12 Symbolic Constraints and Propagation

In this chapter, you learn how *symbolic constraint-propagation procedures* can determine the consequences of interacting constraints.

By way of illustration, you learn about a program that interprets drawings, and about another program that finds relations among time intervals.

Once you have finished this chapter, you will know that, when a domain is well understood, it is often possible to describe the objects in the domain in a way that uncovers useful, interacting constraints. You will also know how to use *Marr's methodological principles* when you work on difficult problems.

PROPAGATION OF LINE LABELS THROUGH DRAWING JUNCTIONS

In this section, you learn about propagating symbolic labels through nets. In particular, you learn about symbolic constraint propagation in the context of understanding drawings of plane-faced objects, such as those in figure 12.1. The main problem is to determine which lines are boundary lines that separate objects. You see that boundary, convex, concave, shadow, and crack lines come together at junctions in only a few ways, and then you see that this restriction on junction combinations determines the proper physical interpretation for each line in a drawing. Once correct line interpretations are known, it is easy to use known boundary lines to divide

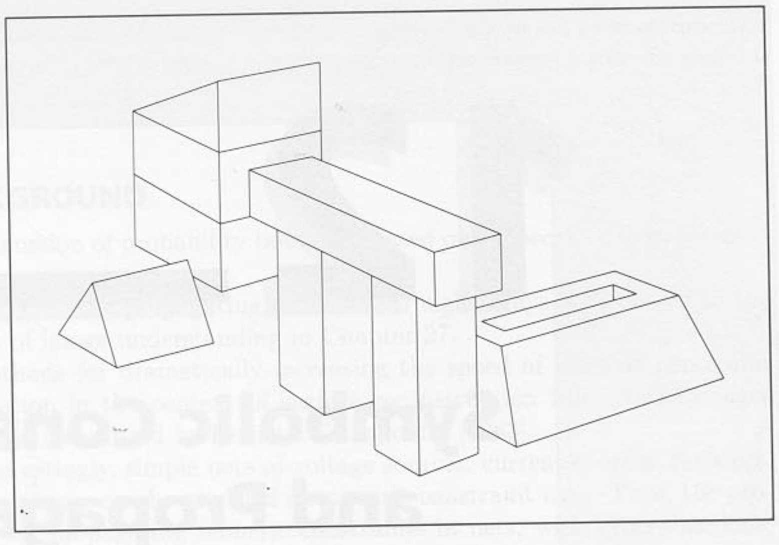


Figure 12.1 Part of drawing analysis is to decide how each line in a drawing should be interpreted.

the drawing into objects. Along the way, you see that some impossible drawings can be detected, because there is no way to interpret all the lines consistently.

There Are Only Four Ways to Label a Line in the Three-Faced-Vertex World

Consider a world populated by crack-free polyhedra with lighting arranged to eliminate all shadows. The lines in drawings of this world represent various naturally occurring edge types. A simple partitioning of these lines is shown in figure 12.2.

All lines are divided into boundary lines and interior lines. **Boundary lines** occur where one object face hides the other. The two regions in the drawing separated by a boundary line *do not* abut along the boundary line. **Interior lines** are those for which the two separated regions *do abut* one another. The interior lines are those that are associated with concave edges and those that are associated with convex edges.

For notational convenience, line interpretations are identified on drawings by **line labels**. There are three such labels:

Line	Label
Convex	+
Concave	−
Boundary	>

You determine the direction of the boundary line label by noting which side of the line corresponds to a face of the object causing the boundary line. Imagine taking a stroll along the line, keeping the boundary-line object on the right. The direction of walking is the direction of the boundary label.

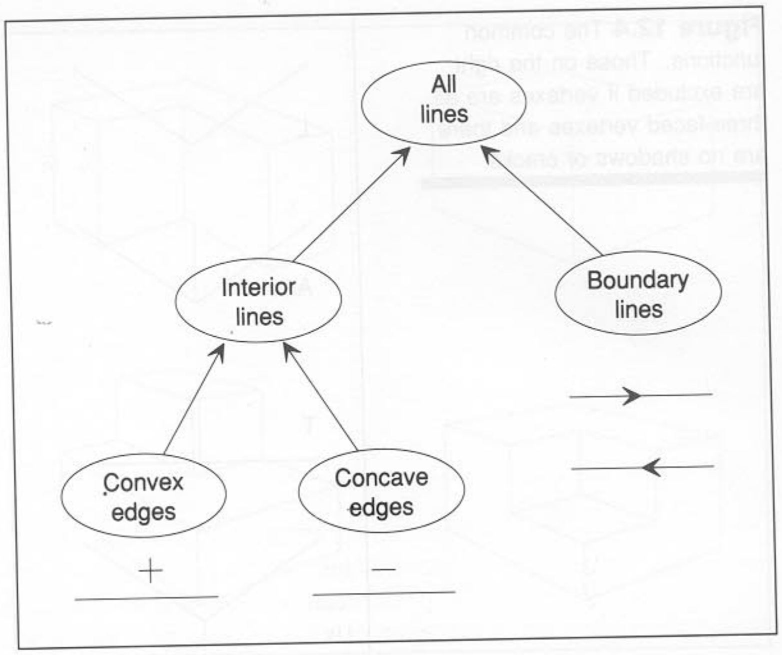


Figure 12.2 Drawings consist of boundary lines and interior lines. The interior lines may be concave or convex.

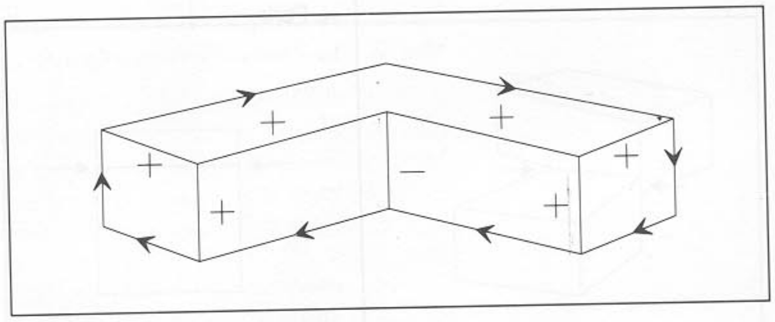


Figure 12.3 An L–shaped solid illustrates the three basic line interpretations: convex lines, marked with plus labels; concave lines, marked with minus labels; and boundary lines, marked with boundary labels.

Combinations of line labels surrounding junctions are called **junction labels**. Natural constraints severely limit the number of junction labels that are physically realizable.

It is easy to label each of the lines in figure 12.3 such that all the junction labels are physically realizable by using your intuition. By so labeling a drawing, you exploit your understanding of the physical situation to arrive at interpretations for the lines. The key idea to pursue now is that of turning the process around, using knowledge about allowable junction labels to derive an understanding of the physical reality.

Accordingly, you need a catalog of physically realizable junctions. To keep straight the distinction between a drawing and the actual physical world, note that **junctions** in drawings denote physical **vertexes** in the world, and **lines** denote physical **edges**.

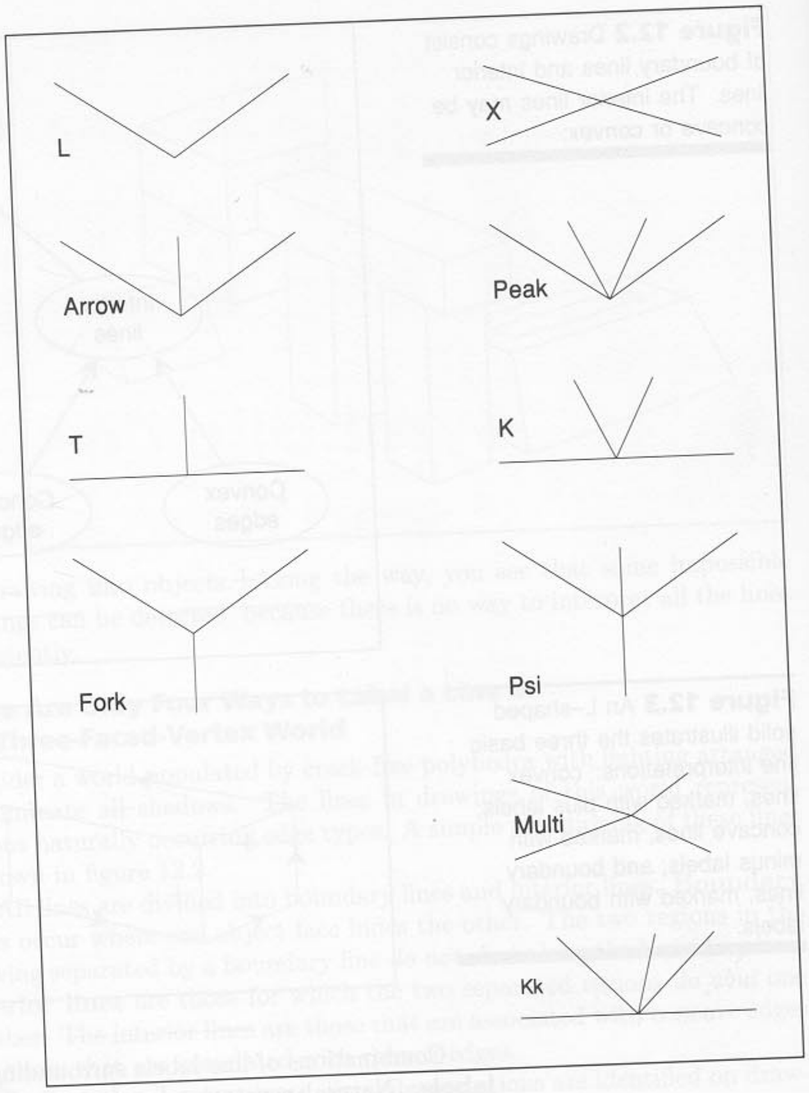


Figure 12.4 The common junctions. Those on the right are excluded if vertexes are all three-faced vertexes and there are no shadows or cracks.

Junctions can be categorized according to the number of lines coming together and the size of the angles between the lines. Figure 12.4 assigns mnemonic names to the common categories.

Fortunately, the following simple assumptions exclude all junctions other than Forks, Arrows, Ls, and Ts:

- Limited line interpretations: There are no shadows or cracks.
- Three-faced vertexes: All vertexes are the intersection of exactly three object faces. The vertexes at the top of the Great Pyramids of Egypt are forbidden. The vertexes in figure 12.5 are allowed.

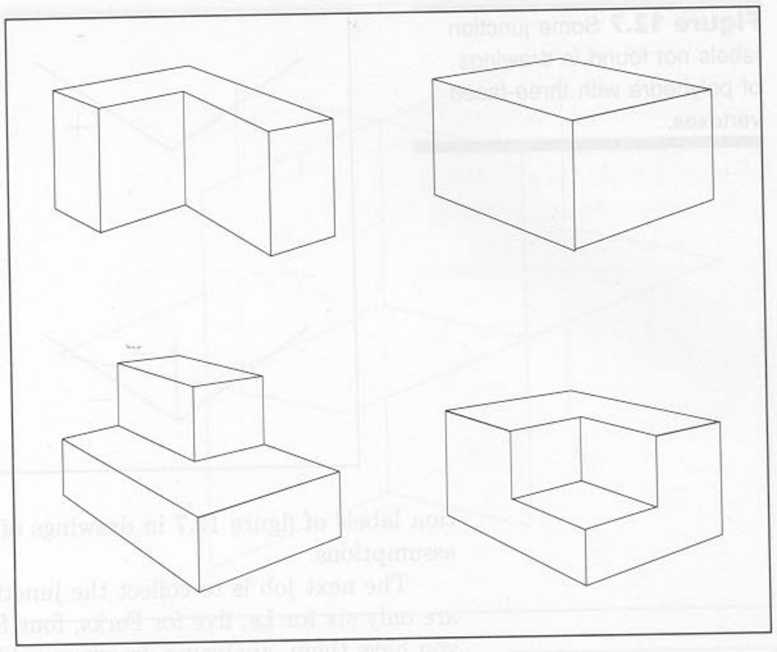


Figure 12.5 Some objects with exclusively three-faced vertexes.

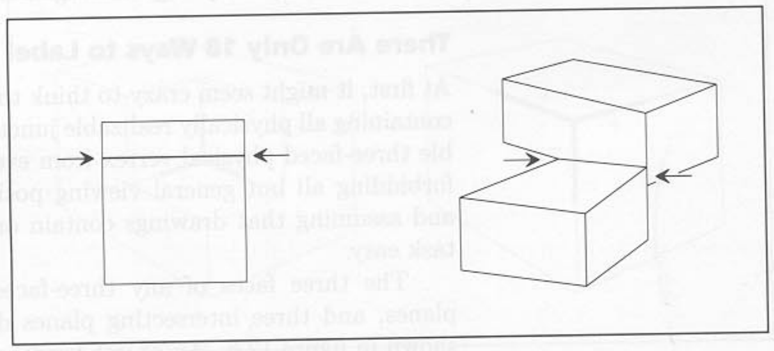


Figure 12.6 The criterion of general viewing position excludes both of these configurations because any perturbation of the viewing position changes the junctions indicated. On the left, you see the front and top of a cube, viewed without perspective.

