Achilles: Hah!

Zeno: And now the Tortoise is but a single rod ahead of Achilles. Within only a moment, Achilles has attained that spot.

Achilles: Ho ho!

Zeno: Yet, in that short moment, the Tortoise has managed to advance a slight amount. In a flash, Achilles covers that distance, too.

Achilles: Hee hee hee!

Zeno: But in that very short flash, the Tortoise has managed to inch ahead by ever so little, and so Achilles is still behind. Now you see that in order for Achilles to catch the Tortoise, this game of "try-to-catch-me" will have to be played an INFINITE number of times—and therefore Achilles can NEVER catch up with the Tortoise!

Tortoise: Heh heh heh heh!

Achilles: Hmm . . . hmm . . . hmm . . . hmm . . . hmm . . . That argument sounds wrong to me. And yet, I can't quite make out what's wrong with it.

Zeno: Isn't it a teaser? It's my favorite paradox.

Tortoise: Excuse me, Zeno, but I believe your tale illustrates the wrong principle, does it not? You have just told us what will come to be known, centuries hence, as Zeno's "Achilles paradox", which shows (ahem!) that Achilles will never catch the Tortoise; but the proof that Motion Is Inherently Impossible (and thence that Motion Unexists) is your "dichotomy paradox", isn't that so?

Zeno: Oh, shame on me. Of course, you're right. That's the one about how, in getting from A to B, one has to go halfway first—and of that stretch one also has to go halfway, and so on and so forth. But you see, both those paradoxes really have the same flavor. Frankly, I've only had one Great Idea—I just exploit it in different ways.

Achilles: I swear, these arguments contain a flaw. I don't quite see where, but they cannot be correct.

Zeno: You doubt the validity of my paradox? Why not just try it out? You see that red flag down there, at the far end of the runway?

Achilles: The impossible one, based on an Escher print?

Zeno: Exactly. What do you say to you and Mr. Tortoise racing for it, allowing Mr. T a fair head start of, well, I don't know—

Tortoise: How about ten rods?

Zeno: Very good—ten rods.

Achilles: Any time.

Zeno: Excellent! How exciting! An empirical test of my rigorously proven Theorem! Mr. Tortoise, will you position yourself ten rods upwind?

(The Tortoise moves ten rods closer to the flag.)

Are you both ready?

Tortoise and Achilles: Ready!

Zeno: On your mark! Get set! Go!

CHAPTER I

The MU-puzzle

Formal Systems

ONE OF THE most central notions in this book is that of a *formal system.* The type of formal system I use was invented by the American logician Emil Post in the 1920's, and is often called a "Post production system". This Chapter introduces you to a formal system and moreover, it is my hope that you will want to explore this formal system at least a little; so to provoke your curiosity, I have posed a little puzzle.

"Can you produce MU?" is the puzzle. To begin with, you will be supplied with a *string* (which means a string of letters).* Not to keep you in suspense, that string will be MI. Then you will be told some rules, with which you can change one string into another. If one of those rules is applicable at some point, and you want to use it, you may, but—there is nothing that will dictate which rule you should use, in case there are several applicable rules. That is left up to you—and of course, that is where playing the game of any formal system can become something of an art. The major point, which almost doesn't need stating, is that you must not do anything which is outside the rules. We might call this restriction the "Requirement of Formality". In the present Chapter, it probably won't need to be stressed at all. Strange though it may sound, though, I predict that when you play around with some of the formal systems of Chapters to come, you will find yourself violating the Requirement of Formality over and over again, unless you have worked with formal systems before.

The first thing to say about our formal system—the *MIU-system*—is that it utilizes only three letters of the alphabet: M, I, U. That means that the only strings of the MIU-system are strings which are composed of those three letters. Below are some strings of the MIU-system:

MU
UIM
MUUMUU
UIIUMIUUIMUIIUMIUUIMUIIU

* In this book, we shall employ the following conventions when we refer to strings. When the string is in the same typeface as the text, then it will be enclosed in single or double quotes. Punctuation which belongs to the sentence and not to the string under discussion will go *outside* of the quotes, as logic dictates. For example, the first letter of this sentence is 'F', while the first letter of 'this sentence' is 't'. When the string is in Quadrata Roman, however, quotes will usually be left off, unless clarity demands them. For example, the first letter of Quadrata is Q.

