barwise and Etchemendy<sup>[1]</sup>, who have been using Situation Theory to place diagrammatic and sentential representations within a common semantic framework. The result is a rigorous heterogeneous reasoning system often allowing more efficient proofs. Like other existing diagrammatic systems, three classes of inference are supported: concluding new diagram states from formal sentences, new formal sentences from diagrams, and new formal sentences from other formal sentences. The goal of the work here is to explore explicitly the neglected fourth possibility: inferring diagrams directly from other diagrams.

Diagram-to-diagram inference in its most extreme form would give first-class status almost exclusively to picture-like<sup>1</sup>, spatial entities, with as little recourse as possible to sentential representations. Thus in the work to be presented here, the diagrams are taken as intrinsically spatial structures, without an underlying sentential structural

description (*not* “Line *A* from point *x* to point *y*; Line *B* from...”); picture-like entities are used directly to represent sets of pictures. Unfortunately, much familiar deductive machinery associated with sentential representations is lost (variables, quantifiers, term unification, substitution). One fundamental goal is to generalize that familiar machinery and invent new versions specialized to pictorial representations.

## 2. A framework for deduction with picture-like objects

Figure 1 shows a standard deductive chain in a sentential system for algebraic manipulation (certain subtleties about equality are finessed here by casting this as a rewrite system). New methods are needed to support comparable deductions using picture-like representations such as those in the geometry example of Figure 2. These methods

$$\text{Axiom 1 : } \forall x \forall y \quad x + y \rightarrow y + x$$

$$\text{Axiom 2 : } \forall v \quad v + 0 \rightarrow v$$

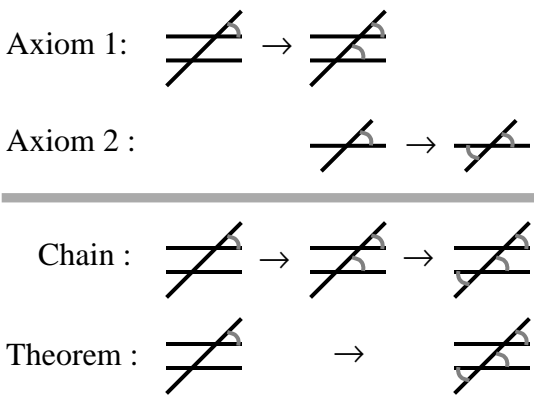
---

$$\text{Chain : } \forall w \quad 0 + w \rightarrow w + 0 \rightarrow w$$

$$\text{Theorem : } \forall w \quad 0 + w \rightarrow w$$

**Figure 1 .A classical example of deductive chaining: the derivation of a left-additive identity from commutative and right-additive identity axioms.**

1. While the terms “diagram”, “picture”, “image”, etc. in many contexts make useful distinctions, for the current purposes they are used interchangeably to represent a broad class of spatially structured representations. Their various computational properties in the context of the current explorations, have yet to be distinguished.



**Figure 2 . A hypothetical examples of chaining in a graphical deduction system for Euclidean geometry.**

should allow the diagrammatic rules to stand by themselves, without underlying sentential descriptions, and yet still support the deduction indicated rigorously. Currently this is not possible -- the gist of the deduction may seem sensible, but it takes a lot of hand-waving to make it work. For example, “Axiom 2” has only two lines drawn in it, but must be applied to a picture with three lines in it. No explicit machinery has been defined to permit that. Similarly those two lines cross in their middles, yet where the axiom must be applied the relevant crossing is two-thirds of the way from the ends. Again no machinery has been carefully defined here to justify this action. The pictures must, in some carefully defined way, apply to a whole set of cases, not just the literal example shown in the “axiom” itself. Comparable difficulties are handled in familiar logics by mechanisms of variables and quantifiers, unification and substitution, etc. There are no obvious analogs of such machinery for the pure picture case (e.g., where are the variables of Figure 2?)

Still, some sufficient functional analogs of the familiar machinery are needed to support deduction. This has required re-examining the familiar case of Figure 1 in a more general framework, one which will be broad enough to also encompass picture-based systems like those suggested in Figure 2. This general framework can guide the invention of specific theoretical machinery for picture systems to accomplish much of what is accomplished with variables, variable binding, term-unification, etc. in familiar logics.