- General position: The choice of viewpoint is such that no junctions change type with a small change of viewpoint. The viewpoints in figure 12.6 are forbidden.

These assumptions are in force only temporarily; later, they will be relaxed. The reason that these assumptions help you is that they reduce the number of junction possibilities and hence the number of interpretations possible for junction-surrounding lines.

Now, because there are four ways to label any given line, there must be $4^2 = 16$ ways to label an L. Similarly, there must be $4^3 = 64$ ways to label any particular Fork, Arrow, or T. Thus, the upper bound on the number of junction labels is 208. Curiously, only 18 of these combinations are physically realizable. It is not possible, for example, to find the junc-

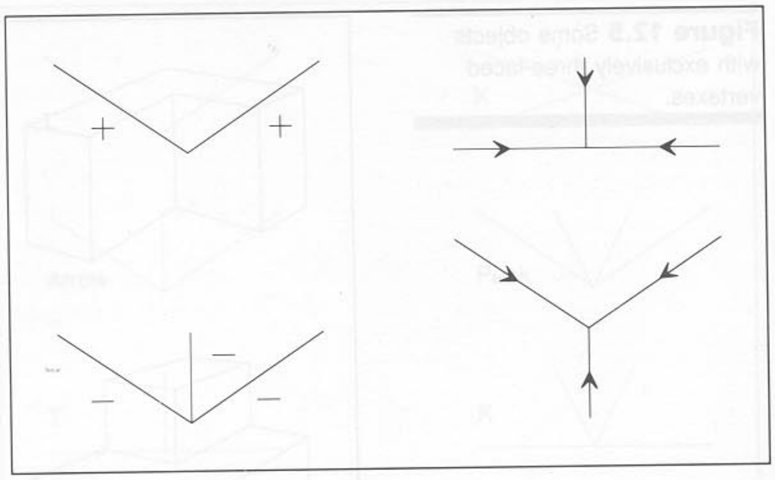


Figure 12.7 Some junction labels not found in drawings of polyhedra with three-faced vertexes.

tion labels of figure 12.7 in drawings of real polyhedral objects, given our assumptions.

The next job is to collect the junction labels that are possible. There are only six for Ls, five for Forks, four for Ts, and three for Arrows. Once you have them, analyzing drawings is like working easy jigsaw puzzles.

There Are Only 18 Ways to Label a Three-Faced Junction

At first, it might seem crazy to think that you can build a junction catalog containing all physically realizable junction labels by looking at every possible three-faced physical vertex from every possible direction. Fortunately, forbidding all but general viewing positions makes the task manageable, and assuming that drawings contain only three-faced vertexes makes the task easy.

The three faces of any three-faced vertex define three intersecting planes, and three intersecting planes divide space into eight octants, as shown in figure 12.8. An object forming a vertex plainly must occupy one or more of the eight octants so formed. Accordingly you can make a complete junction catalog by a two-step process: consider all ways of filling up eight octants with object material; and view each of the resulting vertexes from the unfilled octants.

Of course, if no octants are filled, or if all are filled, then there is no vertex, and consequently, there is nothing to consider. But suppose seven of the eight are filled, as in the left half of figure 12.9. Evidently, the seven-octant situation validates a Fork junction label in which each of the three lines involved bears a minus label. Note that the only junction of interest in the drawing is the one in the center. The surrounding drawing is only a visual aid to understanding how the seven filled octants produce a single drawing junction. Note further that, because seven octants are filled, there can be only one octant from which to look at the vertex. The junction

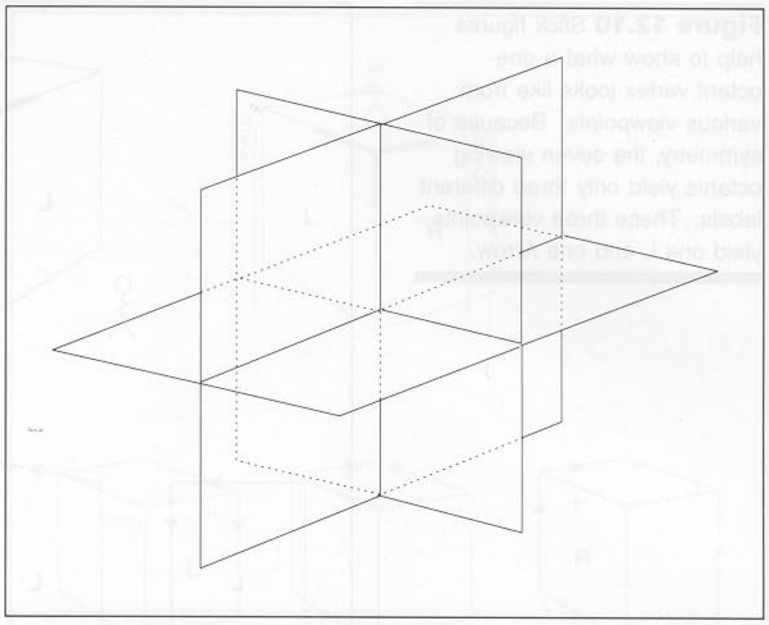


Figure 12.8 The three faces of a three-faced vertex divide space into eight octants. Here the planes meet at right angles. They need not.

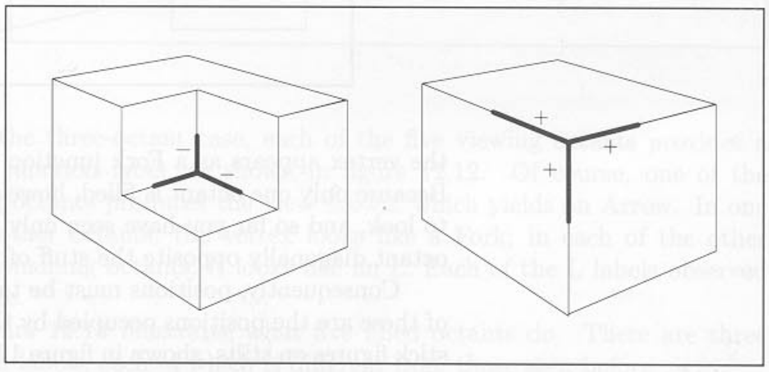


Figure 12.9 Junctions seen when seven octants are filled or when one is. On the left, the three concave lines are seen, no matter where the viewer stands within the one available viewing octant. On the right, the view from one octant is such that there is a Fork surrounded by convex labels.

type seen is a Fork, no matter what particular position is taken within the viewing octant. Also, the planes forming the octants do not need to be at right angles.

Fortunately, invariance within a viewing octant and indifference to plane angle hold in general. The junction type does not change as the viewpoint moves within one viewing octant or as the angles between the planes change.

So far, the junction catalog has but one entry, a Fork. One new entry is suggested by the right half of figure 12.9, in which the junction of interest is surrounded again by a drawing that provides a visual aid to understanding just what is filled and what is empty. From the point of view shown,

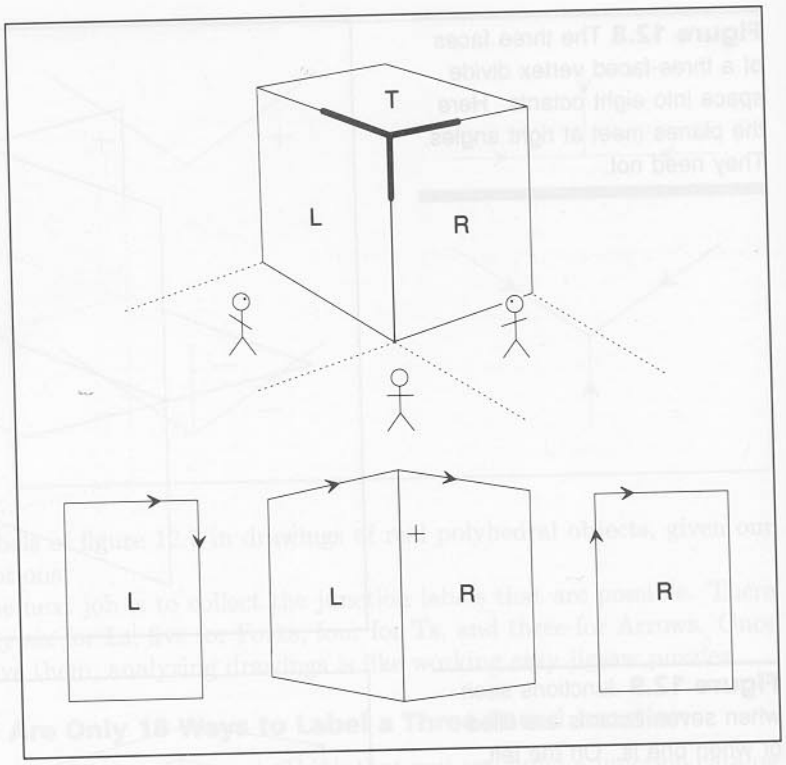


Figure 12.10 Stick figures help to show what a one-octant vertex looks like from various viewpoints. Because of symmetry, the seven viewing octants yield only three different labels. These three viewpoints yield one L and one Arrow.

the vertex appears as a Fork junction with each line labeled with a plus. Because only one octant is filled, however, there must be seven from which to look, and so far you have seen only the junction label derived from the octant diagonally opposite the stuff of the object.

Consequently, positions must be taken in the six other octants. Three of these are the positions occupied by the stick figures in figure 12.10. Two stick figures on stilts, shown in figure 12.11, occupy two positions above the plane defined by the top of the cube. And one final stick figure, standing on top of the cube, occupies the final position. All six stick-figure views provide only two new junction labels, because three of the views produce one kind of Arrow, and the other three produce one kind of L.

Now consider the situations with two, four, or six octants filled. All are excluded by the initial three-faced presumption. Suppose, for example, that two octants are to be filled. If the two were adjacent, then the edges between them would be cracks, there would be four object faces at the central vertex, and the vertex would not be three-faced. If the two filled octants were not adjacent, then they would meet either along an edge or at a common point. Either way, there would be more than three faces at the central vertex. Similar arguments exclude the four- and six-octant cases, leaving only the three- and five-octant cases to be considered.

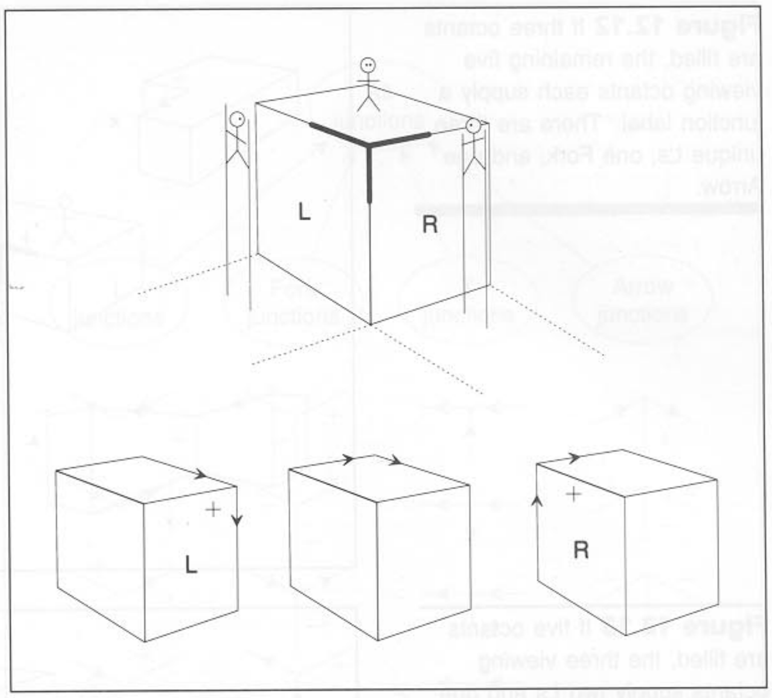


Figure 12.11 Stick figures help to show what a one-octant vertex looks like from various viewpoints. Because of symmetry, the seven viewing octants yield only three different labels.

In the three-octant case, each of the five viewing octants provides a unique junction label, as shown in figure 12.12. Of course, one of the viewing octants produces the view shown, which yields an Arrow. In one of the other octants, the vertex looks like a Fork; in each of the other three remaining octants, it looks like an L. Each of the L labels observed is unique.

Figure 12.13 illustrates what five filled octants do. There are three junction labels, each of which is different from those seen before.

Finally, because cracks are forbidden, Ts can be labeled in only four ways, all of which are consequences of partial occlusion. Thus, the total number of ways to label a junction is now 18, as collected together in figure 12.14.

Note that there are three junction labels in the Fork column that include boundary labels. All three could be considered rotated versions of one another. Three distinct labels appear, to emphasize that there are three distinct ways for a Fork to be labeled with boundary lines.

Now, all possible ways in which three-faced vertexes can be formed have been enumerated, and you have viewed each such vertex from all possible directions. You conclude that the 18 junction labels are all that there can be. Any other label cannot correspond to a physically realizable three-faced vertex.