But although all of these are legitimate strings, they are not strings which are "in your possession". In fact, the only string in your possession so far is MI. Only by using the rules, about to be introduced, can you enlarge your private collection. Here is the first rule:

RULE I: If you possess a string whose last letter is I, you can add on a U at the end.

By the way, if up to this point you had not guessed it, a fact about the meaning of "string" is that the letters are in a fixed order. For example, MI and IM are two different strings. A string of symbols is not just a "bag" of symbols, in which the order doesn't make any difference.

Here is the second rule:

RULE II: Suppose you have M*x*. Then you may add M*xx* to your collection.

What I mean by this is shown below, in a few examples.

From MIU, you may get MIUIU.
From MUM, you may get MUMUM.
From MU, you may get MUU.

So the letter '*x*' in the rule simply stands for any string; but once you have decided which string it stands for, you have to stick with your choice (until you use the rule again, at which point you may make a new choice). Notice the third example above. It shows how, once you possess MU, you can add another string to your collection; but you have to get MU first! I want to add one last comment about the letter '*x*': it is not part of the formal system in the same way as the three letters 'M', 'I', and 'U' are. It is useful for us, though, to have some way to talk in general about strings of the system, symbolically—and that is the function of the '*x*': to stand for an arbitrary string. If you ever add a string containing an '*x*' to your "collection", you have done something wrong, because strings of the MIU-system never contain "*x*"'s!

Here is the third rule:

RULE III: If III occurs in one of the strings in your collection, you may make a new string with U in place of III.

Examples:

From UMIIIMU, you could make UMUMU.
From MIIII, you could make MIU (also MUII).
From IIMII, you can't get anywhere using this rule.
(The three I's have to be consecutive.)
From MIII, make MU.

Don't, under any circumstances, think you can run this rule backwards, as in the following example:

From MU, make MIII. ⇐ This is wrong.

Rules are one-way.

Here is the final rule:

RULE IV: If UU occurs inside one of your strings, you can drop it.

From UUU, get U.
From MUUUIII, get MUIII.

There you have it. Now you may begin trying to make MU. Don't worry if you don't get it. Just try it out a bit—the main thing is for you to get the flavor of this MU-puzzle. Have fun.

Theorems, Axioms, Rules

The answer to the MU-puzzle appears later in the book. For now, what is important is not finding the answer, but looking for it. You probably have made some attempts to produce MU. In so doing, you have built up your own private collection of strings. Such strings, producible by the rules, are called *theorems*. The term "theorem" has, of course, a common usage in mathematics which is quite different from this one. It means some statement in ordinary language which has been proven to be true by a rigorous argument, such as Zeno's Theorem about the "unexistence" of motion, or Euclid's Theorem about the infinitude of primes. But in formal systems, theorems need not be thought of as statements—they are merely strings of symbols. And instead of being *proven*, theorems are merely *produced*, as if by machine, according to certain typographical rules. To emphasize this important distinction in meanings for the word "theorem", I will adopt the following convention in this book: when "theorem" is capitalized, its meaning will be the everyday one—a Theorem is a statement in ordinary language which somebody once proved to be true by some sort of logical argument. When uncapitalized, "theorem" will have its technical meaning: a string producible in some formal system. In these terms, the MU-puzzle asks whether MU is a theorem of the MIU-system.

I gave you a theorem for free at the beginning, namely MI. Such a "free" theorem is called an *axiom*—the technical meaning again being quite different from the usual meaning. A formal system may have zero, one, several, or even infinitely many axioms. Examples of all these types will appear in the book.

Every formal system has symbol-shunting rules, such as the four rules of the MIU-system. These rules are called either *rules of production* or *rules of inference*. I will use both terms.

The last term which I wish to introduce at this point is *derivation*. Shown below is a derivation of the theorem MUIIU:

(1)	MI	axiom
(2)	MII	from (1) by rule II

(3) MIIII — from (2) by rule II
(4) MIIIIU — from (3) by rule I
(5) MUIU — from (4) by rule III
(6) MUIUUIU — from (5) by rule II
(7) MUIIU — from (6) by rule IV

A derivation of a theorem is an explicit, line-by-line demonstration of how to produce that theorem according to the rules of the formal system. The concept of derivation is modeled on that of proof, but a derivation is an austere cousin of a proof. It would sound strange to say that you had *proven* MUIIU, but it does not sound so strange to say you have *derived* MUIIU.