The general framework we have used casts rules such as those in Figures 1 and 2 simply as map-

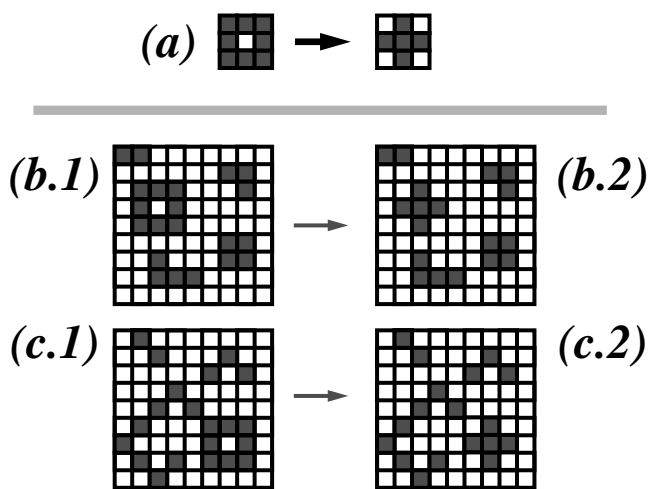
pings between sets of objects in a domain,  $f: A \rightarrow B$ . Thus Axiom 1 of the algebra example is a particular mapping of elements in the set of expressions,  $A=\{3+5, 9+12, 0+4, \dots\}$  into those in the set of expressions  $B=\{5+3, 12+9, 4+0, \dots\}$ . Similarly, Axiom 2 is another mapping,  $g: C \rightarrow D$ , which maps elements in the set  $C=\{3+0, 4+0, \dots\}$  into those in the set  $D=\{3, 4, \dots\}$ .

These sets of expressions are specified by the use of variables and quantifiers. The first lesson of the general framework, however, is this: what matters is that sets and mappings are specified, not that variables and quantifiers are used. Thus if one had other ways to specify sets and mappings between them, ways more appropriate for pictures, one might still have the rudiments of a deduction system. Several such specification systems more appropriate to pictures, are suggested in [2]. One in particular is discussed below.

### 3. The BITPICT system

In this section an example of a particularly simple formal system is presented. Although it is not powerful enough to handle the Euclidean geometry example in full generality, it is sufficient to give a better sense of what picture deduction systems might be like. It can be used to solve some reasonable spatial problems, and it has interesting underlying theoretical structure which will be discussed in Section 4.

In the approach taken here, the primitive notions are the specifications for sets (e.g., of arithmetic expressions, pictures) and for the mappings between them. For the BITPICT system, the universe is pictures, specifically, bitmaps, i.e., regular grids of picture elements (pixels) that are either black or white. In the familiar algebra case, a set of numerical expressions is specified by an expression with universally quantified variables. Here a set of bitmap pictures is specified by specifying a small piece of a bitmap, called a *bitpict*. A bitpict specifies the set of all bitmaps which contain that piece somewhere. The algebraic rules are made up of a pair of expressions (on the left- and right-hand sides of the arrows) that use the same variables. For bitpicts, a rule is an ordered pair of bitpicts whose pixel subsets are the same, though their bit



**Figure 3 . (a) A BITPICT rule mapping the 3x3 grid of pixels containing a circle to the 3x3 grid containing a cross. (b.1) and (c.1) show two example pictures in the left-hand set of the rule. (b.2) and (c.2) show the corresponding pictures in the right-hand set.**

values may differ. Such a rule is taken to be a mapping that simply associates left- and right-hand pictures that differ only in that the left-hand bitpict has been replaced by the right-hand one. (See Figure 3.)

This description in terms of sets and mappings is important for working out the underlying theory wherein deduction of theorems corresponds to chaining these mappings. Much interesting computation can be done, however, without that theory, and for understanding examples using the BITPICT computational system described below, it is sufficient to think of the rules as graphical search and replace operators.