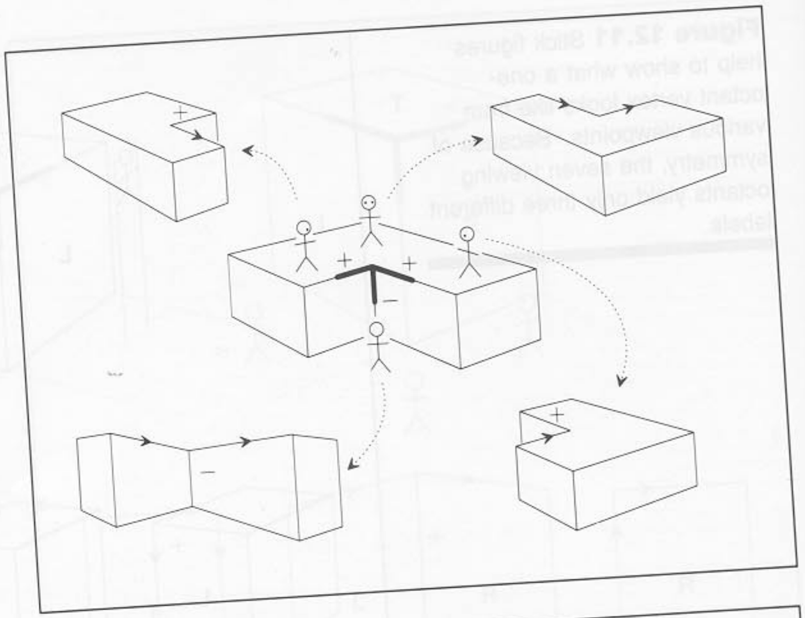


Figure 12.12 If three octants are filled, the remaining five viewing octants each supply a junction label. There are three unique Ls, one Fork, and one Arrow.

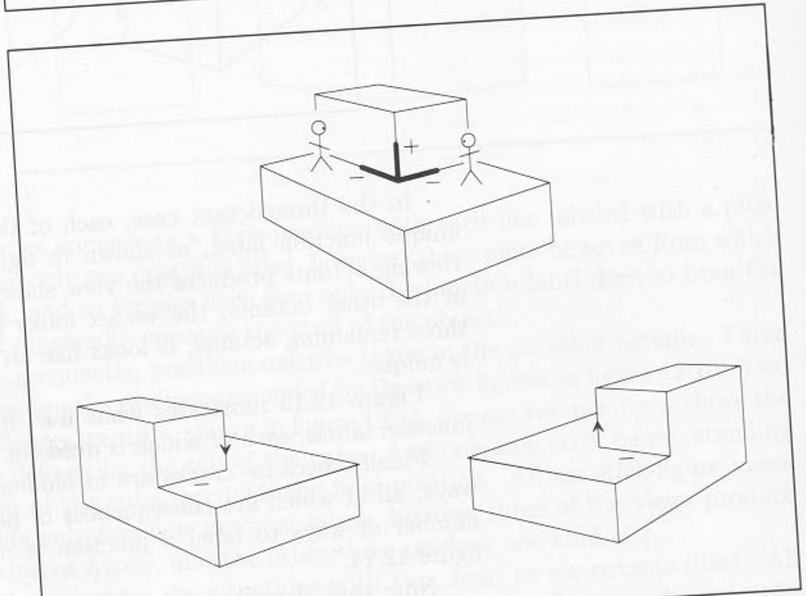


Figure 12.13 If five octants are filled, the three viewing octants supply two Ls and one Arrow.

Finding Correct Labels Is Part of Line-Drawing Analysis

Now let us examine examples showing how the junction catalog can be used. At first, assume that each object is suspended in space. Consequently, each object's background border has only boundary labels. Also note that there is only one kind of Arrow junction in the junction catalog that has boundary labels on its barbs. For any such Arrow, the shaft must be labeled with a

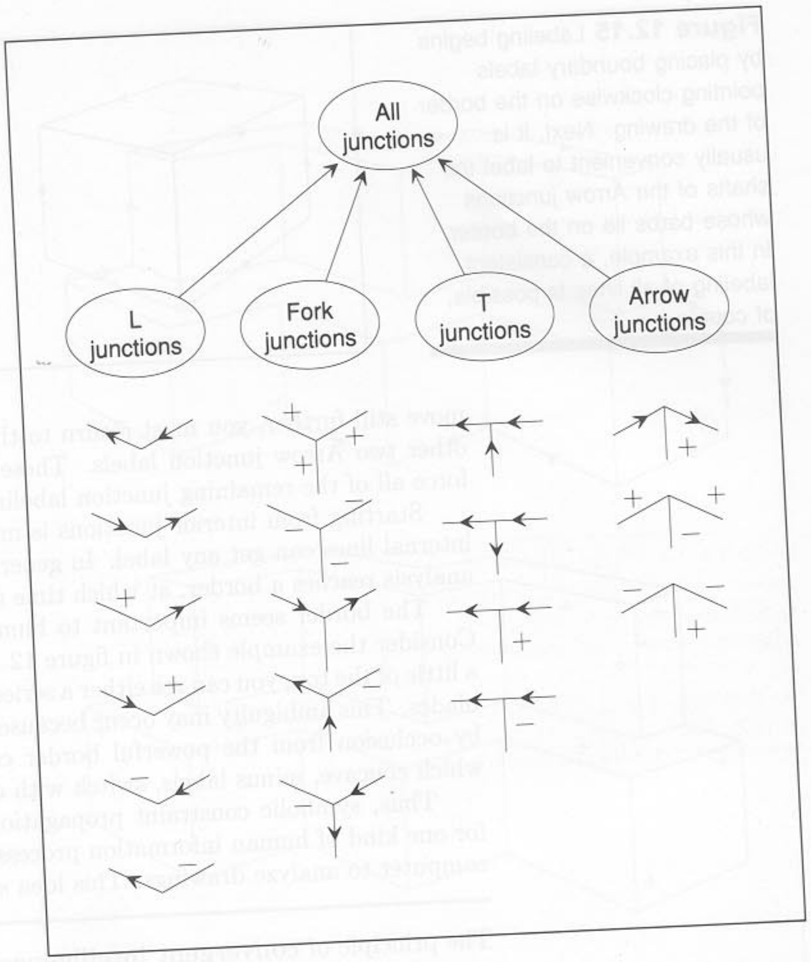


Figure 12.14 Eighteen junction configurations are possible. Were it not for natural constraints, there would be 208.

plus. Furthermore, there is only one kind of Fork with any plus label. For any such Fork, all the lines are forced to be labeled with plus labels.

Now consider the cube shown in figure 12.15. Because you are to imagine that the cube is suspended in space, the lines bordering on the background certainly can be labeled with boundary labels. Next, each of the Arrow's shafts is forced to be labeled with a plus because the barbs already have boundary labels. Now, only the central Fork remains to be investigated. Because all the Fork junction's lines already have plus labels assigned through previous considerations, it remains only to check that a Fork with three plus labels is in the junction catalog. It is.

Now consider the slightly harder example in figure 12.16, which is a sort of two-tiered, double L–shaped figure. Again, it is useful to begin by labeling the background border. Then, it is easy to move toward the interior using the Arrows with boundary labels on their barbs, together with the fact that a plus on any Fork line forces two more plus labels. To

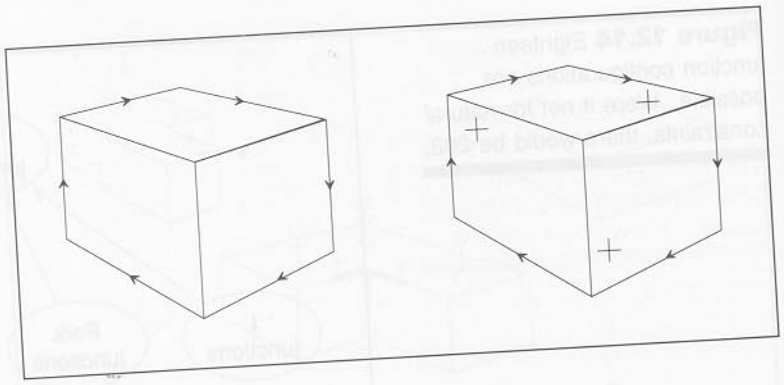


Figure 12.15 Labeling begins by placing boundary labels pointing clockwise on the border of the drawing. Next, it is usually convenient to label the shafts of the Arrow junctions whose barbs lie on the border. In this example, a consistent labeling of all lines is possible, of course.

move still further, you must return to the junction catalog to pick up the other two Arrow junction labels. These other two Arrow junction labels force all of the remaining junction labeling, as shown.

Starting from interior junctions is more difficult. Unlike border lines, internal lines can get any label. In general, some ambiguity remains until analysis reaches a border, at which time the ambiguity is usually resolved.

The border seems important to human analysis of drawings as well. Consider the example shown in figure 12.17. By covering up the sides and a little of the top, you can see either a series of ordinary steps or a row of saw blades. This ambiguity may occur because the interior junctions, separated by occlusion from the powerful border constraints, undergo reversals in which concave, minus labels, switch with convex, plus labels.

Thus, symbolic constraint propagation offers a plausible explanation for one kind of human information processing, as well as a good way for a computer to analyze drawings. This idea suggests the following principle:

The principle of **convergent intelligence**:

▷ The world manifests constraints and regularities. If a computer is to exhibit intelligence, it must exploit those constraints and regularities, no matter of what the computer happens to be made.

It is also interesting that the theory is useful not only in analyzing normal drawings, but also in identifying illegal drawings—those that cannot correspond to real objects. The drawing in figure 12.18 is illegal, a conclusion you can reach through a labeling argument. Proceeding as before, background lines, Arrow junctions with plus-marked barbs, and Fork junctions with plus labels can be exploited as shown. But now one junction is illegally labeled. The Arrow on the end of one arm insists on a minus label for that arm, whereas the Fork on the other arm demands a plus label for that arm. But because there is no L with one minus arm and one plus arm, the drawing cannot be a view of a polyhedron with three-faced vertexes.

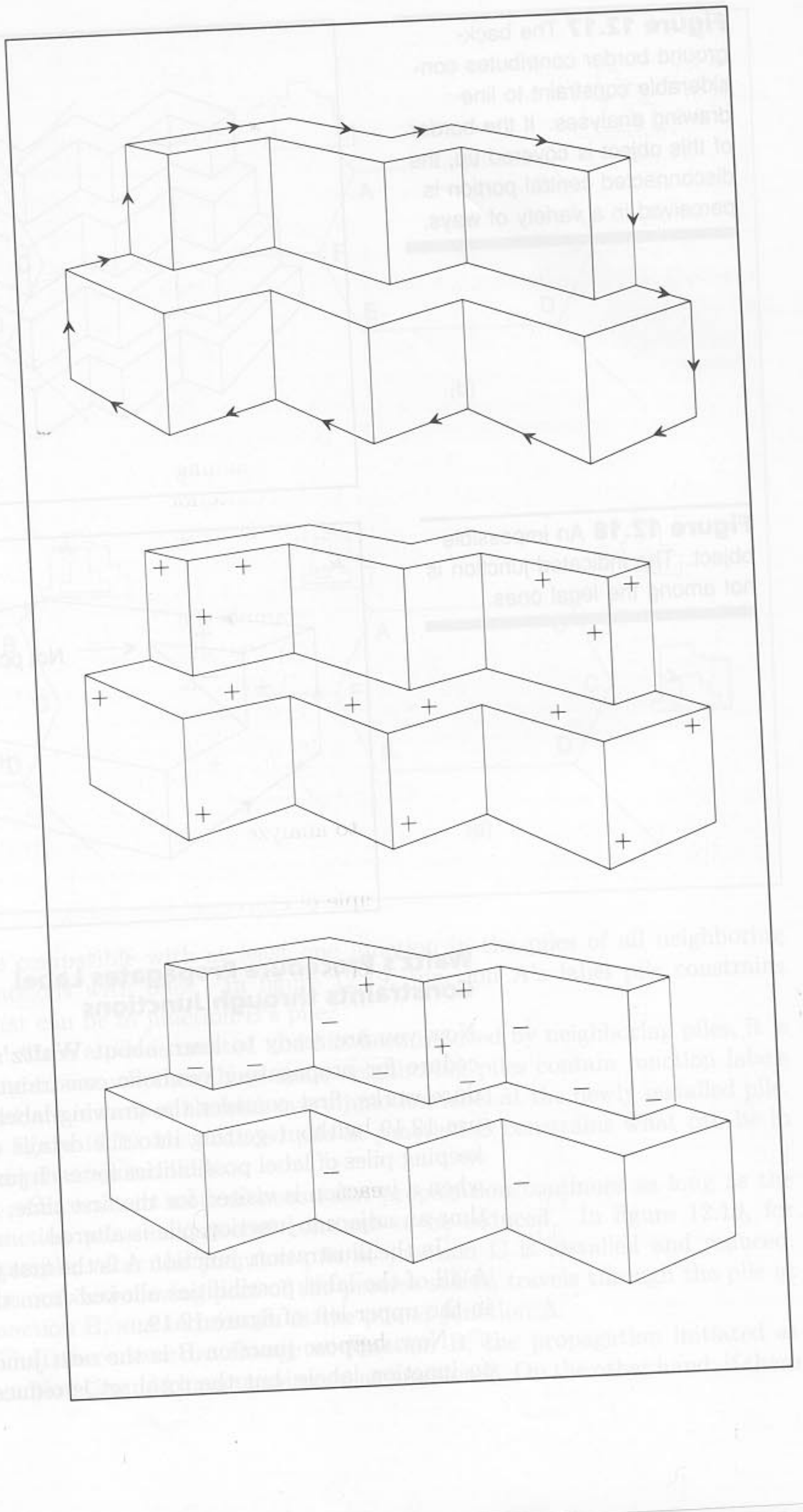


Figure 12.16 Labeling of this two-tiered figure begins with the background border. Next the shafts of the border Arrows begin a propagation of plus labels that continues through all Forks encountered. The rest of the job requires use of two other Arrow junctions found in the junction catalog.

