pieces that remain. Breaking the problem up prematurely, without applying massive computation to decide on the appropriate breakup, may lead to getting stuck.

Finally, evolution and the human mind work at a meta-level, in a different space than the algorithms of computer scientists do (see discussion of recursion in chapter 8). Problems like the TSP are addressed in problem space: an algorithm (devised by a computer scientist) searches over a particular problem instance for solutions. Evolution and the human mind work in program space: they evolve modular programs or call on such programs to devise the algorithms.

In chapter 8, I talked about recursion as an example of a module in the mind. I noted that recursion and other such concepts are modules for looking at problems and identifying not solutions to particular problems but algorithms for solving them.

The development of these modules is a hard problem, probably requiring some degree of hard search. This search uses large resources: the minds of many scientists over many decades and even millennia. Once found, a module can be passed on verbally. This is how computer science advances. In chapter 13, I discuss this impact of language on the evolution of mind at more length.

The search involved in developing a module such as recursion is a search for one module using existing modules. That is, the search can call many functions that one already knows how to compute. So what is necessary is incremental progress, which though it requires hard computation may be feasible. Evolution is works similarly. It works one step at a time to improve a program.

Evolution also makes use of some interesting tricks. For example, it has crafted the programs of our minds to learn during life. This is a kind of algorithm little studied by computer scientists.

In the next section, I discuss another reason that NP-complete problems arising in nature can often be solved extremely rapidly by exploiting compact structure.

## 11.4 Constraint Propagation

Constraint propagation is another trick that allows solution of many seemingly intractable problems and that seems likely to be critical to intelligence. Naturally arising problems often turn out to be sufficiently overconstrained so that doing the obvious thing almost always works. The best illustration of this is Avrim Blum's (1994) algorithm for graph 3-coloring.

Recall that a graph is simply a collection of nodes with a collection of edges that connect some pairs of nodes. A $k$-coloring of the graph is an assignment of one of $k$ colors to each of the nodes in such a way that no edge connects two nodes of the same color. So, for example, 3-coloring a graph is painting each of its nodes red,

white, or blue so that no edge connects two white nodes, two red nodes, or two blue nodes. For a large graph, finding a 3-coloring can be hard; graph 3-coloring is NP-complete. Moreover, coloring a graph with three colors seems harder than coloring one with four or five, and also the more edges there are in the graph, the harder it seems to be to figure out a coloring. However, appearances can be deceiving. Blum succeeded in devising a simple, intuitive algorithm that easily finds a 3-coloring for almost all such graphs provided the graph is in fact 3-colorable, has enough edges, and has its edges somewhat randomly distributed.

To show that Blum's algorithm can work even for what might seem to be very hard problems, imagine an adversary who is going to construct the hardest possible graph for the algorithm to 3-color. He is free to exploit his knowledge of the algorithm in order to craft the hardest possible graph that can be 3-colored.

To provide a 3-colorable graph with $n$ nodes, the adversary divides the set of nodes into three groups, A, B, and C (without revealing the division). He doesn't add any edges between the nodes in group A, or between the nodes in groups B or C, respectively. As long as he doesn't add any edges there, the graph is guaranteed to be 3-colorable (and it will also be clear that this procedure is without loss of generality: the adversary can create any possible 3-colorable graph starting with a division into three groups). Now, to make the problem hard, the adversary starts adding edges between some nodes in group A and some nodes in group B, and also between nodes in groups A and C, and also between nodes in groups B and C. He chooses which edges to add between these groups to make the problem as hard as possible, then scrambles the nodes so it is not apparent which nodes are in which group (see figure 11.4). Then the job is to figure out how to 3-color the graph, for which it will suffice to figure out how the adversary divided the nodes into the three groups, but there might be several ways to 3-color it depending on how he has added edges. This seems likely to be a pretty hard job, considering that the adversary tried to construct the hardest possible graph and knew the solution algorithm, and considering that finding 3-colorings is NP-hard.

To make the problem more realistic, characterize the adversary as not all-powerful. In fact, every so often he makes a random mistake in creating the graph. For each possible edge between a node in group A and a node in group B, if the adversary decides he doesn't want to place that edge in the graph, that edge will appear anyway with probability $\epsilon$, some very small number. And likewise for each possible edge between nodes in groups A and C or between nodes in groups B and C.

Now the graph is done, and using the algorithm, we are ready to begin coloring it. Here's how it works. Pick two nodes that are connected by an edge. Call these two nodes node 1 and node 2. Color node 1 blue and node 2 white.