A simple reasoning system based on bitpicts has been implemented on a Symbolics Lisp Machine. The architecture of the system is similar to that of standard production systems. That is, like classical production systems it has a set of rules that interact via a shared blackboard. Each rule has a left-hand side that constantly looks at the blackboard for a match. When a rule finds a match, its right-hand side directs how to change the contents of the blackboard. The change is made, and the process iterates. The rules are engineered so the evolution of the blackboard solves the intended problem.

Unlike standard production systems, where the structures on the board and in the rules are typi-

cally strings of symbols, in the BITPICT system they are simple picture fragments like those in Figure 3. Picture fragments are matched and replaced and problems are solved by the fragment-by-fragment evolution of the picture on the blackboard.

### 3.1 A BITPICT example: Counting the tangled forest

An example of the BITPICT system solving a spatial problem with graphical deductions is presented in Figure 4. The problem, counting the disconnected components in a tangled forest of bifurcating trees, is not easy to represent in a sentential way, yet has a very natural solution with bitpicts.

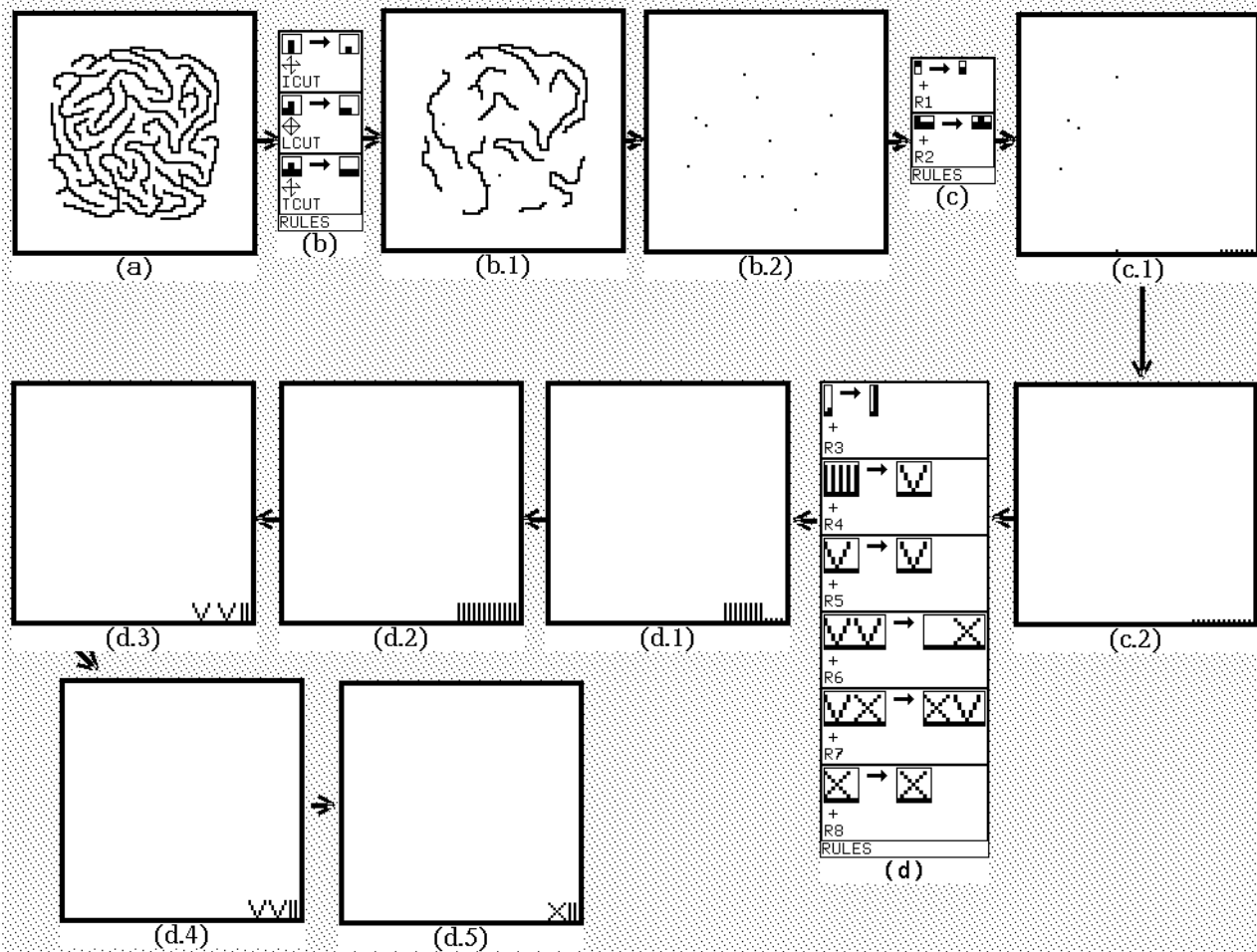
Part (a) of the figure shows a sample tangled forest on a square blackboard. The first set of rules, (b), simplifies the problem, basically by nibbling back the tips of the branches, nibbling at straight, corner and “T” sections respectively. The rules neither create nor destroy connected components, and so each component is gradually reduced, (b.1), to a dot (b.2).