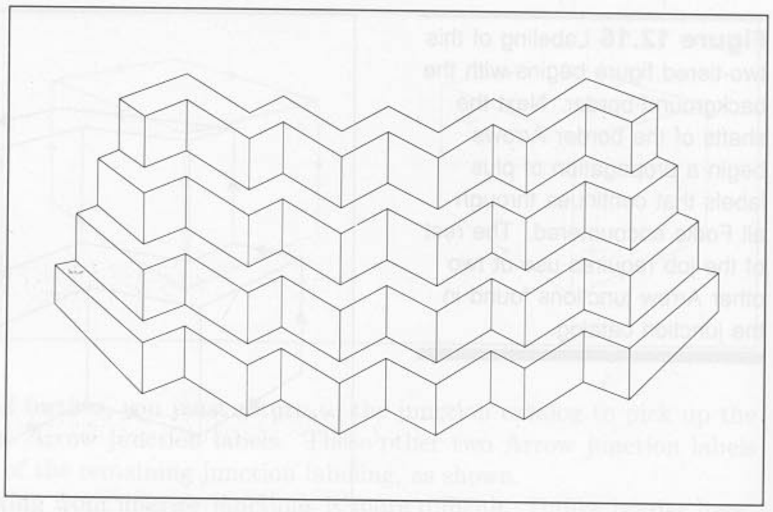


Figure 12.17 The background border contributes considerable constraint to line-drawing analyses. If the border of this object is covered up, the disconnected central portion is perceived in a variety of ways.

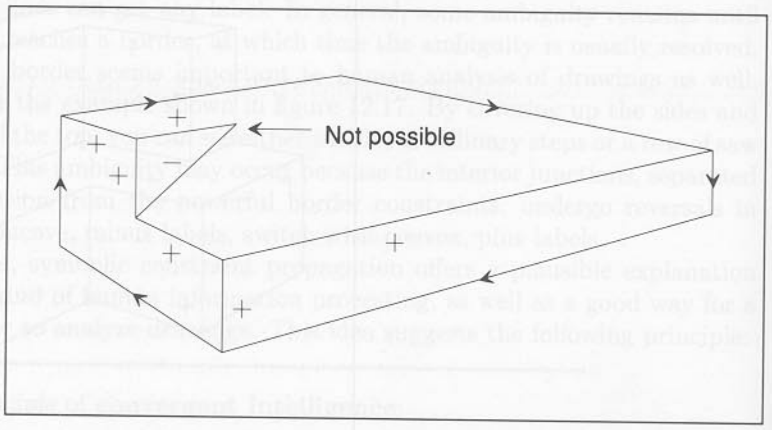


Figure 12.18 An impossible object. The indicated junction is not among the legal ones.

Waltz's Procedure Propagates Label Constraints through Junctions

Now you are ready to learn about **Waltz's procedure**, a powerful procedure for propagating symbolic constraints. To see how Waltz's procedure works, first consider the drawing-labeling problem abstractly, in figure 12.19, without getting into the details of the actual labels. Think of keeping piles of label possibilities for each junction. These piles are created when a junction is visited for the first time, and they are reexamined each time an adjacent junction pile is altered.

In the illustration, junction A is the first junction visited, so you pile on A all of the label possibilities allowed from the junction catalog, as shown in the upper left of figure 12.19.

Now suppose junction B is the next junction visited. Again, you pile on junction labels, but the total set is reduced immediately to those that

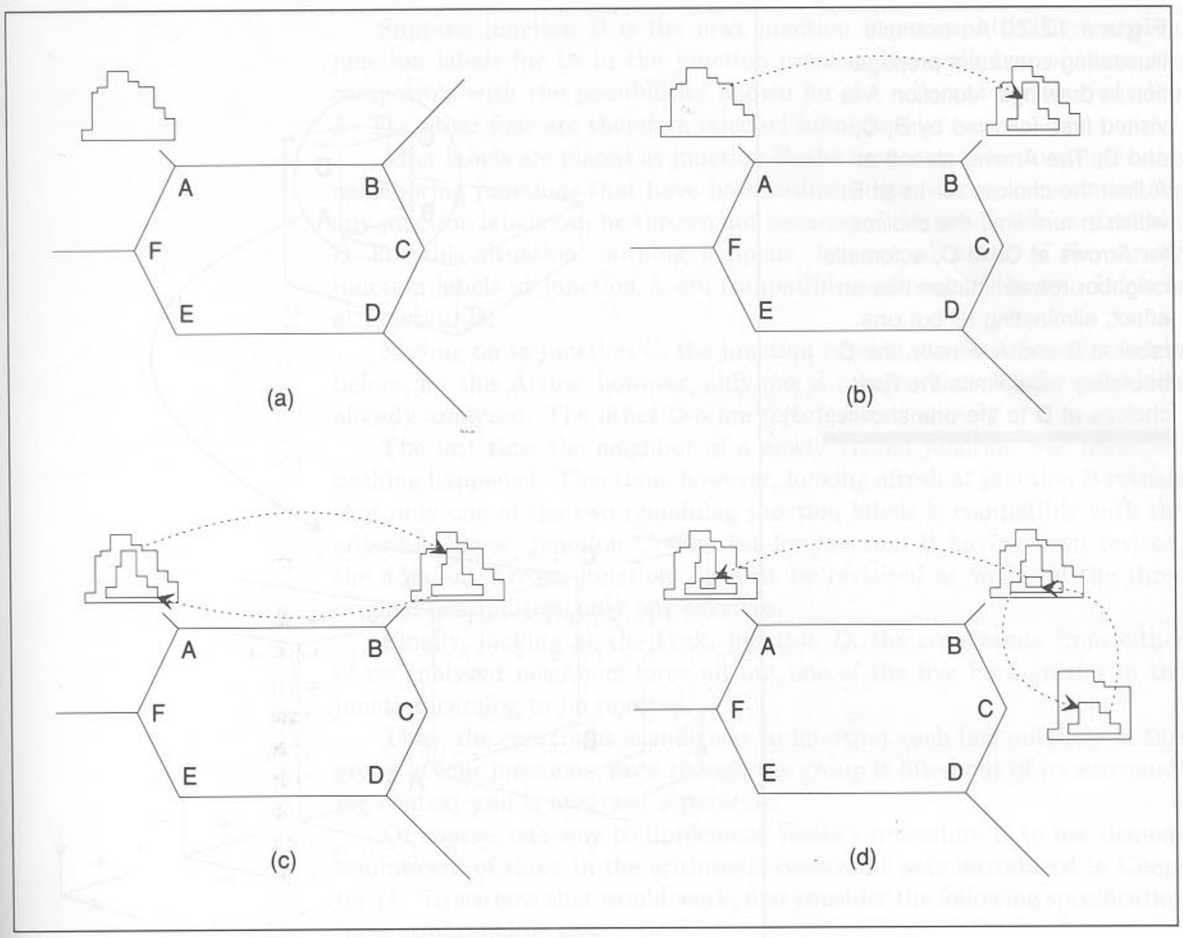


Figure 12.19 Label propagation in networks. At first, an initial junction pile is placed at an arbitrary junction. Propagation continues as long as reduction occurs at each junction pile encountered.

are compatible with at least one junction in the piles of all neighboring junctions with piles. In figure 12.19, junction A's label pile constrains what can be in junction B's pile.

Once a pile is created and has been reduced by neighboring piles, it is time to see whether those same neighboring piles contain junction labels that are incompatible with every junction label at the newly installed pile. In figure 12.19, the reduced pile at junction B constrains what can be in the pile at junction A.

Once set in motion, constraint propagation continues as long as the junction piles encountered continue to be reduced. In figure 12.19, for example, after the junction pile at junction C is installed and reduced, the outward-moving part of the process starts, travels through the pile at junction B, and terminates at the pile at junction A.

If there were no change at junction B, the propagation initiated at junction C would have terminated at junction B. On the other hand, if there

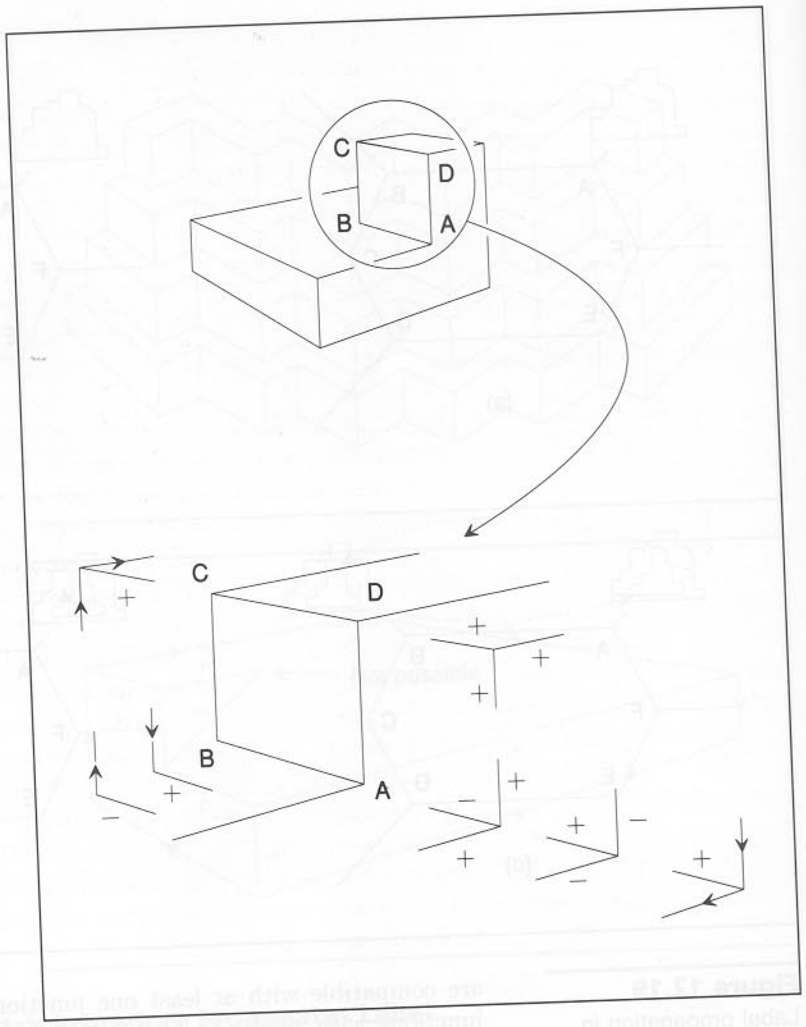


Figure 12.20 An example illustrating constraint propagation in drawings. Junction A is visited first, followed by B, C, and D. The Arrows placed at A limit the choices for Ls at B, which in turn limit the choices for Arrows at C. At C, automatic neighbor reexamination has an effect, eliminating all but one label at B and A. Finally, the C boundary label limits the Fork choices at D to the one shown.

already were label piles at all junctions in the illustration, a pile reduction at junction C could initiate a propagation series that would travel all the way around the loop, reduce the piles at each junction, and ultimately lead to further reduction at junction C, the junction where the round-the-loop propagation was initiated. Looping cannot continue forever, however.

Now let us move from the abstract to the concrete. Look at figure 12.20, in which two Arrows, a Fork, and an L are buried in a drawing. Suppose, further, that these four junctions are lifted out of the drawing and are analyzed with the plus, minus, and boundary line labels.

If junction A is the first junction visited and none of its neighbors have been visited, then the first step is to bring in all the possible Arrow junction labels, piling them on junction A.

Suppose junction B is the next junction investigated. There are six junction labels for Ls in the junction catalog, but only two of these are compatible with the possibilities known for the adjacent Arrow, junction A. The other four are therefore rejected immediately.

After labels are placed at junction B, the next step is to investigate the neighboring junctions that have been examined previously to see whether any junction labels can be thrown out because of the new labels at junction B. For this situation, nothing happens, because all three of the Arrow junction labels at junction A are compatible with one of the two L labels at junction B.