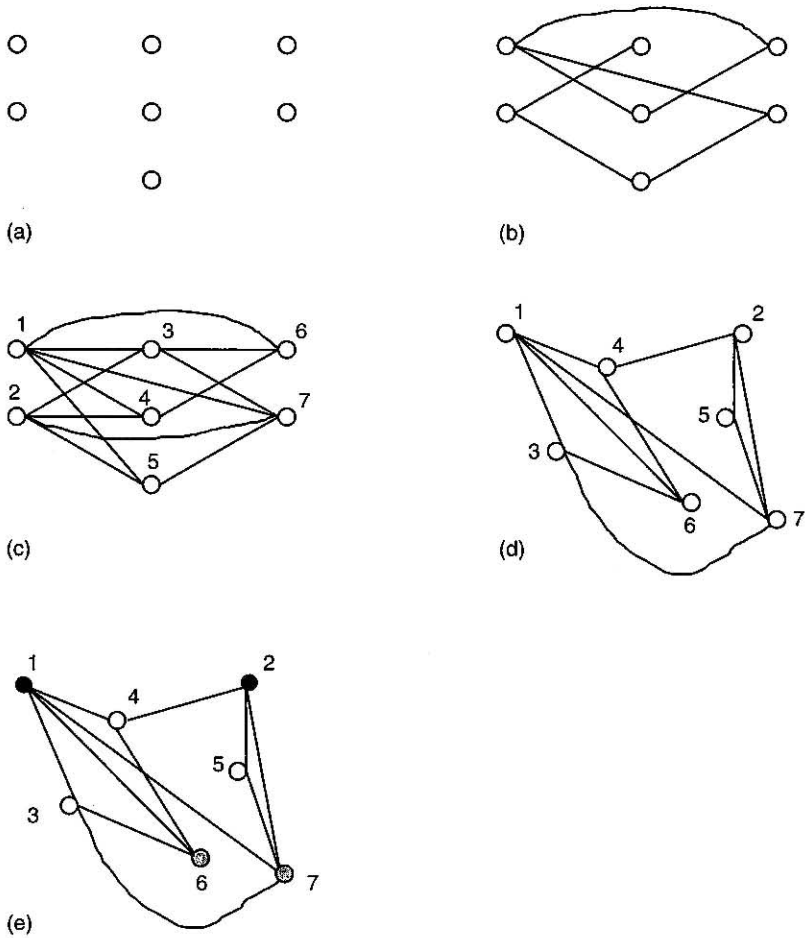


**Figure 11.4**
*(a)* The adversary divides the nodes (seven here) into three sets. *(b)* He chooses edges between some nodes in each of the three sets. *(c)* Because he is fallible, some random additional edges are added. *(d)* The nodes are randomly scrambled (the nodes are numbered so that the reader can follow the scrambling). The production of a (slightly randomized) 3-colorable graph is now complete. It is far from obvious by inspection that the graph is 3-colorable. Indeed, the process seems to produce graphs that are hard to 3-color because the adversary can design the graph to be as hard as possible and his only fallibility is that additional random edges appear. *(e)* The graph may easily be 3-colored by the following procedure. Color node 1 black and node 6 gray. Since these are both connected to node 3, node 3 must be white. Since 3 and 1 are connected to 7, 7 must be gray. Since 1 and 6 are connected to 4, 4 must be white. Since 4 and 7 are connected to 2, 2 must be black. Since 2 and 7 are connected to 5, 5 must be white. The whole graph has been colored by simply following constraints.

Now pick all the nodes in the graph that are connected to both node 2 (white) and node 1 (blue). Obviously, these can all safely be colored red. Now pick all the nodes that are connected to both a red node and a blue node, and color them white. Then pick all the nodes connected to a red node and a white node, and color them blue. Keep on going until all the nodes in the graph have been colored or until there are no more nodes connected to two already-colored nodes.

It's clear that this procedure will never color two nodes the same if they are connected by an edge because at each coloring of groups of nodes, there was only one possible color to use consistent with the graph's being 3-colorable. So the only way the algorithm can fail is if at some step there are nodes left uncolored and none of them is connected by an edge to two differently colored nodes.

Why should we expect to be able to color all the nodes this way? When we picked the first two nodes and colored node 1 blue and node 2 white, why should we expect that a node would be connected to both of those that could be colored red?

