

HUMAN PROBLEM SOLVING

ALLEN NEWELL

University Professor
Carnegie-Mellon University

HERBERT A. SIMON

Professor of Computer Science and Psychology
Carnegie-Mellon University

PRENTICE-HALL, INC., Englewood Cliffs, New Jersey

© 1972 by Prentice-Hall, Inc., Englewood Cliffs. N. J.

All rights reserved. No part of this book may be reproduced in any form or by any means without permission in writing from the publisher.

13-445403-0

Library of Congress Catalog Card Number: 79-15252

Printed in the United States of America

Current printing (last digit):

10 9 8 7

PRENTICE-HALL INTERNATIONAL, INC., *London*
PRENTICE-HALL OF AUSTRALIA, PTY. LTD., *Sydney*
PRENTICE-HALL OF CANADA, LTD., *Toronto*
PRENTICE-HALL OF INDIA PRIVATE LIMITED, *New Delhi*
PRENTICE-HALL OF JAPAN, INC., *Tokyo*

1

INTRODUCTION

The aim of this book is to advance our understanding of how humans think. It seeks to do so by putting forth a theory of human problem solving, along with a body of empirical evidence that permits assessment of the theory.

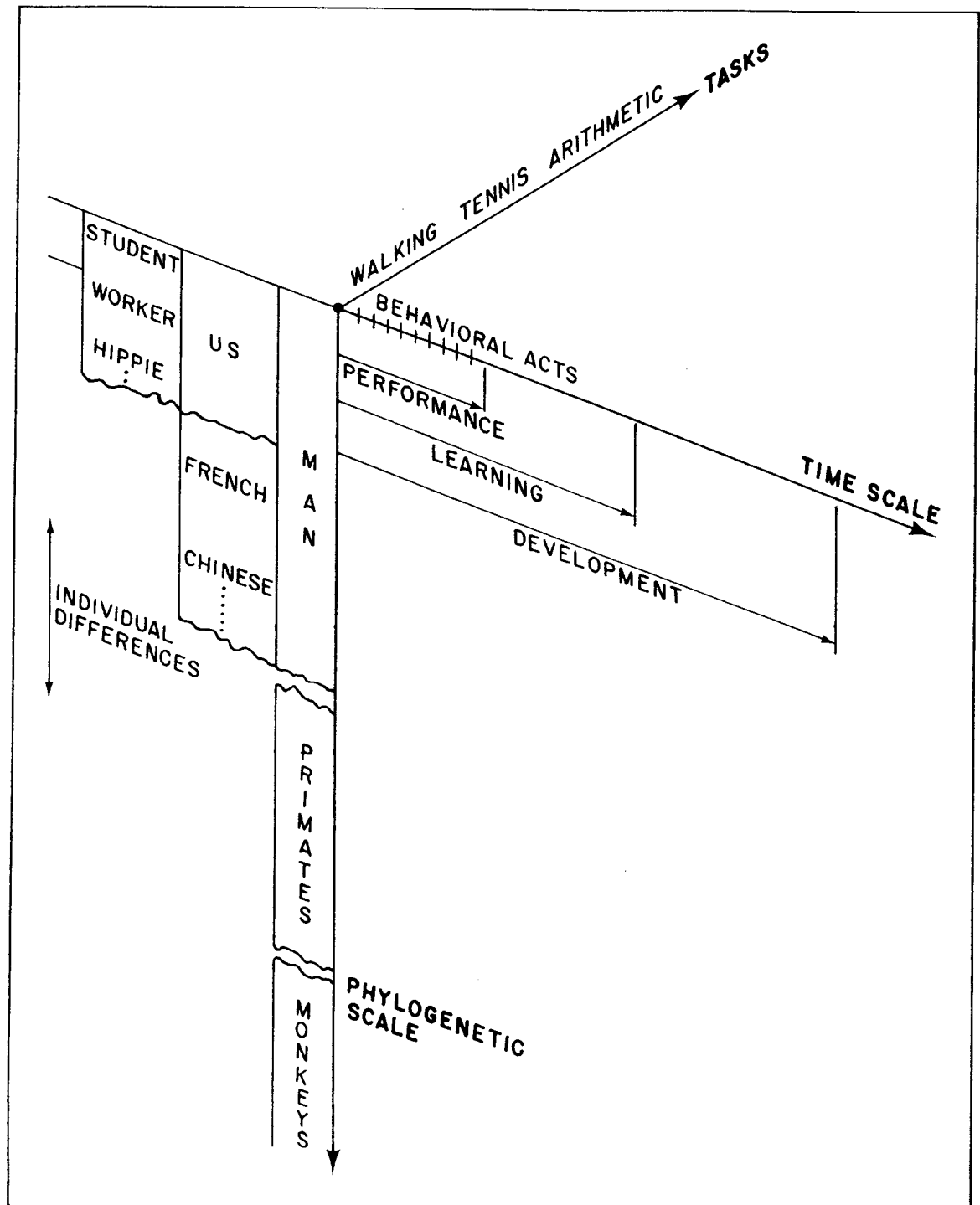
No single work advances understanding very far. The aims of a scientific work are limited by the formal character of the theory, by the phenomena it encompasses, by the experimental situations it uses, by the types of subjects it studies, and by the data it gathers. Of course, a theory may speak beyond its initial base—all scientists hope for just that. But science is a series of successive approximations. Not all things can be done at once, and even if one aspires to go far, he must start somewhere. If one aims at covering all of human thinking in a single work, the work will necessarily be superficial. If one aims at probing in depth, then many aspects of the subject, however important, will be left untouched.