Inside and Outside the System

Most people go about the MU-puzzle by deriving a number of theorems, quite at random, just to see what kind of thing turns up. Pretty soon, they begin to notice some properties of the theorems they have made; that is where human intelligence enters the picture. For instance, it was probably not obvious to you that all theorems would begin with M, until you had tried a few. Then, the pattern emerged, and not only could you see the pattern, but you could understand it by looking at the rules, which have the property that they make each new theorem inherit its first letter from an earlier theorem; ultimately, then, all theorems' first letters can be traced back to the first letter of the sole axiom MI—and that is a proof that theorems of the MIU-system must all begin with M.

There is something very significant about what has happened here. It shows one difference between people and machines. It would certainly be possible—in fact it would be very easy—to program a computer to generate theorem after theorem of the MIU-system; and we could include in the program a command to stop only upon generating U. You now know that a computer so programmed would never stop. And this does not amaze you. But what if you asked a friend to try to generate U? It would not surprise you if he came back after a while, complaining that he can't get rid of the initial M, and therefore it is a wild goose chase. Even if a person is not very bright, he still cannot help making some observations about what he is doing, and these observations give him good insight into the task—insight which the computer program, as we have described it, lacks.

Now let me be very explicit about what I meant by saying this shows a difference between people and machines. I meant that it is *possible* to program a machine to do a routine task in such a way that the machine will never notice even the most obvious facts about what it is doing; but it is inherent in human consciousness to notice some facts about the things one is doing. But you knew this all along. If you punch "1" into an adding machine, and then add 1 to it, and then add 1 again, and again, and again, and continue doing so for hours and hours, the machine will never learn to anticipate you, and do it itself, although any person would pick up the

repetitive behavior very quickly. Or, to take a silly example, a car will never pick up the idea, no matter how much or how well it is driven, that it is supposed to avoid other cars and obstacles on the road; and it will never learn even the most frequently traveled routes of its owner.

The difference, then, is that it is *possible* for a machine to act unobservant; it is impossible for a human to act unobservant. Notice I am not saying that all machines are necessarily incapable of making sophisticated observations; just that some machines are. Nor am I saying that all people are always making sophisticated observations; people, in fact, are often very unobservant. But machines can be made to be totally unobservant; and people cannot. And in fact, most machines made so far are pretty close to being totally unobservant. Probably for this reason, the property of being unobservant seems to be the characteristic feature of machines, to most people. For example, if somebody says that some task is "mechanical", it does not mean that people are incapable of doing the task; it implies, though, that only a machine could do it over and over without ever complaining, or feeling bored.

Jumping out of the System

It is an inherent property of intelligence that it can jump out of the task which it is performing, and survey what it has done; it is always looking for, and often finding, patterns. Now I said that an intelligence can jump out of its task, but that does not mean that it always will. However, a little prompting will often suffice. For example, a human being who is reading a book may grow sleepy. Instead of continuing to read until the book is finished, he is just as likely to put the book aside and turn off the light. He has stepped "out of the system" and yet it seems the most natural thing in the world to us. Or, suppose person A is watching television when person B comes in the room, and shows evident displeasure with the situation. Person A may think he understands the problem, and try to remedy it by exiting the present system (that television program), and flipping the channel knob, looking for a better show. Person B may have a more radical concept of what it is to "exit the system"—namely to turn the television off! Of course, there are cases where only a rare individual will have the vision to perceive a system which governs many peoples' lives, a system which had never before even been recognized as a system; then such people often devote their lives to convincing other people that the system really is there, and that it ought to be exited from!

How well have computers been taught to jump out of the system? I will cite one example which surprised some observers. In a computer chess tournament not long ago in Canada, one program—the weakest of all the competing ones—had the unusual feature of quitting long before the game was over. It was not a very good chess player, but it at least had the redeeming quality of being able to spot a hopeless position, and to resign then and there, instead of waiting for the other program to go through the

boring ritual of checkmating. Although it lost every game it played, it did it in style. A lot of local chess experts were impressed. Thus, if you define "the system" as "making moves in a chess game", it is clear that this program had a sophisticated, preprogrammed ability to exit from the system. On the other hand, if you think of "the system" as being "whatever the computer had been programmed to do", then there is no doubt that the computer had no ability whatsoever to exit from that system.