The next rules, (c), move the dots down, and when they hit the bottom of the window, they are moved to the right, as shown in (c.1), until they are all canonically aligned in the lower right corner, (c.2). Again the invariant of number of components is preserved, and the problem of counting trees has been reduced to counting these neatly arrayed dots.

The final rules, (d) count the aligned dots by converting each to a vertical bar (an “I”) and from there to Roman numerals, with rules for simplifying (e.g., five I’s to a V), maintaining alignment (e.g., shifting V’s and X’s right as needed), and sorting (e.g., VX to XV). The final result is the Roman numeral, XII, indicating the number of trees in the original tangled forest.

The critical computational point is that the evolving state of the blackboard is governed by the picture-to-picture deduction rules, with no need for underlying sentential representation.

We have other BITPICT rule systems that simulate balls falling through hoppers and bouncing off



**Figure 4.** Counting the tangled forest. (a) A tangled forest of bifurcating trees. The rules, (b), nibble at the tips of straight, corner “L”, and “T” sections, reducing each component, (b.1), to a dot (b.2). The rules, (c), move the dots down, and to the right, (c.1), into the lower right corner, (c.2). The rules, (d), count the aligned dots in Roman Numerals, yielding the answer, XII. (The small icons under the LH bitpict of each bitpict rule indicate the invariances operating in matching those rules. The presence of vertical and horizontal lines appearing like a “+” sign in the icon indicate that vertical and horizontal translations are allowed. The diagonal segments indicate permitted rotations - - with a full pinwheel indicating all four rotations. Reflections are permitted if the diagonal segments are shown reflected -- so a full diamond with a cross indicates full translation, rotation, and reflection invariance.)

baffles, that do cartoon animations, that play tic-tac-toe, that solve simple wheel-and-pulley rotation direction problems, run PacMan, and more. (For some general examples, see <sup>[2][5]</sup>, for graphical interface mock-ups see <sup>[3][4]</sup>)

Note that the BITPICT system, like most rule-based expert systems, does not do any high level deductive chaining. It simply matches a low level instance against the LHS’s of its high level rules, and finds the corresponding RHS, and then iterates chaining low-level deductions. It does not prove any “theorems” along the way, in the sense of Figures 1 and 2. Bitpict rules, however do support such high level formal deductive chaining. More details on the theory can be found in <sup>[5]</sup>. Here a

brief overview of the principal issues of how such chaining can be accomplished is presented.

#### 4. Deductive chaining as function composition

If individual rules are mappings, then chaining corresponds to a composition of those mappings (i.e., a new mapping which would be equivalent to applying the second rule to the result of the first). Thus in Figure 1, the result is the mapping which associates elements of the set,  $A'=\{0+3, 0+4, \dots\}$  with corresponding elements of the set,  $D'=\{3,4, \dots\}$ .