We were able to find a node connected to both of the first two nodes because the adversary was error-prone. Every possible edge in the graph between any node in group A and any node in group B was there with probability at least $\epsilon$, even if he didn't want to include it. So, node 1 is connected to a fraction at least $\epsilon$ of all the other nodes in the graph, and so is node 2. And these connections are random: whether one is there is independent of whether the other is there (except that the adversary may have added extra edges). So, nodes 1 and 2 can be expected to *both* be connected to a fraction at least $\epsilon^2$ of the nodes in the graph. This means that as long as $\epsilon > 1/\sqrt{n}$, there are likely to be some nodes connected to both of them, as we need. And as more and more nodes are colored in the graph, there are increasing numbers of ways to find nodes connected to two previously colored nodes, so it becomes increasingly likely that the whole graph can be colored.

By doing a detailed analysis (taking into account that if the first two nodes chosen do not generate a full coloring, another two nodes can be tried), Blum was able to prove that as long as $n$, the number of nodes in the graph, is large, and as long as $\epsilon > 1/\sqrt{n}$, it is highly likely that this algorithm will generate a complete 3-coloring of the graph.

What happened here? As long as there are enough constraints and a degree of randomness in the distribution, it is trivial to color the graph. There is only one possible avenue to explore, and it never runs out. The problem of graph 3-coloring, although provably NP-complete, is in fact easy in the presence of randomness adding enough edges to the graph.

It seems plausible that human thought solves a lot of problems in related fashion. We solve all kinds of perceptual problems so rapidly that our minds cannot possibly

have engaged in much search. When we look at an object, for example, we can identify it in a fraction of a second, and since neurons only fire on a scale of tens of milliseconds, this leaves time for sequential paths at most 100 steps long. When we speak or understand speech, we produce or process words in fractions of a second. When an expert plays chess, possible next moves to explore come into his mind rapidly. These are not random moves but rather chosen moves of high relevance that require sophisticated computations to produce. He may spend an hour thinking over various deep sequences of such moves and comparing the consequences, but some computation is generating suggestions of which moves to consider rapidly.

The suggestion that propagation of constraints in related fashion is important to human intelligence is an old one. One of the earliest clear statements of this idea is due to David Waltz, in his 1972 Ph.D. dissertation (Waltz 1975). He wrote an algorithm for analyzing line drawings of three-dimensional objects such as cubes and polygons. Given a two-dimensional drawing showing the edges of such objects and the edges of the shadows they produce, Waltz's algorithm used constraint propagation to rapidly produce a correct labeling of the figure showing the three-dimensional structure (see figure 11.5).

Waltz's procedure works by attempting to label each edge as one of the following: it is a crack in an object, the boundary of a shadow, or the boundary of a convex object, or it bounds a concave object. The labeling further indicates illumination information: the bounded surface is directly illuminated, shadowed by an object, or self-shadowed by virtue of facing away from the light source. Putting the illumination information together with the edge-type information gives over 50 possible edge labels. Waltz made as well a catalogue of possible junctions of edges: edge lines can meet at an L, at a T, at a fork, and so on, through a list of about ten kinds of inter-

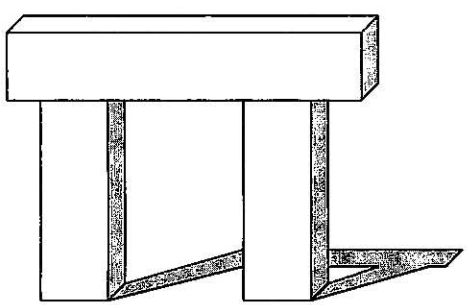

**Figure 11.5**
Waltz's algorithm is able to extract three-dimensional structure and illumination information from line drawings such as this one.

sections. Waltz further exhaustively catalogued the ways the edges coming into a junction can be labeled. And here is the key point: the number of junction labelings that are physically possible if the junction is to appear from a line drawing corresponding to a real three-dimensional collection of objects is much smaller than the total number of possible junction labelings one could imagine. For example, one could imagine that either of the two edges in an L junction could be labeled in any of 50 possible ways, so that L junctions could be imagined to have $2{,}500 = 50^2$ possible labelings. But most of these are not physically possible; if the edge coming in from one side bounds a convex, shaded object, then the edge coming in from the other side also does. In fact, there are only 80 legal types of L junctions.

