Chapter 1
Introduction

§1.1 Human Logic

Humans are, among other things, information processors. We acquire information about the world and use this information to further our ends. One of the strengths of human information processing is our ability to represent and manipulate logical information, not just simple facts but also more complex forms of information, such as negations, alternatives, constraints, and so forth.

To illustrate this ability, consider a simple puzzle in the world of children's blocks. We are given some facts about the arrangement of five blocks in a stack, and we are asked to determine their exact arrangement.

The sentences shown below constitute the premises of the problem. The first sentence tells us the exact location of the red block. The second sentence is not so exact, giving us only a constraint on the relative locations of the green and the blue block. The third sentence tells us what is not true, without saying what is true. The fourth sentence says that one condition holds or another but does not say which. The fifth sentence assures us that an object exists but does not give its identity.

The red block is on the green block.
*The green block is somewhere **above** the blue block.*
*The green block is **not** on the blue block.*
*The yellow block is on the green block **or** the blue block.*
*There is **some** block on the black block.*

Even though the information we need is not literally present in what we are given, it is possible to derive that information. In particular, the conclusions shown below all follow from the premises above.

The red block is on the green block.
The green block is on the yellow block.
The yellow block is on the blue block.
The blue block is on the black block.
The black block is directly on the table.

Unfortunately, it is not always apparent which conclusions may be safely drawn from a given set of premises. What's more, even when we are given the conclusions, as in this example, their correctness may not be immediately obvious.

In order to persuade others of a conclusion that we have drawn, as well as to convince ourselves, it is useful to write down a *proof*, i.e. a series of intermediate conclusions in which each step is immediately obvious.

As an example, consider the following informal proof that, given the premises shown above, the yellow block is on the blue block.

We are told that the yellow block is on the green block or the blue block. We are also told that the red block is on the green block. Given the assumption that there can be only one block on another and that a block cannot be two colors at once, we can conclude that the yellow block is not on the green block. But then, by elimination, the yellow block must be on the blue block.

The concept of proof, in order to be meaningful, requires that we be able to recognize certain reasoning steps as immediately obvious. In other words, we need to be familiar with the reasoning "atoms" out of which complex proof "molecules" are built.

One of Aristotle's great contributions to philosophy was his recognition that what makes a step of a proof immediately obvious is its form rather than its content. It does not matter whether you are talking about blocks or stocks or automobiles. What matters is the structure of the facts with which you are working.

Consider, for example, the reasoning step shown below. We know that all Accords are Hondas, and we know that all Hondas are Japanese cars. Consequently, we can conclude that all Accords are Japanese cars.

All Accords are Hondas.
All Hondas are Japanese.
Therefore, all Accords are Japanese.

Now consider another example. We know that all borogoves are slithy toves, and we know that all slithy toves are mimsy. Consequently, we can conclude that all borogoves are mimsy. What's more, in order to reach this conclusion, we do not need to know anything about borogoves or slithy toves or what it means to be mimsy.

All borogoves are slithy toves.
All slithy toves are mimsy.
Therefore, all borogoves are mimsy.

What is interesting about these examples is that they share the same reasoning structure, viz. the pattern shown below.

All x are y.
All y are z.
Therefore, all x are z.

The existence of such reasoning patterns is fundamental in logic but raises important questions. Which patterns are correct? Are there many such patterns or just a few?

Let us consider the first of these questions first. Obviously there are patterns that are just plain wrong in the sense that they can lead to incorrect conclusions. Consider, as an example, the (faulty) reasoning pattern shown below.

All x are y.
Some y are z.
Therefore, some x are z.

Now let us take a look at an instance of this pattern. If we replace *x* by *Toyotas* and *y* by *Japanese* and *z* by *made in America*, we get the following line of argument, leading to a conclusion that happens to be correct.

All Toyotas are Japanese cars.
Some Japanese cars are made in America.
Therefore, some Toyotas are made in America.

On the other hand, if we replace *x* by *Toyotas* and *y* by *cars* and *z* by *Porsches*, we get a line of argument leading to a conclusion that is incorrect.