It is very important when studying formal systems to distinguish working *within* the system from making statements or observations *about* the system. I assume that you began the MU-puzzle, as do most people, by working within the system; and that you then gradually started getting anxious, and this anxiety finally built up to the point where without any need for further consideration, you exited from the system, trying to take stock of what you had produced, and wondering why it was that you had not succeeded in producing MU. Perhaps you found a reason why you could not produce MU; that is thinking about the system. Perhaps you produced MIU somewhere along the way; that is working within the system. Now I do not want to make it sound as if the two modes are entirely incompatible; I am sure that every human being is capable to some extent of working inside a system and simultaneously thinking about what he is doing. Actually, in human affairs, it is often next to impossible to break things neatly up into "inside the system" and "outside the system"; life is composed of so many interlocking and interwoven and often inconsistent "systems" that it may seem simplistic to think of things in those terms. But it is often important to formulate simple ideas very clearly so that one can use them as models in thinking about more complex ideas. And that is why I am showing you formal systems; and it is about time we went back to discussing the MIU-system.

M-Mode, I-Mode, U-Mode

The MU-puzzle was stated in such a way that it encouraged some amount of exploration within the MIU-system—deriving theorems. But it was also stated in a way so as not to imply that staying inside the system would necessarily yield fruit. Therefore it encouraged some oscillation between the two modes of work. One way to separate these two modes would be to have two sheets of paper; on one sheet, you work "in your capacity as a machine", thus filling it with nothing but M's, I's, and U's; on the second sheet, you work "in your capacity as a thinking being", and are allowed to do whatever your intelligence suggests—which might involve using English, sketching ideas, working backwards, using shorthand (such as the letter 'x'), compressing several steps into one, modifying the rules of the system to see what that gives, or whatever else you might dream up. One thing you might do is notice that the numbers 3 and 2 play an important role, since I's are gotten rid of in three's, and U's in two's—and doubling of length (except for the M) is allowed by rule II. So the second sheet might

also have some figuring on it. We will occasionally refer back to these two modes of dealing with a formal system, and we will call them the *Mechanical mode (M-mode)* and the *Intelligent mode (I-mode)*. To round out our modes, with one for each letter of the MIU-system, I will also mention a final mode—the *Un-mode (U-mode)*, which is the Zen way of approaching things. More about this in a few Chapters.

Decision Procedures

An observation about this puzzle is that it involves rules of two opposing tendencies—the *lengthening rules* and the *shortening rules*. Two rules (I and II) allow you to increase the size of strings (but only in very rigid, prescribed ways, of course); and two others allow you to shrink strings somewhat (again in very rigid ways). There seems to be an endless variety to the order in which these different types of rules might be applied, and this gives hope that one way or another, MU could be produced. It might involve lengthening the string to some gigantic size, and then extracting piece after piece until only two symbols are left; or, worse yet, it might involve successive stages of lengthening and then shortening and then lengthening and then shortening, and so on. But there is no guarantee of it. As a matter of fact, we already observed that U cannot be produced at all, and it will make no difference if you lengthen and shorten till kingdom come.

Still, the case of U and the case of MU seem quite different. It is by a very superficial feature of U that we recognize the impossibility of producing it: it doesn't begin with an M (whereas all theorems must). It is very convenient to have such a simple way to detect nontheorems. However, who says that that test will detect *all* nontheorems? There may be lots of strings which begin with M but are not producible. Maybe MU is one of them. That would mean that the "first-letter test" is of limited usefulness, able only to detect a portion of the nontheorems, but missing others. But there remains the possibility of some more elaborate test which discriminates perfectly between those strings which can be produced by the rules, and those which cannot. Here we have to face the question, "What do we mean by a test?" It may not be obvious why that question makes sense, or is important, in this context. But I will give an example of a "test" which somehow seems to violate the spirit of the word.

Imagine a genie who has all the time in the world, and who enjoys using it to produce theorems of the MIU-system, in a rather methodical way. Here, for instance, is a possible way the genie might go about it:

Step 1: Apply every applicable rule to the axiom MI. This yields two new theorems: MIU, MII.
Step 2: Apply every applicable rule to the theorems produced in step 1. This yields three new theorems: MIIU, MIUIU, MIIII.