THE SCOPE OF THE BOOK

Our first task, then, is to indicate the scope of this work—what it includes and what it deliberately excludes. The boundaries need not be carefully marked; but the central focus should be made clear.

Figure 1.1 attempts to compress in one diagram many of the dimensions

FIGURE 1.1
dimensions of variation of the human system



along which the total human system can vary. Its purpose is to demark the particular portions included in the present work, not to provide a new or total view. The focus of the diagram is the individual human being. He is a system consisting of parts: sensory subsystems, memory, effectors, arousal subsystems, and so on. It makes little difference for immediate purposes that we are unsure just what is the most appropriate way to enumerate the subsystems. We can limit study—and we will—to one or a few parts.

Task Dimension. A human behaves in a number of different classes of situations, which we will come to call *task environments*. In the figure, we have represented these in a single dimension, but clearly a whole geography is intended: a human does mathematics, he walks, he interacts with his fellow man, he drives cars, he makes love, he argues, he buys food, he dies.

Performance-Learning-Development Dimension. Holding the task environment constant, we usually distinguish a human who is *performing* a task from one who is *learning* to perform a task, or one who is *developing* with respect to a task. Within the first group, those who are performing, we can focus on successively smaller acts, so that we need not even be concerned with an entire performance. It is not important that the distinctions are imperfect. (For example, it is often impossible to distinguish with confidence between learning and maturation.) The distinctions among these three kinds of activities are to some extent correlated with time scale, and we have so depicted them.

Individual-Difference Dimension. A single man may be viewed as a member of various *populations*. Each man differs—both in systematic ways and simply by virtue of his unique genetic endowment and historical fate—from all other men. Differences dependent on age are strongly related to development and are indicated along the temporal axis. But cultures also differ, and within a culture various demographic variables, such as socioeconomic status, isolate distinct populations. More broadly, of course, Man is only one among innumerable species, and so the vertical axis extends to other organisms.

The Book's Focus

From the many directions of variation expressed or implied in Figure 1.1 limited regions can be carved out for attention: the development of intelligence as a function of organismic level (in the evolutionary sense); the learning of personal roles in social groups of college students; and so on. Each region provides a starting point for investigation and description; each leaves out most of the human phenomena.

The present study is concerned with the performance of intelligent adults in our own culture. The tasks discussed are short (half-hour), moderately difficult problems of a symbolic nature. The three main tasks we use—chess, symbolic logic, and algebra-like puzzles (called cryptarithmic puzzles)—typify this class of problems. The study is concerned with the integrated activities that constitute problem solving. It is not centrally concerned with perception, motor skill, or what

are called personality variables. The study is concerned primarily with performance, only a little with learning, and not at all with development or differences related to age. Finally, it is concerned with integrated activities, hence deemphasizes the details of processing on the time scale of elementary reactions (that is, half a second or less). Similarly, long-term integrated activities extending over periods of days or years receive no attention.

These restrictions on the scope of our study delineate a central focus, not a set of boundaries. Thus, the possibility that learning is taking place cannot be excluded in complex tasks that last tens of minutes. Likewise, it is an empirical question, not a matter of definition or fiat, to what extent perceptual mechanisms are important in problem solving—although limiting our investigation to symbolic tasks helps to restrict the importance of perception. Again, how the basic mechanisms of immediate memory and immediate processing affect integrated behavior at the next higher level is an open question. One recent author on cognitive processes (Neisser, 1967) has taken the description of the immediate processing mechanisms as the important first task, devoting only a last chapter of his book to the integrated level of behavior that is central to this volume.