All Toyotas are cars.
Some cars are Porsches.
Therefore, some Toyotas are Porsches.

What distinguishes a correct pattern from one that is incorrect is that it must *always* lead to correct conclusions, i.e. conclusions that are true whenever the premises are true. As we will see, this is the defining criterion for what we call *deduction.*

Now, it is noteworthy that there are patterns of reasoning that are sometimes useful but do not satisfy this strict criterion. There is inductive reasoning, abductive reasoning, reasoning by analogy, and so forth.

Induction is reasoning from the particular to the general. The example shown below illustrated this. If we see enough cases in which something is true and we never see a case in which it is false, we tend to conclude that it is always true.

I have seen 1000 black ravens.
I have never seen a raven that is not black.
Therefore, every raven is black.
Now try red Hondas.

Abduction is reasoning from effects to causes. Many things can cause an observed result. We often tend to infer a cause even when our enumeration of possible causes is incomplete.

If there is no fuel, the car will not start.
If there is no spark, the car will not start.
There is spark.
The car will not start.
Therefore, there is no fuel.
What if the car is in a vacuum chamber?

Reasoning by *analogy* is reasoning in which we infer a conclusion based on similarity of two situations, as in the following example.

> *The flow in a pipe is proportional to its diameter.*
> *Wires are like pipes.*
> *Therefore, the current in a wire is proportional to diameter.*
> Now try price.

Of all types of reasoning, deductive reasoning is the only one that *guarantees* its conclusions in all cases. It has some very special properties and holds a unique place in the history of logic. In this book, we concentrate entirely on deduction and leave these other forms of reasoning to others.

§1.2 Formal Logic

Formal Logic is a formal version of human deductive logic. It provides a formal language with an unambiguous syntax and a precise meaning, and it provides rules for manipulating expressions in a way that respects this meaning.

In this regard, there is a strong analogy between the methods of Formal Logic and those of high school algebra. To illustrate this analogy, consider the following algebra problem.

> *Xavier is three times as old as Yolanda. Xavier's age and Yolanda's age add up to twelve. How old are Xavier and Yolanda?*

Typically, the first step in solving such a problem is to express the information in the form of equations. If we let x represent the age of Xavier and y represent the age of Yolanda, we can capture the essential information of the problem as shown below.

$$x - 3y = 0$$
$$x + y = 12$$

Using the methods of algebra, we can then manipulate these expressions to solve the problem. First we subtract the second equation from the first.

$$\begin{array}{r} x - 3y = 0 \\ \underline{x + y = 12} \\ -4y = -12 \end{array}$$

Next, we divide each side of the resulting equation by -4 to get a value for y. Then substituting back into one of the preceding equations, we get a value for x.

$$y = 3$$
$$x = 9$$

Now, consider the following logic problem.

> *If Mary loves Pat, then Mary loves Quincy. If it is Monday and raining, then Mary loves Pat or Quincy. If it is Monday and raining, does Mary love Quincy?*

As with the algebra problem, the first step is formalization. Let p represent the possibility that Mary loves Pat; let q represent the possibility that Mary loves Quincy; let m represent the possibility that it is Monday; and let r represent the possibility that it is raining. With these abbreviations, we can represent the essential information of this problem with the following three logical sentences. The first says that p *implies* q, i.e. if Mary loves Pat, then Mary loves Quincy. The second says that m and r *implies* that p is true *or* q is true, i.e. if it is Monday and raining, then Mary loves Pat or Mary loves Quincy.

$$p \Rightarrow q$$
$$m \wedge r \Rightarrow p \vee q$$

As with Algebra, Formal Logic defines certain operations that we can use to manipulate expressions. The operation shown below is a variant of what is called *propositional resolution.*

$$\begin{array}{ccc} p_1 \wedge ... \wedge p_k & \Rightarrow & q_1 \vee ... \vee q_l \\ r_1 \wedge ... \wedge r_m & \Rightarrow & s_1 \vee ... \vee s_n \\ \hline p_1 \wedge ... \wedge p_k \wedge r_1 \wedge ... \wedge r_m & \Rightarrow & q_1 \vee ... \vee q_l \vee s_1 \vee ... \vee s_n \end{array}$$

There are two elaborations of this operation. (1) If p_i on the left hand side of one sentence is the same as q_j in the right hand side of the other sentence, it is okay to drop the two symbols, with the proviso that *only one* such pair may be dropped. (2) If a constant is repeated on the same side of a single sentence, all but one of the occurrences can be deleted.