Moving on to junction C, the junction catalog supplies three entries as before; for this Arrow, however, only one is compatible with the neighbors already analyzed. The other two are rejected immediately.

The last time the neighbor of a newly visited junction was revisited, nothing happened. This time, however, looking afresh at junction B reveals that only one of the two remaining junction labels is compatible with the adjacent Arrow, junction C. The list for junction B having been revised, the adjacent Arrow, junction A, must be revisited as well. Of the three original possibilities, only one survives.

Finally, looking at the Fork, junction D, the constraints from either of its analyzed neighbors force all but one of the five Fork entries in the junction catalog to be rejected.

Thus, the constraint is sufficient to interpret each line uniquely in this group of four junctions, even though the group is lifted out of its surrounding context and is analyzed separately.

Of course, one way to implement Waltz's procedure is to use demons reminiscent of those in the arithmetic constraint nets introduced in Chapter 11. To see how that would work, first consider the following specification for a **contraction net**.

A **contraction net** is a representation
That is a frame system
In which

▷ Lexically and structurally, certain frame classes identify a finite list of application-specific interpretations.

▷ Procedurally, demon procedures enforce compatibility constraints among connected frames.

Starting with contraction nets, it is easy to specify a **labeled drawing**. Here is one such specification—one that happens to be limited to the original four line labels:

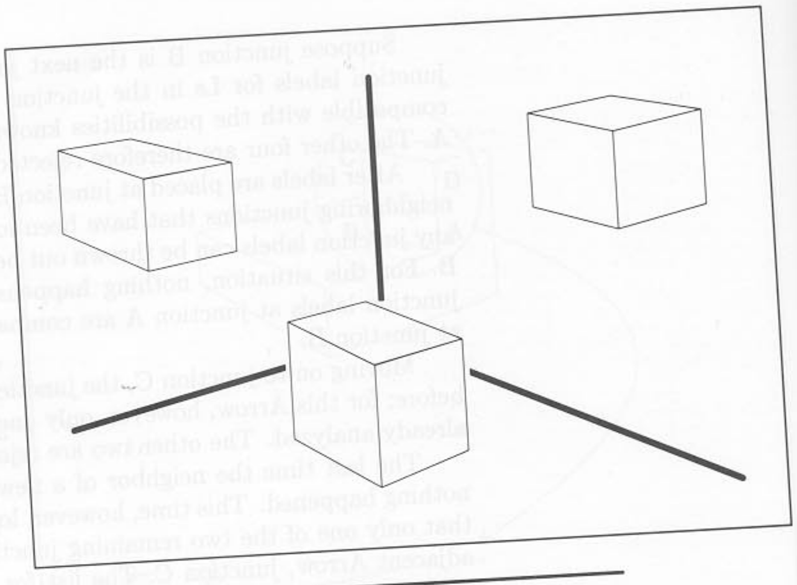


Figure 12.21 Without shadows, there are several ways to interpret a cube: It may be suspended, it may be supported by the floor, or it may be attached to a wall.

A **labeled drawing** is a representation
That is a contraction net
In which
- ▷ Lexically, there are line frames and junction frames. Lines may be convex, concave, or boundary lines. Junctions may be L, Fork, T, or Arrow junctions.
- ▷ Structurally, junction frames are connected by line frames. Also, each junction frame contains a list of interpretation combinations for its connecting lines.
- ▷ Semantically, line frames denote physical edges. Junction frames denote physical vertexes.
- ▷ Procedurally, demon procedures enforce the constraint that each junction label must be compatible with at least one of the junction labels at each of the neighboring junctions.

Many Line and Junction Labels Are Needed to Handle Shadows and Cracks

So far, by assumption, all the examples involve objects that are hanging suspended in space. If a cube is resting on a table, however, the bottom lines represent concave edges; they do not represent boundaries. If a cube is stuck against a wall, as figure 12.21 shows, other lines represent concave edges. Without an additional clue or assumption, several interpretations are equally plausible.

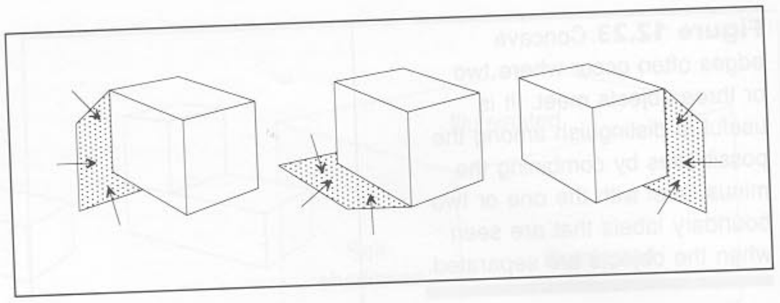


Figure 12.22 Shadows help determine where an object rests against others.

Note, however, that introducing shadows resolves the ambiguity. The block in the middle of figure 12.22 definitely seems supported by a horizontal surface, whereas the ones to the left and right, although less familiar, seem attached vertically. Evidently, expanding labeling theory to include labels for shadows should add further constraint and reduce ambiguity.

Take note that the shadow labels introduced in figure 12.22 indicate a direction, just as boundary labels do: shadow labels are small arrows placed so that they point into the shadowed region.

Now let us reconsider concave lines. Because concave lines are often found where objects come together, the concave-label category can be split into subcategories, indicating the number of objects involved and identifying which object is in front. Suppose a concave edge represents a place where two objects come together. Then, imagine pulling apart the two objects slightly. The concave edge becomes a boundary edge with the label pointing in one of two possible directions, as shown on the left and in the middle of figure 12.23. The two possibilities are indicated by compound symbols made up of the original minus label and the new boundary label. If, by chance, there are three objects, again a compound symbol is used, reflecting what is seen when the objects are pulled apart, as shown on the right in figure 12.23.

Cracks lying between two objects can be treated analogously: Each crack is labeled with a *c*, together with a boundary label that indicates how the two objects involved fit together. With cracks between objects allowed, you have the possibilities shown in figure 12.24. There are now 11 ways that any particular line may be labeled.

Illumination Increases Label Count and Tightens Constraint

The illumination on any face of an object can be classified, as shown in figure 12.25, as directly illuminated, shadowed by another object, or shadowed because it faces away from the light. The three possibilities are denoted by I, for directly <u>i</u>lluminated, S for <u>s</u>hadowed by another object, and SS for facing away from the light—that is, <u>s</u>elf-<u>s</u>hadowed.

Line labels can carry knowledge about these illumination states in addition to information about edge type. If the illumination states and line interpretations were to combine freely, there would be $3^2 = 9$ illumination

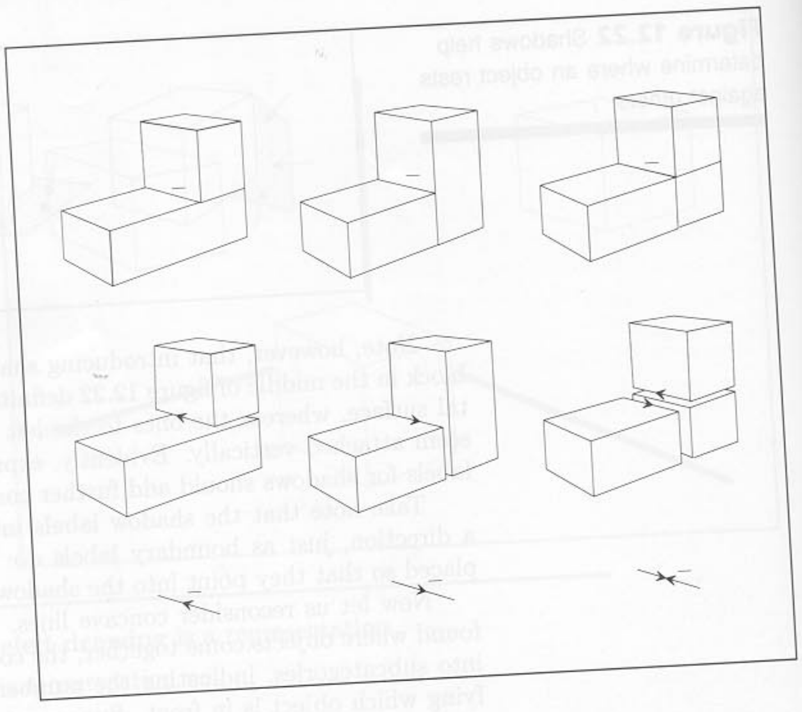


Figure 12.23 Concave edges often occur where two or three objects meet. It is useful to distinguish among the possibilities by combining the minus label with the one or two boundary labels that are seen when the objects are separated.

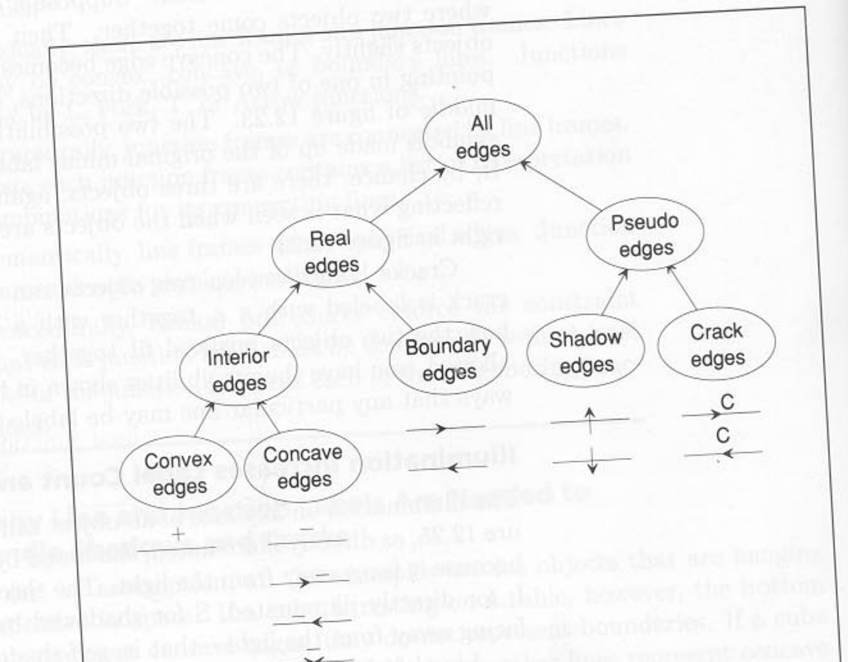


Figure 12.24 The eleven line interpretations and the corresponding labels.

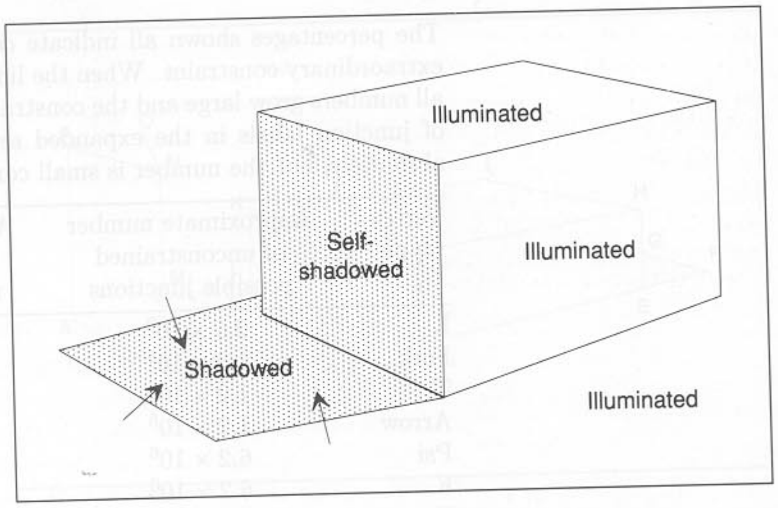


Figure 12.25 Illumination information often provides useful constraint. If there is a single light source, it is convenient to recognize three surface categories: illuminated; shadowed by intervening objects; and self-shadowed by virtue of facing away from the light source.

combinations for each of the 11 line interpretations, giving 99 total possibilities. Only about 50 of these combinations, however, are possible. Some combinations are forbidden because they would require an incredible coincidence, like the projection of a shadow line exactly onto a concave edge. Other combinations are excluded by definition; there cannot be, for example, a combination in which both sides of a shadow line are illuminated.

Now, let us review. Initially, only basic lines were considered: boundary lines, and interior concave and convex lines. Then, shadows were added. Concave lines were split up to reflect the number of objects coming together and the way those objects obscure one another. Cracks between objects were introduced and handled analogously. Finally, line information was combined with illumination information. Just over 50 line labels emerge from this final expansion.

These changes make the number of physically realizable junctions large, both for the original Fork, Arrow, L, and T types, and for other vertex types allowed by relaxing the three-faced-vertexes and general position constraints. What is gained in return for this increased number?