Because the physics of the figure (the assumption that the figure was produced as a line drawing of a three-dimensional collection of objects, and certain assumptions about illumination) strongly limits the possibilities, constraints are produced. To exploit these constraints, Waltz's procedure starts by listing the possible legal interpretations at some junction in the figure. Going to a neighboring junction, only some of the possible interpretations there will be consistent with the ones here because they must agree on the interpretation of common faces and edges. Propagating these constraints from junction to junction in the figure rapidly results in finding a unique assignment of meaning to edges, assuming there is a unique assignment. Some figures famously have two or more competing interpretations: we can choose to see them as concave or convex. But such examples are rare in natural figures, and occur mainly in drawings that are carefully constructed by artists or psychologists to explore the mechanisms of perception.

Waltz's thesis was impressively difficult, carefully cataloguing many dozens of possible edges and junctions. Later work by Sugihara (1982) was even more painstaking, itemizing hundreds of possibilities that can arise with a greater variety of arrangements of objects and illumination conditions, and thus produced a program that can analyze even more flexible figures. Unfortunately, these programs have not proved to be very useful in the real world because they are not robust, as with much work in AI. Given a correct edge sketch, these programs rapidly produce a correct analysis. But edge sketches generated from real camera input prove to be sufficiently noisy, with spurious edges and breaks, so that the programs break down.

Nonetheless, it seems quite plausible that the basic proposal of constraint propagation is integral to human visual understanding in ways similar to those Waltz proposed as well as to many other mental tasks. Unlike painstaking human analysis, evolution generates robust procedures. It is already very difficult for people to analyze, as Waltz and Sugihara did, what the constraints look like in idealized figures, and perhaps it is too difficult for people to analyze how to extend these ideas

to deal with noise. Indeed, Waltz's analysis may be yet another example where AI researchers, in breaking up a problem up in order to solve it, have introduced intractability through their choice of division, in this case by step 1, which assumed a perfect line drawing, thus breaking up the problem into the subproblems of producing this drawing and analyzing it. But, as Waltz noted, the physics of the world greatly constrains what is possible. Evolution is quite capable of exploiting these constraints.

Indeed, it seems plausible that some kinds of constraint propagation similar to those proposed by Waltz and by Blum underlie many of our thought processes. We don't engage in huge breadth-first searches when we think; our thoughts run along carefully pruned, highly likely paths only. We don't, for example, look at all possible paths in a chess game, as computer programs do. We don't look at all possible next actions when we plan. We jump to consider only certain alternatives. Such an ability to suggest plausible lines of thought is reminiscent of Blum's algorithm: at any given time what we have already figured out constrains us to consider only one or at most a few lines of thought in continuing to analyze the world or to compute what to do next.

Several lines of evidence support and clarify this picture in regard to language understanding. First, there is the difficulty we have in understanding what are known as garden-path sentences because they lead the listener down the garden path and lose her there. Examples, taken from Steven Pinker's book *The Language Instinct*, include "the horse raced past the barn fell" or "fat people eat accumulates." These sentences are hard to understand but have perfectly valid interpretations: The horse that was walked round the track proceeded steadily, but the horse raced past the barn fell; carbohydrates that people eat are quickly broken down, but fat people eat accumulates (Pinker 1994, 212). Apparently, the problem here is that we jump to a conclusion early in the sentence that later proves to be incorrect. When we see "the horse raced," we believe that the meaning has been constrained and fix on an interpretation. Later, when that interpretation proves wrong, we are stuck.

While garden-path sentences indicate that we fix on interpretations early and attempt to reason from the assumed constraints, priming experiments show that we don't fix on a single interpretation of each word instantly when we hear it. If, shortly after we hear a word, another word is flashed on a screen, we recognize the second word faster if it is related to the first. The mind is primed to receive it. Apparently, we are propagating expectations, as in the constraint propagation approach. However, if the priming (first) word has multiple unconnected meanings, it primes related words for all of its meanings. We hear a sentence that includes words with multiple meanings, and even though the sentence is unambiguous and the single meaning

intended for each word may be clear, all the multiple meanings are primed. In some sense, we search all possible alternatives as we hear the words. It seems as if hearing each word turns on circuitry for understanding all the concepts the word can mean, and the consistent meaning for the whole sentence achieves resonance, or perhaps is selected at a higher level. This makes perfect sense. It's not clear how one could converge on the intended meaning for each word without at least invoking the circuitry for understanding the concept unless the word was completely redundant in the sentence and its meaning could be predicted without even reading it. But that couldn't be true for most words. Words are not just tokens, they have meanings, but to have meanings they must invoke the circuitry, the module, for understanding that concept. Until we have invoked the module code, we can't decide what the constraints are because the constraints depend on the semantics of the world, on the compressed description we have for exploiting the structure of the world.