We can use this operation to solve the problem of Mary's love life. Looking at the two premises above, we notice that p occurs on the left-hand side of one sentence and the right-hand side of the other. Consequently, we can cancel the p and thereby derive the conclusion that, if is Monday and raining, then Mary loves Quincy or Mary loves Quincy. Dropping the repeated symbol on the right hand side, we arrive at the conclusion that, if it is Monday and raining, then Mary loves Quincy.

$$\begin{array}{ccc} p & \Rightarrow & q \\ m \wedge r & \Rightarrow & p \vee q \\ \hline m \wedge r & \Rightarrow & q \vee q \\ m \wedge r & \Rightarrow & q \end{array}$$

This example is interesting in that it showcases our formal language for encoding logical information. As with algebra, we use symbols to represent relevant aspects of the

world in question, and we use operators to connect these symbols in order to express information about the things those symbols represent.

The example also introduces one of the most important operations in Formal Logic, viz. resolution (in this case a restricted form of resolution). Resolution has the property of being *complete* for an important class of logic problems, i.e. it is the *only* operation necessary to solve any problem in the class.

§1.3 Computational Logic

The existence of a formal language for representing information and the existence of a corresponding set of mechanical manipulation rules together have an important consequence, viz. the possibility of *automated reasoning* using digital computers.

The idea is simple. We use our formal representation to encode the premises of a problem as data structures in a computer, and we program the computer to apply our mechanical rules in a systematic way. The rules are applied until the desired conclusion is attained or until it is determined that the desired conclusion cannot be attained. (Unfortunately, in some cases, this determination cannot be made; and the procedure never halts. Nevertheless, as discussed in later chapters, the idea is basically sound.)

Computational Logic is a branch of Mathematics that is concerned with the theoretical underpinnings of automated reasoning. It shares much with Formal Logic, but there are some differences.

Formal Logic is *exclusively* concerned with precise syntax and semantics and correctness and completeness of reasoning. With that as the only goal, specialists in Formal Logic place their emphasis on minimal sets of deductive rules in order to simplify their analysis. Unfortunately, these rules are not always easy to implement or efficient to apply.

Like Formal Logic, Computational Logic is concerned with precise syntax and semantics and correctness and completeness of reasoning. However, it is also concerned with efficiency. The upshot of this difference is that specialists place their emphasis of different languages and different sets of rules, paying more attention to those that better suited to automation and efficient application.

In this book, we examine the languages and rules of both Formal Logic and Computational Logic. However, the emphasis here is squarely on automation.

§1.4 Applications

In our discussion so far, our examples have been restricted to children's blocks and automobiles and the love life of the fickle Mary. While Computational Logic has relevance in these important areas, the techniques can be used in other areas as well. The following paragraphs outline some of these uses.

Mathematics. Automated reasoning programs can be used to check proofs and, in some cases, to produce proofs or portions of proofs.

Database Systems. The language of computational logic can be used to encode *integrity constraints* that capture constraints on databases and can be used to define virtual *views* of data in terms of explicitly stored tables. Automated reasoning techniques can be used to optimize queries.

Software Analysis. Automated reasoning tools can be used to verify various properties of programs, such as correctness, termination, complexity, and so forth.

Hardware Engineering. Automated reasoning tools can be used to validate hardware designs, diagnose failures, develop testing programs, and in some cases synthesize designs.

Information Integration on the Internet. The language of computational logic can be used to relate the vocabulary and structure of disparate data sources, and automated reasoning techniques can be used to create programs to integrate the data in these sources.