First, consider how the set of physically realizable junction labels compares to that of the unconstrained junction labels. The following table gives the results for the original set of line labels:

Vertex type	Number of unconstrained possible junctions	Number of physically realizable junctions	Ratio (%)
L	16	6	37.5
Fork	64	5	7.8
T	64	4	6.2
Arrow	64	3	4.7

The percentages shown all indicate constraint, but they do not indicate extraordinary constraint. When the line categories are expanded, however, all numbers grow large and the constraint becomes incredible. The number of junction labels in the expanded set, known as **Waltz's set**, is large absolutely, but the number is small compared with what it might be:

Vertex type	Approximate number of unconstrained possible junctions	Approximate number of physically realizable junctions	Ratio (%)
L	2.5×10^3	80	3.2
Fork	1.2×10^5	500	4.0×10^{-1}
T	1.2×10^5	500	4.0×10^{-1}
Arrow	1.2×10^5	70	5.6×10^{-2}
Psi	6.2×10^6	300	4.8×10^{-3}
K	6.2×10^6	100	1.6×10^{-3}
X	6.2×10^6	100	1.6×10^{-3}
Multi	6.2×10^6	100	1.6×10^{-3}
Peak	6.2×10^6	10	1.6×10^{-4}
Kk	3.1×10^8	30	9.6×10^{-6}

Figure 12.4 shows what all these junction types look like. For the Kk junction, only about one junction label in 10 million is physically realizable. To be sure, the total number of labels has increased to a size too large to use by hand, but still the constraints are so extreme that a computer program using the large set can converge on an unambiguous solution.

In this progression from small set, large fraction to large set, small fraction, you can observe the following powerful idea at work:

The **describe-to-explain principle**:

▷ The act of detailed description may turn probabilistic regularities into entirely deterministic constraints.

The Flow of Labels Can Be Dramatic

Watching a film is the best way to appreciate what can happen when Waltz's set, instead of the dwarfish set for the three-faced vertex world, is used to label a drawing.

Lacking a film, glancing at the drawing in figure 12.26 and at the trace in table 1 provides some feel for how the Waltz procedure works with Waltz's set. It would be tedious to follow the trace in detail, but some overall points are obvious without much effort.

In each of the 80 steps, the step number is followed by a letter denoting the junction involved. The letter is followed by the old number of junction labels in the pile and the new number in the pile.

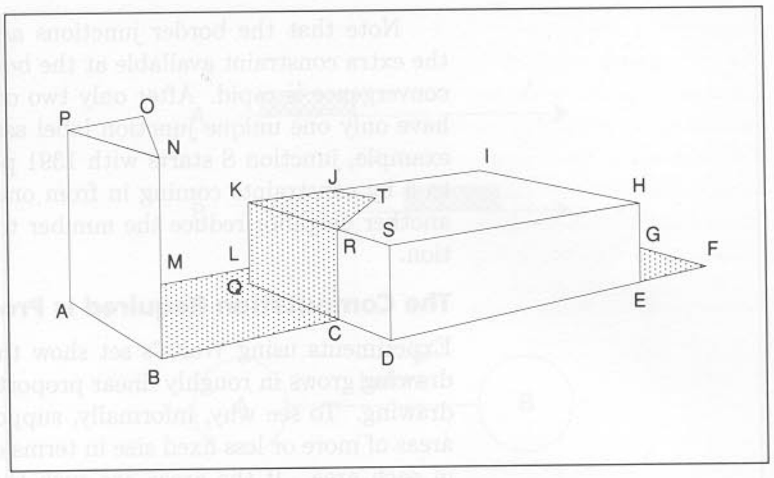


Figure 12.26 A drawing analyzed successfully by Waltz's labeling procedure.

Table 1. A trace of Waltz's labeling procedure in action.

1	A	–	→	123	31	C	20	→	2	61	P	5	→	1
2	A	123	→	76	32	B	14	→	2	62	O	2	→	1
3	B	–	→	79	33	A	20	→	2	63	Q	–	→	123
4	B	79	→	52	34	H	–	→	79	64	Q	123	→	5
5	A	76	→	32	35	H	79	→	1	65	Q	5	→	1
6	C	–	→	388	36	I	–	→	123	66	L	3	→	1
7	C	388	→	78	37	I	123	→	2	67	K	5	→	2
8	B	52	→	34	38	J	–	→	593	68	R	–	→	388
9	A	32	→	26	39	J	593	→	9	69	R	388	→	20
10	D	–	→	79	40	K	–	→	79	70	R	20	→	4
11	D	79	→	15	41	K	79	→	7	71	J	6	→	4
12	C	78	→	20	42	J	9	→	8	72	K	2	→	1
13	B	34	→	14	43	L	–	→	593	73	S	–	→	1391
14	A	26	→	20	44	L	593	→	12	74	S	1391	→	5
15	E	–	→	79	45	K	7	→	5	75	S	5	→	1
16	E	79	→	33	46	J	8	→	6	76	T	–	→	123
17	D	15	→	14	47	I	2	→	1	77	T	123	→	4
18	F	–	→	123	48	M	–	→	593	78	T	4	→	1
19	F	123	→	28	49	M	593	→	4	79	J	4	→	1
20	G	–	→	593	50	M	4	→	1	80	R	4	→	1
21	G	593	→	42	51	L	12	→	3					
22	G	42	→	18	52	B	2	→	1					
23	F	28	→	11	53	A	2	→	1					
24	E	33	→	7	54	C	2	→	1					
25	D	14	→	6	55	N	–	→	79					
26	F	11	→	2	56	N	79	→	1					
27	G	18	→	1	57	O	–	→	123					
28	E	7	→	1	58	O	123	→	2					
29	D	6	→	1	59	P	–	→	79					
30	F	2	→	1	60	P	79	→	5					

Note that the border junctions are visited first. This order exploits the extra constraint available at the border. Further inspection shows that convergence is rapid. After only two or three visits, most of the junctions have only one unique junction label associated with them. In step 74, for example, junction S starts with 1391 possibilities. The number is reduced to 5 by constraints coming in from one neighbor. Then, constraints from another neighbor reduce the number to 1, leaving but a single interpretation.

The Computation Required Is Proportional to Drawing Size

Experiments using Waltz's set show that the work required to analyze a drawing grows in roughly linear proportion with the number of lines in the drawing. To see why, informally, suppose that drawings can be split into areas of more or less fixed size in terms of the lines and junctions contained in each area. If the areas are such that constraint does not flow across their frontiers, then the total time required to analyze a drawing is linearly proportional to the number of areas, and hence is linearly proportional to the number of junctions.

Flow-impeding frontiers exist because the T junctions, common at object boundaries, have little ability to transmit constraint: An obscuring boundary can lie in front of any kind of edge.

PROPAGATION OF TIME-INTERVAL RELATIONS

In this section, you learn about another example of symbolic constraint propagation; this one involves time intervals and the relations among time intervals. The general idea is to use existing information about the relations among time intervals to reach conclusions about other relations. For example, if interval A is before interval B, and interval B is before interval C, then plainly interval A has to be before C.

Time relations are easier to understand visually, especially as they become more complicated. As shown in the upper part of figure 12.27, time intervals can be depicted as objects resting on a time line. Alternatively, as shown in the lower part of figure 12.27, a pair of time intervals and the relation that lies between them can be depicted as two nodes, and a labeled link.

There Are 13 Ways to Label a Link between Interval Nodes Yielding 169 Constraints

Assuming that no interval is allowed to start and stop at the same time, then the interval *before* relation is just one of 13 possible relations between two intervals, seven of which are shown in figure 12.28. The other six are members of symmetric pairs. One such symmetric pair is the *before* pair, because interval A may be before interval B or interval B may be before interval A.

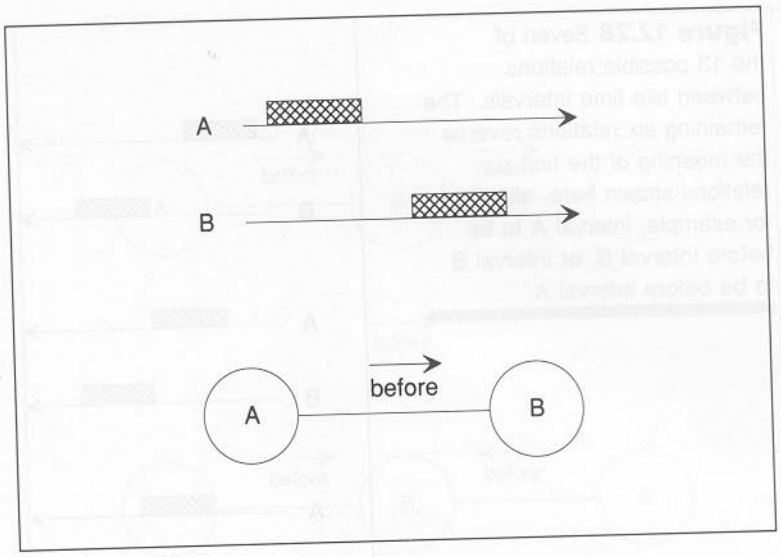


Figure 12.27 Two time intervals and the relation between them. In the upper part, two time intervals are shown as objects resting on a time line, with the object representing interval A preceding the one for interval B. In the lower part, the two time intervals are shown as nodes, and the relation between them is expressed as a link label.

Notationally, an arrow is drawn over each label to indicate the direction in which a relation is to be read. Thus, A $\overrightarrow{\text{before}}$ B means interval A is before interval B, whereas A $\overleftarrow{\text{before}}$ B means interval B is before interval A. Attaching arrows to labels is necessary, rather than making the links directional, because several labels, with arrows in opposite directions, may be attached to one link.

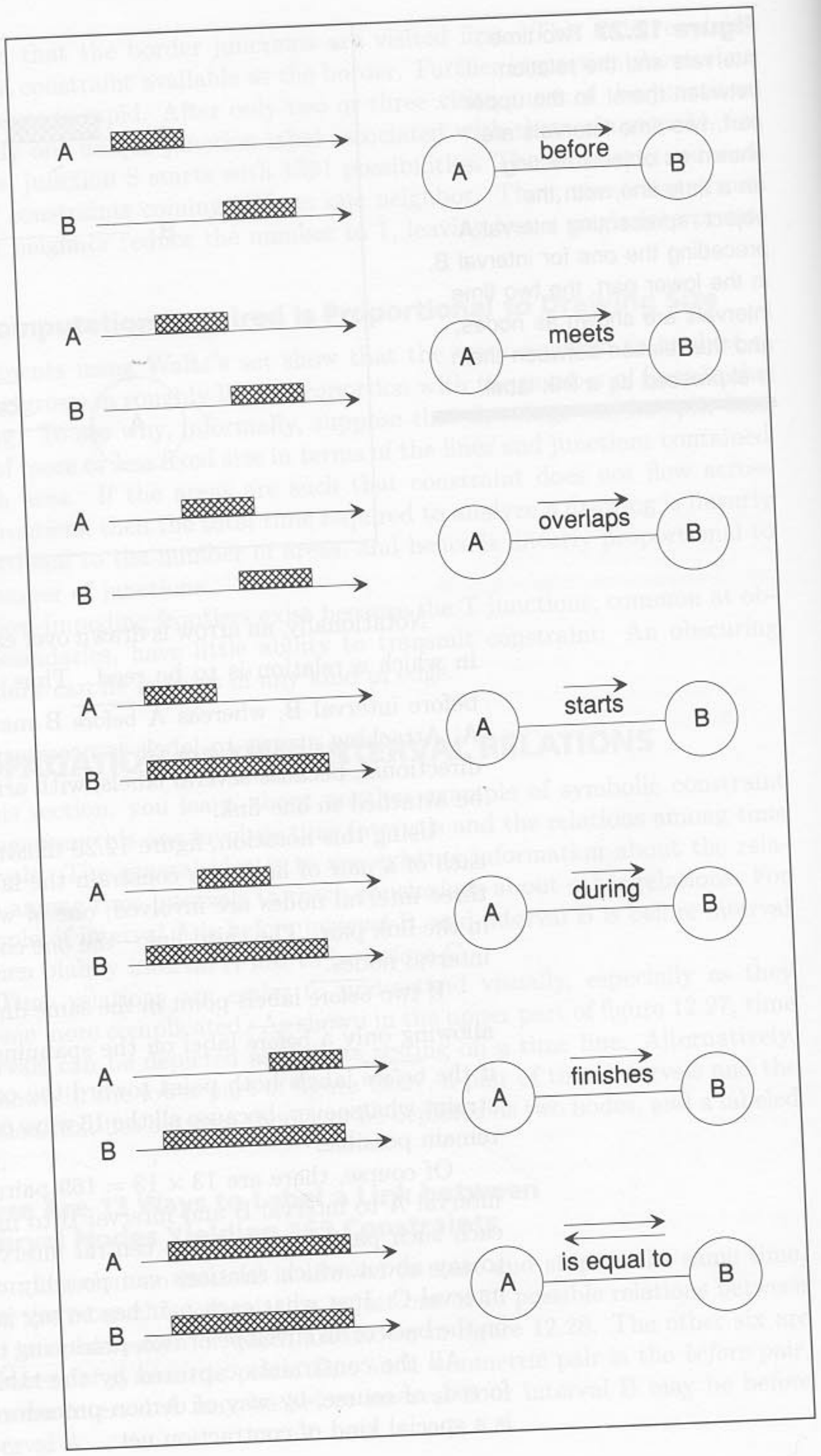
Using this notation, figure 12.29 illustrates the idea that the labels on each of a pair of links can constrain the labels on a third link. As shown, three interval nodes are involved, one of which lies between the two links in the link pair. The third link—the one constrained—spans the other two interval nodes.