The general framework is illustrated in Figure 5 where two mappings,  $f$  and  $g$ , are chained together to produce a third mapping,  $h$ . The mappings  $f$  and  $g$ , might for example correspond to the first two axioms of Figure 1, as described in Section 2. The composition of these two functions is supposed to be what goes on in the chaining in the bottom half of that figure. Figure 5 can be used to understand in more detail what is going on in the familiar case, so that comparable machinery may be invented for picture deduction.

First note the appearance of  $B \cap C$  in the lower half of Figure 5. This reflects the important fact that not all results of the first rule (elements of  $B$ ) may be eligible for applying the second (they must also be elements of  $C$ ). Thus only various *partial compositions* will generally be possible. The *maximal* partial composition is mediated by the intersection of these two sets,  $B \cap C$ . Thus in the example of Figure 1, the intersection of the set  $B = \{3+5, 9+12, 0+4, \dots\}$  and the set  $C = \{3+0, 4+0, \dots\}$  is  $B \cap C = \{3+0, 4+0, \dots\}$  (in this rather degenerate case, the same as  $C$ ). Only elements in this intersection set can mediate the partial composition.

Although what really matters is what is going on with the sets, the machinery must actually work not with the sets themselves, but their specifications. In standard logics, finding the specification of the intersection set is the goal of what is called *unification*: the expression “ $y+x$ ” specifying  $A$  is *unified* with the expression “ $v+0$ ” specifying  $B$ , to yield the unifying expression “ $w+0$ ” which specifies their intersection (variables are all renamed to

$$\text{(Axiom 1) } f: A \rightarrow B$$

$$\text{(Axiom 2) } g: C \rightarrow D$$

---


$$\text{(Chain) } f^{-1}(B \cap C) \rightarrow B \cap C \rightarrow g(B \cap C)$$

$$\text{(Theorem) } h: f^{-1}(B \cap C) \rightarrow g(B \cap C)$$

**Figure 5 . Deductive chaining as the composition of mappings between sets. This is supposed to be exactly structurally analogous to the deductive chains in Figure 1, Figure 2 and the later Figure 6.**

be distinct, for technical reasons). The operation of unification applied to expressions must be equivalent (isomorphic) to the operation of intersection applied to the corresponding sets. Any new mechanisms for specifying sets (e.g., bitpicts specifying sets of bitmaps) will require comparable machinery for finding the specification of the corresponding intersection sets.

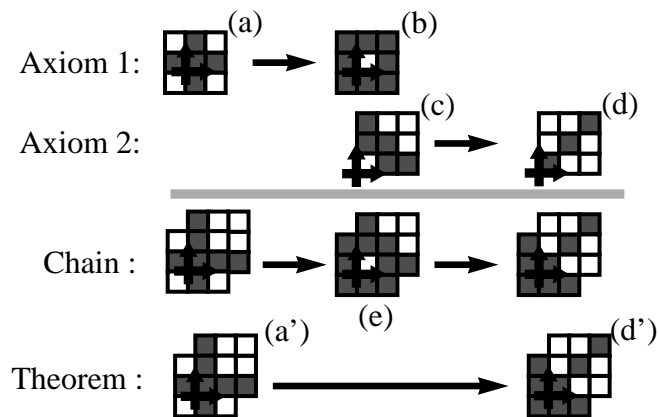
Finding the specification of the intersection set is only the beginning of the machinery needed for composition. In particular, some members of the set  $A$  cannot participate in the chain -- only those that map into the intersection. Thus to specify the new “Theorem”  $h$ , we must have a way to specify this special restricted subset of  $A$ , called “the pre-image of  $B \cap C$  under  $f$ ”, written in Figure 5 as  $f^{-1}(B \cap C)$ . Similarly, the all the results of applying  $h$  will form only a subset of  $D$ , only those elements gotten by applying  $g$  to some element in the intersection (“the image of  $B \cap C$  under  $g$ ”, written in the figure as  $g(B \cap C)$ ). So to specify  $h$  we must also have a way to derive the specification of both these pre-and post-image sets.