Law and Business. The language of computational logic can be used to encode regulations and business rules, and automated reasoning techniques can be used to analyze such regulations for inconsistency and overlap.

§1.5 Reading Guide

Although Computational Logic is a single field of study, there is more than one logic in this field. In the three main units of this book, we look at three different types of logic, each more sophisticated than the last.

Propositional Logic is the logic of propositions. Symbols in the language represent "conditions" in the world, and complex sentences in the language express interrelationships among these conditions. The primary operators are Boolean connectives, such as *and*, *or*, and *not.*

Relational Logic expands upon Propositional Logic by providing a means for explicitly talking about individual objects and their interrelationships (not just monolithic conditions). In order to do so, we expand our language to include object constants, relation constants, variables of various sorts, and quantifiers.

Metalevel Logic goes one step further and provides a means for referring to linguistic expressions as objects in their own right and provides operators for expressing relationships between expressions and other expressions and relationships between expressions and the things they represent. This extension allows us to encode information about information, and it allows us to define the notions of truth and belief.

Despite these differences, there are many commonalities among these logics. In particular, in each case, there is a language with a formal syntax; there is a precise semantics; and there are legal rules for manipulating expressions in the language.

These similarities allow us to compare the various logics and to gain an appreciation of the tradeoff between expressiveness and computational complexity. On the one hand, the introduction of additional linguistic complexity makes it possible to say things that cannot be said in more restricted languages. On the other hand, the

introduction of additional linguistic flexibility has adverse effects on decidability and computational complexity. As we proceed though the material, our attention will range from the completely decidable (and in some cases polynomial) case of Propositional Logic to the completely undecidable case of Higher Order Logic and on to the potentially paradoxical case of Metalevel Logic.

Finally, in the hopes of preventing difficulties, it is worth pointing out a potential source of confusion. This book exists in the *meta* world. It contains sentences about sentences; it contains proofs about proofs. In some places, we use similar mathematical symbology both for sentences *in* logic and sentences *about* logic. Wherever possible, we will try to be clear about this distinction, but the potential for confusion is real. Unfortunately, this comes with the territory. We are using logic to study logic. It is our most powerful intellectual tool.

Historical Notes

Although the prospect of automated reasoning has achieved practical realization only in the last few decades, it is interesting to note that the concept itself is not new. In fact, the idea of building machines capable of logical reasoning has a long tradition.

One of the first individuals to give voice to this idea was Leibnitz. He conceived of "a universal algebra by which all knowledge, including moral and metaphysical truths, can some day be brought within a single deductive system". Having already perfected a mechanical calculator for arithmetic, he argued that, with this universal algebra, it would be possible to build a machine capable of rendering the consequences of such a system mechanically.

Boole gave substance to this dream in the 1800s with the invention of Boolean algebra and the creation of a machine capable of computing accordingly.

The early twentieth century brought additional advances in logic, notably the invention of the predicate calculus by Russell and Whitehead and the proof of the corresponding completeness and incompleteness theorems by Godel in the 1930s.

The advent of the digital computer in the 1940s gave increased attention to the prospects for automated reasoning. Research in artificial intelligence led to the development of efficient algorithms for logical reasoning, highlighted by Robinson's invention of resolution theorem proving in the 1960s.

Today, the prospect of automated reasoning has moved from the realm of possibility to that of practicality, with the creation of *logic technology* in the form of automated reasoning systems, such as Otter, the Prolog Technology Theorem Prover, Epilog, and others.

Exercises

1. *Blocks World.* The premises in the Blocks World problem at the beginning of this chapter are not complete. The problem assumes some things about the world that are not explicitly recorded there. Write out some of these implicit assumptions as sentences in English.

2. *Propositional Resolution*. Consider the problem of Mary, Pat, and Quincy given earlier.

(a) Encode the assumption that Mary loves one of Pat or Quincy but not both. (Hint: the sentence $p \Rightarrow .$ means that p is false.)

(b) Using this additional assumption, use propositional resolution to prove that, if it is Monday and raining, then Mary does *not* love Pat.