If two $\overrightarrow{\text{before}}$ labels point in the same direction, the constraint is severe, allowing only a $\overrightarrow{\text{before}}$ label on the spanning relation. Curiously, however, if the $\overrightarrow{\text{before}}$ labels both point toward the central interval, there is no constraint whatsoever, because all the 13 ways of labeling the spanning relation remain possible.

Of course, there are $13 \times 13 = 169$ pairs of relations that can connect interval A to interval B and interval B to interval C. The key idea is that each such pair, joined at the central interval, interval B, has something to say about which relations can possibly connect interval A directly to interval C. Just what each pair has to say is something you can work out on the back of an envelope or two, producing the table shown in figure 12.30.

All the constraints captured by the tables in figure 12.30 can be enforced, of course, by way of demon procedures in an **interval net**, which is a special kind of contraction net:

Figure 12.28 Seven of the 13 possible relations between two time intervals. The remaining six relations reverse the meaning of the first six relations shown here, allowing, for example, interval A to be before interval B, or interval B to be before interval A.



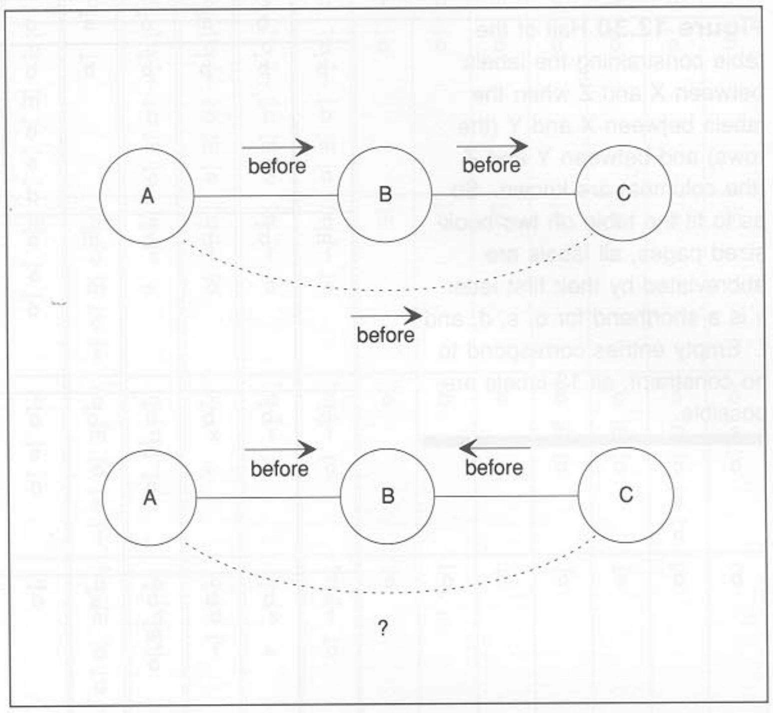


Figure 12.29 In the top half, interval A is before interval B, and interval B is before interval C. These relations force the conclusion that interval A is before interval C. On the other hand, as shown in the bottom half, nothing can be concluded about the relation between interval A and interval C given only that interval A is before interval B and interval C is before interval B.

An **interval net** is a representation
That is a contraction net
In which

▷ Lexically and semantically there are interval frames denoting time intervals and link frames denoting time relations, specifically $\overrightarrow{\text{before}}$, $\overleftarrow{\text{before}}$, $\overrightarrow{\text{during}}$, $\overleftarrow{\text{during}}$, $\overrightarrow{\text{overlaps}}$, $\overleftarrow{\text{overlaps}}$, $\overrightarrow{\text{meets}}$, $\overleftarrow{\text{meets}}$, $\overrightarrow{\text{starts}}$, $\overleftarrow{\text{starts}}$, $\overrightarrow{\text{finishes}}$, $\overleftarrow{\text{finishes}}$, and $\overrightarrow{\overleftarrow{\text{is equal to}}}$.

▷ Structurally, interval frames are connected by link frames.

▷ Procedurally, demon procedures enforce the constraint that the interpretations allowed for a link frame between two intervals must be consistent with the interpretations allowed for the two link frames joining the two intervals to a third interval.

Time Constraints Can Propagate across Long Distances

Given a chain of relations, it may be possible to use the table in figure 12.30 to reach a long-distance conclusion about the relation between two widely

Figure 12.30 Half of the table constraining the labels between X and Z when the labels between X and Y (the rows) and between Y and Z (the columns) are known. So as to fit the table on two book-sized pages, all labels are abbreviated by their first letter. x is a shorthand for o, s, d, and f. Empty entries correspond to no constraint; all 13 labels are possible.

	$\overrightarrow{b}$	$\overrightarrow{m}$	$\overrightarrow{o}$	$\overrightarrow{s}$	$\overrightarrow{d}$	$\overrightarrow{f}$	$\overleftarrow{b}$	$\overleftarrow{m}$	$\overleftarrow{o}$	$\overleftarrow{s}$	$\overleftarrow{d}$	$\overleftarrow{f}$
$\overrightarrow{b}$	$\overrightarrow{b}$	$\overrightarrow{b}$	$\overrightarrow{b}$	$\overrightarrow{b}$	$\overrightarrow{b}\ \overrightarrow{m}\ \overrightarrow{o}\ \overrightarrow{s}\ \overrightarrow{d}$	$\overrightarrow{b}\ \overrightarrow{m}\ \overrightarrow{o}\ \overrightarrow{s}\ \overrightarrow{d}$		$\overrightarrow{b}\ \overrightarrow{m}\ \overrightarrow{o}\ \overrightarrow{s}\ \overrightarrow{d}$	$\overrightarrow{b}\ \overrightarrow{m}\ \overrightarrow{o}\ \overrightarrow{s}\ \overrightarrow{d}$	$\overrightarrow{b}$	$\overrightarrow{b}$	$\overrightarrow{b}$
$\overrightarrow{m}$	$\overrightarrow{b}$	$\overrightarrow{b}$	$\overrightarrow{b}$	$\overrightarrow{m}$	$\overrightarrow{o}\ \overrightarrow{s}\ \overrightarrow{d}$	$\overrightarrow{o}\ \overrightarrow{s}\ \overrightarrow{d}$	$\overleftarrow{b}\ \overleftarrow{m}\ \overleftarrow{o}\ \overleftarrow{s}\ \overleftarrow{d}$	$\overrightarrow{f}\ \overleftarrow{f}\ =$	$\overrightarrow{o}\ \overrightarrow{s}\ \overrightarrow{d}$	$\overrightarrow{m}$	$\overrightarrow{b}$	$\overrightarrow{b}$
$\overrightarrow{o}$	$\overrightarrow{b}$	$\overrightarrow{b}$	$\overrightarrow{b}\ \overrightarrow{m}\ \overrightarrow{o}$	$\overrightarrow{o}$	$\overrightarrow{o}\ \overrightarrow{s}\ \overrightarrow{d}$	$\overrightarrow{o}\ \overrightarrow{s}\ \overrightarrow{d}$	$\overleftarrow{b}\ \overleftarrow{m}\ \overleftarrow{o}\ \overleftarrow{s}\ \overleftarrow{d}$	$\overleftarrow{o}\ \overleftarrow{s}\ \overleftarrow{d}$	$\overrightarrow{x}\ \overleftarrow{x}\ =$	$\overrightarrow{o}\ \overleftarrow{d}\ \overleftarrow{f}$	$\overrightarrow{b}\ \overrightarrow{m}\ \overrightarrow{o}\ \overleftarrow{d}\ \overleftarrow{f}$	$\overrightarrow{b}\ \overrightarrow{m}\ \overrightarrow{o}$
$\overrightarrow{s}$	$\overrightarrow{b}$	$\overrightarrow{b}$	$\overrightarrow{b}\ \overrightarrow{m}\ \overrightarrow{o}$	$\overrightarrow{s}$	$\overrightarrow{d}$	$\overrightarrow{d}$	$\overleftarrow{b}$	$\overleftarrow{m}$	$\overrightarrow{d}\ \overrightarrow{f}\ \overleftarrow{o}$	$\overrightarrow{s}\ \overleftarrow{s}\ =$	$\overrightarrow{b}\ \overrightarrow{m}\ \overrightarrow{o}\ \overleftarrow{d}\ \overleftarrow{f}$	$\overrightarrow{b}\ \overrightarrow{m}\ \overrightarrow{o}$
$\overrightarrow{d}$	$\overrightarrow{b}$	$\overrightarrow{b}$	$\overrightarrow{b}\ \overrightarrow{m}\ \overrightarrow{o}\ \overrightarrow{s}\ \overrightarrow{d}$	$\overrightarrow{d}$	$\overrightarrow{d}$	$\overrightarrow{d}$	$\overleftarrow{b}$	$\overleftarrow{b}$	$\overrightarrow{d}\ \overrightarrow{f}\ \overleftarrow{b}\ \overleftarrow{m}\ \overleftarrow{o}$	$\overrightarrow{d}\ \overrightarrow{f}\ \overleftarrow{b}\ \overleftarrow{m}\ \overleftarrow{o}$		$\overrightarrow{b}\ \overrightarrow{m}\ \overrightarrow{o}\ \overrightarrow{s}\ \overrightarrow{d}$
$\overrightarrow{f}$	$\overrightarrow{b}$	$\overrightarrow{m}$	$\overrightarrow{o}\ \overrightarrow{s}\ \overrightarrow{d}$	$\overrightarrow{d}$	$\overrightarrow{d}$	$\overrightarrow{f}$	$\overleftarrow{b}$	$\overleftarrow{b}$	$\overleftarrow{b}\ \overleftarrow{m}\ \overleftarrow{o}$	$\overleftarrow{b}\ \overleftarrow{m}\ \overleftarrow{o}$	$\overleftarrow{b}\ \overleftarrow{m}\ \overleftarrow{o}\ \overleftarrow{s}\ \overleftarrow{d}$	$\overrightarrow{f}\ \overleftarrow{f}\ =$

separated intervals. Figure 12.31 shows how. First, the pair of relations connecting interval A, interval B, and interval C enable a conclusion about which relations can connect interval A to interval C.

Next, the relation that connects interval A to interval C is used in concert with the relation that connects interval C to D to determine which relations can connect interval A to D. Continued iteration may eventually provide some constraint on the relation between interval A and Z.

Figure 12.30 Continued. The other half of the table constraining the labels between X and Z when the labels between X and Y (the rows) and between Y and Z (the columns) are known.