We do indeed invoke our understanding of the world in parsing and interpreting sentences, as is shown again in examples that fail to be garden-path sentences because the garden path is precluded by semantics. Trueswell, Tanenhaus, and Garnsey (1994) performed experiments to track readers' eye movements, which can be accurately (and safely) done by bouncing a laser off the back of the eyeball. The sentence "The defendant examined by the lawyer turned out to be unreliable" has a garden-path quality until we read the word *by*: we may first have posited that the defendant was examining rather than being examined. Readers of this sentence glance back to the beginning when they hit the word *by*, presumably checking their understanding. By contrast, the sentence "The evidence examined by the lawyer turned out to be unreliable" is much easier to parse, and readers' eyes do not glance back to check understanding. The mind must be exploiting its understanding of semantics to make this distinction (Pinker 1994).

Such constraint propagation algorithms are very natural from the point of view taken in this book. I have discussed throughout how the world has enormous amounts of structure and how the processes of mind are based on a compact description of the world exposing and exploiting this structure. Saying the world has structure means that the world is greatly overconstrained. These are dual views of the same phenomenon. The world is not described by random bit strings. Rather, it is highly structured so that only certain special bit strings can occur. Thus, it can be described compactly. Just as, in Waltz's procedure, only a tiny proportion of possible edge labelings would be consistent with physics, so, more generally, only a tiny proportion of possible thought extensions make sense at any given time.

We build all these modules, all this code, in our minds to exploit the compact structure of the world. To extend our thought in this way, we exploit the constraints

inherent in the compressed representation that underlies our minds, not just for vision or for understanding sensation or for speech but also for much of our thought about all subjects. Our reasoning and planning usually are able to follow one line, or else we are stuck.

Animals probably reason in this fashion as well. Our compressed description of the world is partly encoded in our DNA (see chapter 12). It's natural for evolution to produce constraint propagation algorithms because constraint propagation algorithms work fast, and evolution must produce animals that act in real time. A creature that ponders is at risk of being eaten in the meantime. The modules encoded in our DNA are no doubt based on or related to modules in simpler creatures, just as most of our DNA is. Our reasoning goes much further than theirs, but it gets started using modules encoding the basic structure of the world, and without these initial constraints nothing else would be possible. Evolution likely created early the reasoning procedures that propagate constraints, thus achieving rapid though sometimes inaccurate responses, and then it tuned over time to achieve more sophisticated constraint propagation procedures that give deeper though still rapid analysis.

But human beings reason quite a bit further than other animals. The world is overconstrained, so it is possible to keep on extending, to keep on discovering new structure. But our thought processes go even further extending based on existing constraints: we explore several alternatives, looking ahead to decide which is correct.

When an expert plays chess, she does not usually see the next move as completely constrained. Rather, she explores two or three possibilities, hoping to extend them far enough into the future so that she can see one line as being clearly better. As I mentioned previously, when we hear speech, we cannot immediately fasten on a single meaning for each word. Rather, we extend all possible meanings for each word far enough so that we can make a decision, based on the semantics of the world, as to which is the intended meaning.

Extensions of this type can suffer from combinatorial explosion. If all but two possibilities at each chess position can be ruled out, that is much better than having to consider all legal moves. An average chess position has 35 possible next moves, and if they all had to be considered, the player would not be able to look very far ahead at all. But if the player considers two moves at each position and looks a depth $d$ ahead, she must keep $2^d$ possibilities in mind. This permits looking further but still limits the depth.

In constructing the modules in our minds, in constructing new thoughts, we can go only so far. We build new ideas by combining the concepts we already have, by using the modules we have built. Constraints on the world allow us to prune, suggesting certain avenues of thought. Like a chess master, we may consider several possible

plans to some depth. But we are still limited as to how far we can go or the number of possibilities we can consider.

The problem of reasoning about the world is thus hard. But people have made enormous progress at it. As I discuss in chapter 13, this is largely because of language. Individuals engage in computationally intensive searches, trying different ways of extending their knowledge. When someone finds a new discovery, a new sequence of thought that goes on beyond what is fully constrained by old modules and yet that usefully exploits structure in the world, he builds a new module in his mind. And, crucially, because human beings have language, he is able to guide others to construct the new module. Thus people have over tens of thousands of years built vast numbers of modules that exploit the structure of the world in new ways. These provide massive numbers of new constraints that continue to allow us to extend. We have thus greatly extended the program of thought. It is our access to this huge additional program that, in my view, separates human beings from other animals.