Information Processing Theory

The reasons for our choices are various and largely opportunistic (although with an opportunism that has lasted twenty years and thus may constitute philosophic conviction). For several decades psychology, responding to different opportunities and convictions, focused on learning, lower organisms, and tasks that are simple from an adult human viewpoint. Within the last dozen years a general change in scientific outlook has occurred, consonant with the point of view represented here. One can date the change roughly from 1956: in psychology, by the appearance of Bruner, Goodnow, and Austin's *Study of Thinking* and George Miller's "The magical number seven"; in linguistics, by Noam Chomsky's "Three models of language"; and in computer science, by our own paper on the Logic Theory Machine.

As these titles show, the common new emphasis was not the investigation of problem solving, but rather the exploration of complex processes and the acceptance of the need to be explicit about internal, symbolic mechanisms. Nor do all four of these works stem from a specific common lineage, unless it be the whole of applied mathematics and technology: control theory, information theory, operational mathematics including game theory and decision theory, computers and programming. These topics emerged in World War II and ramified through the late forties and early fifties in many directions, the new approach to the study of man being only one.

We ourselves, through the forties and early fifties, were largely concerned with human organizational behavior and were influenced strongly by the growing technologies just mentioned. But the specific opportunity that has set the course of the present work is the development of a science of information processing (now generally termed computer science). Thus, this study is concerned with think-

ing—or that subspecies of it called problem solving—but it approaches the subject in a definite way. It asserts specifically that thinking can be explained by means of an information processing theory. This assertion requires some explanation.

The present theory views a human as a processor of information. Both of these notions—*information* and *processing*—are long-established, highly general concepts. Thus, the label could be thought vacuous unless the phrase *information processing* took on additional technical meaning.

One may try to provide this meaning by saying that a computer is an instance of an information processor. This would suggest that the phrase is a metaphor: that man is to be modeled as a digital computer. Metaphors have their own good place in science, though there is neither terminology nor metatheory of science to explicate the roles of metaphors, analogs, models, theories and descriptions, or the passage from one category to another (Simon and Newell, 1956). Something ceases to be metaphor when detailed calculations can be made from it; it remains metaphor when it is rich in features in its own right, whose relevance to the object of comparison is problematic. Thus a computer can indeed be a metaphor for man; then it becomes relevant to discover whether man is all bits on the inside.

But an alternative to metaphor is at hand. An abstract concept of an information processing system has emerged with the development of the digital computer. In fact, a whole array of different abstract concepts has developed, as scientists have sought to capture the essence of the new technology in different ways (Minsky, 1967). The various features that make the digital computer seem machinelike—its fast arithmetic, its simply ordered memory, its construction by means of binary elements—all have faded in the search for the essential. Thus, in this book we will introduce a suitable abstract information processing system to describe how man processes task-oriented symbolic information. This is not the most abstract possible way to describe an information processing system, but it is tailored to our scientific needs.

An information processing theory is not restricted to stating generalities about Man. With a model of an information processing system, it becomes meaningful to try to represent in some detail a particular man at work on a particular task. Such a representation is no metaphor, but a precise symbolic model on the basis of which pertinent specific aspects of the man's problem solving behavior can be calculated. This model of symbol manipulation remains very much an approximation, of course, hypothesizing in an extreme form the neatness of discrete symbols and a small set of elementary processes, each with precisely defined and limited behavior. This abstraction, though possibly severe, does provide a grip on symbolic behavior that was not available heretofore. It does, equally, steer away from physiological models, with their concern for fidelity to continuous physiological mechanisms, either electrical, chemical, or hormonal. Perhaps the nonphysiological nature of the theory is not as disadvantageous as one might first believe, for the collection of mechanisms that are at present somewhat understood in neuropsychology is not at all adequate to the tasks dealt with in this book. We could not have proceeded to construct theories of human behavior in these tasks had we restricted ourselves to mechanisms that can today be provided with physiological bases.

We have brought out the general grounding of our work in computer science to explain the limitation of our aims—asserting that this limitation is really an opportunity, since information processing systems provide our first precise notion of what symbols and symbol manipulation could mean. However, once we have chosen to study symbolic systems in a technical and precise way, many alternative paths still remain open along both the task dimension and the performance-learning-development dimension. Something needs to be said of our choices here.