	$\overrightarrow{b}$	$\overrightarrow{m}$	$\overrightarrow{o}$	$\overrightarrow{s}$	$\overrightarrow{d}$	$\overrightarrow{f}$	$\overleftarrow{b}$	$\overleftarrow{m}$	$\overleftarrow{o}$	$\overleftarrow{s}$	$\overleftarrow{d}$	$\overleftarrow{f}$
$\overleftarrow{b}$		$\overrightarrow{d}$ $\overrightarrow{f}$ $\overleftarrow{b}$ $\overleftarrow{m}$ $\overleftarrow{o}$	$\overrightarrow{d}$ $\overrightarrow{f}$ $\overleftarrow{b}$ $\overleftarrow{m}$ $\overleftarrow{o}$	$\overrightarrow{d}$ $\overrightarrow{f}$ $\overleftarrow{b}$ $\overleftarrow{m}$ $\overleftarrow{o}$	$\overrightarrow{d}$ $\overrightarrow{f}$ $\overleftarrow{b}$ $\overleftarrow{m}$ $\overleftarrow{o}$	$\overleftarrow{b}$	$\overleftarrow{b}$	$\overleftarrow{b}$	$\overleftarrow{b}$	$\overleftarrow{b}$	$\overleftarrow{b}$	$\overleftarrow{b}$
$\overleftarrow{m}$	$\overrightarrow{b}$ $\overrightarrow{o}$ $\overrightarrow{m}$ $\overleftarrow{d}$ $\overleftarrow{f}$	$\overrightarrow{s}$ $\overleftarrow{s}$ =	$\overrightarrow{d}$ $\overrightarrow{f}$ $\overleftarrow{o}$	$\overrightarrow{d}$ $\overrightarrow{f}$ $\overleftarrow{o}$	$\overrightarrow{d}$ $\overrightarrow{f}$ $\overleftarrow{o}$	$\overleftarrow{m}$	$\overleftarrow{b}$	$\overleftarrow{b}$	$\overleftarrow{b}$	$\overleftarrow{b}$	$\overleftarrow{b}$	$\overleftarrow{m}$
$\overleftarrow{o}$	$\overrightarrow{b}$ $\overrightarrow{m}$ $\overrightarrow{o}$ $\overleftarrow{d}$ $\overleftarrow{f}$	$\overrightarrow{o}$ $\overleftarrow{d}$ $\overleftarrow{f}$	$\overrightarrow{x}$ $\overleftarrow{x}$ =	$\overrightarrow{d}$ $\overrightarrow{f}$ $\overleftarrow{o}$	$\overrightarrow{d}$ $\overrightarrow{f}$ $\overleftarrow{o}$	$\overleftarrow{o}$	$\overleftarrow{b}$	$\overleftarrow{b}$	$\overleftarrow{b}$ $\overleftarrow{m}$ $\overleftarrow{o}$	$\overleftarrow{b}$ $\overleftarrow{m}$ $\overleftarrow{o}$	$\overleftarrow{b}$ $\overrightarrow{m}$ $\overleftarrow{o}$ $\overleftarrow{s}$ $\overleftarrow{d}$	$\overleftarrow{o}$ $\overleftarrow{s}$ $\overleftarrow{d}$
$\overleftarrow{s}$	$\overrightarrow{b}$ $\overrightarrow{m}$ $\overrightarrow{o}$ $\overleftarrow{d}$ $\overleftarrow{f}$	$\overrightarrow{o}$ $\overleftarrow{d}$ $\overleftarrow{f}$	$\overrightarrow{o}$ $\overleftarrow{d}$ $\overleftarrow{f}$	$\overrightarrow{s}$ $\overleftarrow{s}$ =	$\overrightarrow{d}$ $\overrightarrow{f}$ $\overleftarrow{o}$	$\overleftarrow{o}$	$\overleftarrow{b}$	$\overleftarrow{m}$	$\overleftarrow{o}$	$\overleftarrow{s}$	$\overleftarrow{d}$	$\overleftarrow{d}$
$\overleftarrow{d}$	$\overrightarrow{b}$ $\overrightarrow{m}$ $\overrightarrow{o}$ $\overleftarrow{d}$ $\overleftarrow{f}$	$\overrightarrow{o}$ $\overleftarrow{d}$ $\overleftarrow{f}$	$\overrightarrow{o}$ $\overleftarrow{d}$ $\overleftarrow{f}$	$\overrightarrow{o}$ $\overleftarrow{d}$ $\overleftarrow{f}$	$\overrightarrow{x}$ $\overleftarrow{x}$ =	$\overleftarrow{o}$ $\overleftarrow{d}$ $\overleftarrow{s}$	$\overleftarrow{b}$ $\overleftarrow{m}$ $\overleftarrow{o}$ $\overleftarrow{d}$ $\overleftarrow{s}$	$\overleftarrow{o}$ $\overleftarrow{s}$ $\overleftarrow{d}$	$\overleftarrow{o}$ $\overleftarrow{s}$ $\overleftarrow{d}$	$\overleftarrow{d}$	$\overleftarrow{d}$	$\overleftarrow{d}$
$\overleftarrow{f}$	$\overrightarrow{b}$	$\overrightarrow{m}$	$\overrightarrow{o}$	$\overrightarrow{o}$	$\overrightarrow{o}$ $\overrightarrow{s}$ $\overrightarrow{d}$	$\overrightarrow{f}$ $\overleftarrow{f}$ =	$\overleftarrow{b}$ $\overleftarrow{m}$ $\overleftarrow{o}$ $\overleftarrow{s}$ $\overleftarrow{d}$	$\overleftarrow{o}$ $\overleftarrow{s}$ $\overleftarrow{d}$	$\overleftarrow{o}$ $\overleftarrow{s}$ $\overleftarrow{d}$	$\overleftarrow{d}$	$\overleftarrow{d}$	$\overleftarrow{f}$

A Complete Time Analysis Is Computationally Expensive

Now suppose you want to do a complete analysis, ensuring that you have used whatever link labels are given to determine the absolute minimum number of labels at every link in a net of n time intervals.

Because you are free to place a link between any two nodes, if there are n nodes, there can be as many as $n-1$ links emerging from each. Avoiding double counting by dividing by two, there must be $\frac{n\times(n-1)}{2}$ links.

For each link between two nodes, there are $n-2$ other nodes that can serve as a central interval. Thus, there are $n-2$ link pairs that can, in

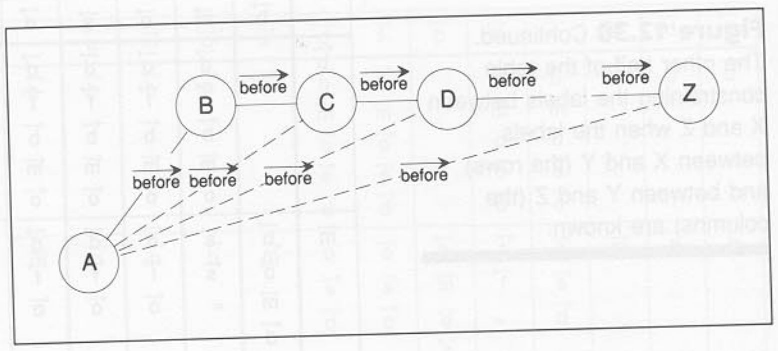


Figure 12.31 A chain of time-interval relations labeled $\overrightarrow{\text{before}}$ enables the placement of other $\overrightarrow{\text{before}}$ labels, including the placement of a $\overrightarrow{\text{before}}$ label between the ends of the chain.

principle, force a reduction in the number of labels on a link. Each time you cycle through all the links, you have to examine $\frac{n\times(n-1)}{2} \times (n-2)$ link pairs.

If no labels are eliminated during a cycle, you stop. But if any labels are eliminated, you have to continue.

In the worst case, only one label of one link is eliminated during a cycle. Because each of the $\frac{n\times(n-1)}{2}$ links may have as many as 13 labels, you may have to cycle through all the links, in the worst case, $13 \times \frac{n\times(n-1)}{2}$ times. Thus, the worst-case number of pair examinations is $13 \times \frac{n\times(n-1)}{2} \times \frac{n\times(n-1)}{2} \times (n-2)$, which is order n^5. This is not good if n is large.

Reference Nodes Can Save Time

Because a complete time analysis is computationally expensive, it may be important to single out some of the time-interval nodes to serve as *reference nodes.* When you want to determine the possible labels on a link between two given interval nodes, you start by looking for paths between the two nodes such that all links, except for the first and last, must connect reference node pairs. Then you collect all the nodes in all such paths and perform a complete analysis using those nodes.

Suppose, for example, that you single out interval nodes that represent the morning and the afternoon. Further suppose that each is connected to three other nodes as shown in figure 12.32.

Next, suppose you want to know the possible links between interval X, attached to the morning node, and interval Y, attached to the afternoon node. There is only one path between them through interval nodes, and that path happens to contain four nodes. Accordingly, you need to do a complete analysis of only a four-node, six-link net. Without reference nodes, you would have to do a complete analysis of an eight-node, 28-link net.

Of course, there is a price for confining your analysis to reference nodes: you may overlook a constraint now and then. Generally, however, with carefully selected reference intervals, little or nothing is lost.

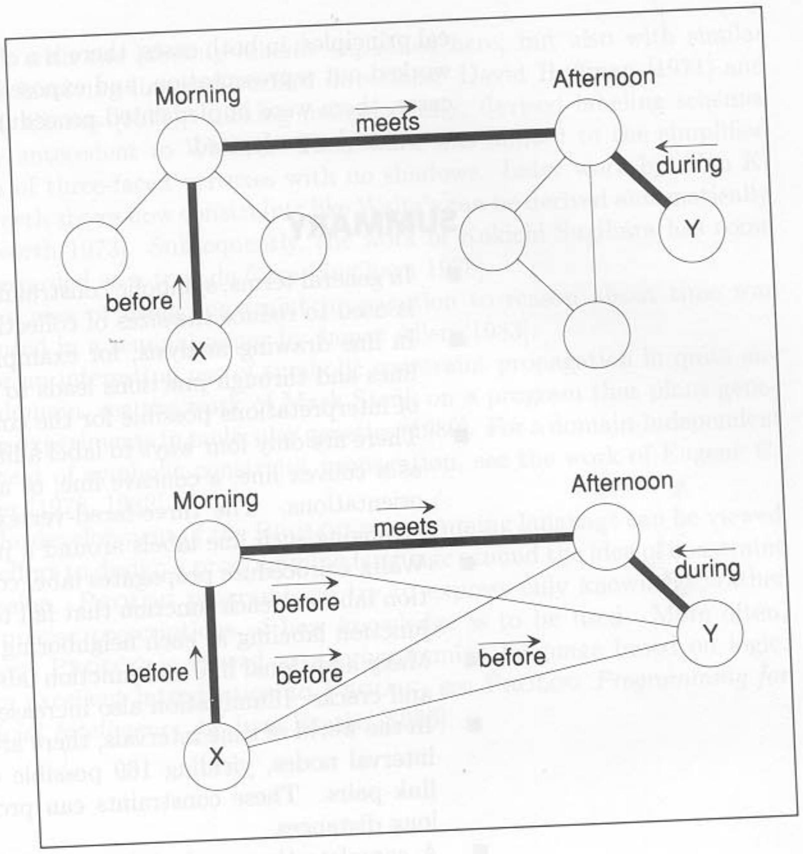


Figure 12.32 Reference nodes can reduce dramatically the amount of computation required to do constraint propagation. As shown in the upper part, only the nodes on reference-node paths are considered. As shown in the lower part, an analysis involving the four reference-path nodes quickly indicates that interval X is before interval Y.

FIVE POINTS OF METHODOLOGY

History shows that many of the methods of artificial intelligence are like sirens, seducing you into misapplication. Consequently, to keep yourself focused on solving problems, rather than on showing off particular methods, you should follow Marr's methodological principles—those championed by the late David Marr:

To follow **Marr's methodological principles**:

▷ First, *identify the problem.*

▷ Then, *select or develop an appropriate representation.*

▷ Next, *expose constraints or regularities.*

▷ Only then, *create particular procedures.*

▷ Finally, *verify via experiments.*

You can easily imagine that the original developers of drawing-analysis and interval-calculation procedures could have followed such methodological principles; in both cases, there is a clearly identified problem, a carefully worked-out representation, and exposed constraints. Furthermore, in both cases, there were implemented procedures that enabled convincing experiments to be performed.

SUMMARY

- In general terms, symbolic constraint boxes propagate information that is used to reduce the sizes of collections of possible interpretations.
- In line drawing analysis, for example, information propagating along lines and through junctions leads to a steady reduction in the number of interpretations possible for the connected junctions.
- There are only four ways to label a line in the three-faced-vertex world: as a convex line, a concave line, or a boundary line with two possible orientations. The three-faced-vertex world permits only 18 ways of arranging such line labels around a junction.
- Waltz's procedure propagates label constraints by eliminating all junction labels at each junction that fail to be compatible with at least one junction labeling at each neighboring junction.
- Many additional line and junction labels are needed to handle shadows and cracks. Illumination also increases label count.
- In the world of time intervals, there are 13 ways to label a link between interval nodes, yielding 169 possible constraints on links that bridge link pairs. These constraints can propagate time information across long distances.
- A complete time analysis is computationally expensive, but much of the expense can be avoided if reference nodes are used, albeit with some reduction in capability.
- The world manifests many constraints and regularities. For a computer to exhibit intelligence, it is necessary to exploit those constraints and regularities no matter of what the computer happens to be made.
- The act of detailed description may turn probabilistic regularities into deterministic constraints.
- Marr's approach specifies five steps. First, you identify the problem. Then, you select or develop an appropriate representation. Next, you expose constraints or regularities. After that, you create particular procedures. Finally, you verify via experiments.

BACKGROUND

Early ad hoc methods for line-drawing analysis, such as those developed by Adolfo Guzman [1968], stimulated much of the subsequent work on line-drawing analysis, including David Waltz's seminal thesis on drawing analysis, on which this chapter is largely based [Waltz 1972]. Waltz dealt not only with the labeling scheme explained here, but also with similar schemes involving line and surface directions. David Huffman [1971] and Maxwell Clowes [1971], working independently, devised labeling schemes directly antecedent to Waltz's. Their work was limited to the simplified domain of three-faced vertexes with no shadows. Later work by Alan K. Mackworth shows how constraints like Waltz's can be derived automatically [Mackworth 1973]. Subsequently, the work of Kokichi Sugihara has come to be regarded as a tour de force [Sugihara 1978].

The idea of using constraint propagation to reason about time was introduced in a seminal paper by James Allen [1983].

For an interesting use of symbolic constraint propagation in quite another domain, see the work of Mark Stefik on a program that plans gene-cloning experiments in molecular genetics [1980]. For a domain-independent treatment of symbolic constraint propagation, see the work of Eugene C. Freuder [1978, 1982].

The development of the PROLOG programming language can be viewed as an effort to design a programming language around the idea of constraint expression. PROLOG programmers try to express only knowledge, rather than precise prescriptions of how knowledge is to be used. More often, however, PROLOG is viewed as a programming language based on logic. For an excellent introduction to PROLOG, see PROLOG *Programming for Artificial Intelligence*, by Ivan Bratko [1986].