Step 3: Apply every applicable rule to the theorems produced in step 2. This yields five new theorems: MIIIIU, MIIUIIU, MIUIUIUIU, MIIIIIIII, MUI.

.
.
.

This method produces every single theorem sooner or later, because the rules are applied in every conceivable order. (See Fig. 11.) All of the lengthening-shortening alternations which we mentioned above eventually get carried out. However, it is not clear how long to wait for a given string

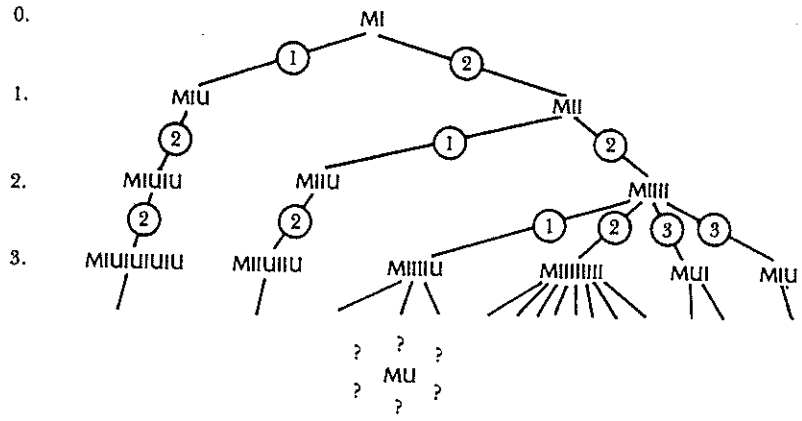

FIGURE 11. A systematically constructed "tree" of all the theorems of the MIU-system. The N*th level down contains those theorems whose derivations contain exactly* N *steps. The encircled numbers tell which rule was employed. Is* MU *anywhere in this tree?*

to appear on this list, since theorems are listed according to the shortness of their derivations. This is not a very useful order, if you are interested in a specific string (such as MU), and you don't even know if it has any derivation, much less how long that derivation might be.

Now we state the proposed "theoremhood-test":

> Wait until the string in question is produced; when that happens, you know it is a theorem—and if it never happens, you know that it is not a theorem.

This seems ridiculous, because it presupposes that we don't mind waiting around literally an infinite length of time for our answer. This gets to the crux of the matter of what should count as a "test". Of prime importance is a guarantee that we will get our answer in a finite length of time. If there is a test for theoremhood, a test which does always terminate in a finite

amount of time, then that test is called a *decision procedure* for the given formal system.

When you have a decision procedure, then you have a very concrete characterization of the nature of all theorems in the system. Offhand, it might seem that the rules and axioms of the formal system provide no less complete a characterization of the theorems of the system than a decision procedure would. The tricky word here is "characterization". Certainly the rules of inference and the axioms of the MIU-system do characterize, *implicitly*, those strings that are theorems. Even *more* implicitly, they characterize those strings that are *not* theorems. But implicit characterization is not enough, for many purposes. If someone claims to have a characterization of all theorems, but it takes him infinitely long to deduce that some particular string is not a theorem, you would probably tend to say that there is something lacking in that characterization—it is not quite concrete enough. And that is why discovering that a decision procedure exists is a very important step. What the discovery means, in effect, is that you can perform a test for theoremhood of a string, and that, even if the test is complicated, it is *guaranteed to terminate.* In principle, the test is just as easy, just as mechanical, just as finite, just as full of certitude, as checking whether the first letter of the string is M. A decision procedure is a "litmus test" for theoremhood!

Incidentally, one requirement on formal systems is that the set of *axioms* must be characterized by a decision procedure—there must be a litmus test for axiomhood. This ensures that there is no problem in getting off the ground at the beginning, at least. That is the difference between the set of axioms and the set of theorems: the former always has a decision procedure, but the latter may not.

I am sure you will agree that when you looked at the MIU-system for the first time, you had to face this problem exactly. The lone axiom was known, the rules of inference were simple, so the theorems had been implicitly characterized—and yet it was still quite unclear what the consequences of that characterization were. In particular, it was still totally unclear whether MU is, or is not, a theorem.