In the case of standard logics, this is accomplished using the so-called unifying substitutions. The specifications of  $B$  and  $C$  can be unified by substitution with the bindings  $[y \rightarrow w, x \rightarrow 0, v \rightarrow w]$ . (I.e., making these substitutions on “ $y+x$ ” and “ $v+0$ ” turns both into “ $w+0$ ” ( $B \cap C$ ).) One simply performs the same substitutions on the expressions for  $A$  and  $D$  to get the expressions of the correspondingly restricted subsets. That this works (as the reader may verify) may be familiar, but it is by no means trivial. Thus, whatever new machinery is invented to represent sets of pictures and mappings between them must not only support ways to calculate intersections (generalized unification), but also ways to implement restrictions of the original mappings to this critical intersection set (e.g., by some generalization of binding and functional equivalent of substitution).

A class of such systems more suitable for pictures is discussed in [4], and illustrated in Figure 6 for fixed-position bitpicts (bitpicts with a fixed origin located at the crossed arrows in the figure). The underlying theory is based on taking pictures themselves as functions over the plane and using

various algebraic structures built over sets of functions. For example a bitmap can be considered a function over the plane in that each pixel location  $(i,j)$  has a pixel value associated with it. Bitpicts then are partial functions, since they specify only a fragment of such a mapping, with the rest of the plane undefined. Partial functions have a natural (partial-) ordering based on extension, wherein the completely undefined function is maximum (the null bitpict specifies the set of all bitmaps), and all fully defined functions are minimal (complete bitmaps specify a single picture). Intersections of sets of pictures is accomplished by conjoining bitpicts (technically a *meet* operation in the partial ordering), yielding an analog of unification. An example appears in the middle column of Figure 6, where the bitpict (b) of Axiom 1 and (c) of Axiom 2 are unified to give (e) which mediates the chain. This bitpict (e) is the least restrictive bitpict which contains (b) and (c), and specifies the set which is exactly the intersection of the sets specified by (b) and (c).

This partial ordering structure further allows the rigorous definition of machinery to do the restrictions of the mappings  $f$  and  $g$  to the pre- and post-images of this intersection set. Instead of applying the substitution operator with a given substitution vector to various terms, one applies the meet operation with a distinguished element of the partial order structure to the various other elements. (See [5] for details.)



**Figure 6 . A high-level deductive chain using fixed-position bitpicts. The chaining is made rigorous by machinery based on meet operations on the partially ordered set of bitpicts considered as partial functions of the plane. The small crossed arrows within each bitpict indicate the location of its fixed origin.**

Derived pictorial “theorems” can be very useful. For bitpicts they can encapsulate the behavior of spatial aggregates, with graphical device behavior, for example, built up from graphical “axioms” about component behavior.

## 5. Conclusions

This overview has been fairly cursory. The main point is that new kinds of machinery can be defined to do rigorous deductions using non-sentential representations, where instead picture-like entities represent sets of pictures directly.

Ultimately inference systems for spatial reasoning will need to be heterogeneous. A critical area for future research is to understand how this fourth sort of deduction, diagram-to-diagram, can be integrated with the three other, more familiar modes: diagram-to-sentence, sentence-to-sentence, and sentence-to-diagram.

## 6. References

- [1] Barwise, J. and Etchemendy, J., Visual information and valid reasoning, in *Visualization in Teaching and Learning Mathematics*, S. Cunningham and W. Zimmerman (Eds.), Washington: Mathematical Association of America, 1991.
- [2] Furnas, George W., Formal models for imaginal deduction, *Proceedings of the Twelfth Annual Conference of the Cognitive Science Society*, Hillsdale, NJ: Lawrence Erlbaum, 1990, 662-669.
- [3] Furnas, George W., “Graphical Reasoning for Graphical Interfaces,” in *SIGGRAPH VIDEO #56*, 1990.
- [4] Furnas, George W., New Graphical Reasoning Models for Understanding Graphical Interfaces, *Human Factors in Computing Systems CHI '91 Conference Proceedings*, New Orleans, April 28 - May 2, 1991, 71-78.
- [5] Furnas, George W., Deduction with pictures only: A model based on deductive chaining as the composition of functions via operations on their specifications, *Bellcore Technical Memorandum*, 1992.