Relation to Artificial Intelligence. The most important influence upon our choice of tasks such as chess and symbolic logic is the development of the field of artificial intelligence. This is the part of computer science devoted to getting computers (or other devices) to perform tasks requiring intelligence. As will become clear, a theory of the psychology of problem solving requires not only good task analyses but also an inventory of possible problem solving mechanisms from which one can surmise what actual mechanisms are being used by humans. Thus, one must work with task environments in which artificial intelligence has provided the requisite array of plausible mechanisms. The task areas represented in this book satisfy these conditions: game players, theorem provers, and puzzle solvers constitute a large fraction of the existing artificial intelligence systems. Many other task areas that are attractive on other grounds have not yet been studied extensively from the standpoint of artificial intelligence, hence are less useful for our purposes than those mentioned above.

On two counts the previous paragraph is insufficient to explain our choice of tasks. First, it assumes that artificial intelligence moves independently of psychology. This is demonstrably not the case. Much of the work in artificial intelligence started from psychological concerns. Second, and more important, the explanation simply raises the new question of why artificial intelligence research should take up the types of tasks represented in this book and not others.

A partial answer is that many other tasks have in fact been worked on. Language processing is an example. But while there has been a great deal of work in linguistics, almost all of it has focused on grammatical analysis (*competence*), rather than on the use of language (*performance*). Programs dealing with semantics and pragmatics are fewer, more recent, and somewhat less developed than the problem solving programs represented here. Thus, we will have little to say about language processing behavior as such. However, we shall see that our theory of problem solving has strong implications for what the linguist calls the deep structure of language.

Pattern recognition is another area in which there has been much work, both theoretical and empirical, even more extensive than the body of work upon which we draw. We make little use of this research here, because sequential, integrated behavior is at the heart of most thinking, while most artificial intelligence pattern recognition machines are built around the single act of recognition. While many

of the schemes do involve some sequential processing, none of them is adequate for modeling general sequential behavior.

There are many kinds of thinking that one might like to study: designing a house, discovering a new scientific law, preparing a law case, arguing over political parties, creating new music, daydreaming while watching the clouds, preparing a five-year economic plan, and so on. Detailed theories of these and many other kinds of thinking are largely beyond the current state of the art. Of course, there have been investigations into some of these areas, many of them still in midstream. Only their incomplete state and our limits of space and energy have inhibited us from including some of them in this work, since they are in fact part of the same story we wish to tell.

Emphasis on Performance. Turning to the performance-learning-development dimension, our emphasis on performance again represents a scientific bet. We recognize that what sort of information processing system a human becomes depends intimately on the way he develops. The kernel from which development starts—say, the neonate—already contains a genetically determined set of mechanisms of immense complexity. How complex they are is easily appreciated by anyone who follows the acts of self organization that take place in the embryo as it progresses to the neonatal starting position. Still, by common scientific assent, the emerging system is remarkably content-free, and without the powers of integrated action shown by the normal adult. Many constraints on the nature of the fully developed system arise from the requirement of self organization—help from the external environment (say via language) can only be used after the system has developed itself to a point where it is capable of such assimilation. Yet, acknowledging this, it still seems to us that we have too imperfect a view of the system's final nature to be able to make predictions from the development process to the characteristics of the structures it produces.

Similar remarks apply to learning. Humans learn continuously, and much that they do involves using in obvious ways information gathered for a specific purpose, rather than solving difficult problems like those studied in this book. One enters a department store: "Where do I find men's suits?" "Third floor, down the center aisle and to your right"; "Thank you"; and off one goes, following directions. Several phenomena here are closely allied to the interests of this book: language production and reception; deciding to ask for information to solve a problem; following directions, once assimilated; perhaps (if the directions were imperfect) solving some smallish subproblems along the way. Certainly this is the behavior of an information processing system. But the task in this case is carefully contrived to permit simple learning to substitute for most of the work of problem solving.

Learning is a second-order effect. That is, it transforms a system capable of certain performances into a system capable of additional ones (usually better ones; usually accomplished without losing much of the preexisting performance capability; and usually integrated with existing capability so they can be evoked on appropriate occasions). The study of learning, if carried out with theoretical precision, must start with a model of a performing organism, so that one can represent,

as learning, the changes in the model.¹ The mathematization of learning theory in the last decade shows this very well (Atkinson, Bower, and Crothers, 1965). In the prototype version of mathematical learning theories, the organism is represented by a set of probabilities of occurrence of a fixed set of responses; learning involves changes in these probabilities under the impact of experience.

The study of learning takes its cue, then, from the nature of the performance system. If performance is not well understood, it is somewhat premature to study learning. Nevertheless, we pay a price for the omission of learning, for we might otherwise draw inferences about the performance system from the fact that the system must be capable of modification through learning. It is our judgment that in the present state of the art, the study of performance must be given precedence, even if the strategy is not costless. Both learning and development must then be incorporated in integral ways in the more complete and successful theory of human information processing that will emerge at a later stage in the development of our science.

Other Omissions. The omission of both sensory and motor skills, and many aspects of perception, from our study is perhaps plausible on the surface: we are concerned with central symbolic activities. Nevertheless, at least one scientist, Bartlett, in *Thinking* (1958), made motor skills the key metaphor for attempting to understand thinking. And a whole school, the Gestaltists, have used perception as the touchstone of central activity (Wertheimer, 1945). On the perceptual side, a basis for our choice has already been indicated: our concern with sequential behavior. However, as even this book will reveal, the boundary between perceptual and symbolic behavior may not be maintainable for long, especially if work pushes in the direction of better models of the immediate processor.

On the motor side the situation is peculiar. Bartlett chose to compare thinking with motor skills precisely because he felt sequential behavior was central both in thinking and in motor skills, where it had been studied intensively. Our own feeling is that, while there has been much experimental investigation, motor skills have not yet found a mechanistic representation having anything like the power of the information processing representation exploited here. Furthermore, motor skills seem in considerable part to be nonsymbolic—and that makes them a poor model for a system where symbols are central.

Our final choice is to exclude motivational and personality variables—what Abelson (1963) covered in part by the term “hot cognition.” We omit them by reason of convictions, not about the importance or unimportance of the phenomena, but about the order in which theory should develop. Many motivational and emotional phenomena operate through the lens of the cognitive system, as the work of Schachter and Singer (1962) and others has indicated. A plausible scientific strategy is to put our cognitive models in order before moving to these other phenomena. Our one exploratory foray in the direction of motivation and emotion was precisely in this vein (Simon, 1967a).

¹ For empirical purposes, of course, one can always get by for a while by talking in the language of experimental protocols—simply describing the changed behavior in the experimental situation.

With this general explication of the scope of this study, we can now turn to a brief overview of the type of theory we will be presenting and the types of problems that are central to it.

THE SHAPE OF THE THEORY

The theory proclaims man to be an information processing system, at least when he is solving problems. What that means, and how it can be brought to terms with the real world, are questions to consider in extenso. However, the theory has a certain gross shape that can be characterized briefly. We say of Newtonian mechanics that it is axiomatic and deductive; of classical astronomy, that it is empirical, but not experimental; of biochemistry, that it consists in part of a large catalog of mechanisms; of intelligence testing, that it deals with the human as a static collection of traits or factors; of a part of sociology, that it deals with humans as ideal types. Such statements do not convey the content of a theory, nor how successful it is; they give an overall framework within which the pieces of the theory fit as it is assembled sequentially. The statements also suggest, sometimes, what issues will be important—what problems the theory will have to contend with repeatedly. Let us try now to provide such a picture of the shape of the present theory.

A Process Theory

The theory posits a set of processes or mechanisms that produce the behavior of the thinking human. Thus, the theory is reductionistic; it does not simply provide a set of relations or laws about behavior from which one can often conclude what behavior must be. (The elementary processes and their organization, of course, are not explained: reduction is always relative.) Thus, the theory purports to explain behavior—and not just to describe it, however parsimoniously (Newell, 1969a). (We are aware that some would dispute such a distinction, viewing all causal explanations as simply descriptions.)

The processes posited by the theory presumably exist in the central nervous system; they are internal to the organism. Nothing in *our* theory says they are in the central nervous system, but massive physiological evidence indicates that is clearly where they belong. We do not speculate on the physiological correspondences of mechanisms presented in this book. Because of the gap that still exists between physiological and behavioral science, nothing would be served by such speculation. There do not yet exist plausible physiological localizations (even in mechanism, much less in physical space) for immediate memory, the symbol (that is, the engram), or the act of adding two plus two. Without these, nothing much of physiological interest can be said about the material in this book. Furthermore, the search for such physiological mechanisms needs no motivation from the present theory (John, 1967).

As far as the great debates about the empty organism, behaviorism, intervening variables, and hypothetical constructs are concerned, we take these simply as

a phase in the historical development of psychology. Our theory posits internal mechanisms of great extent and complexity, and endeavors to make contact between them and the visible evidences of problem solving. That is all there is to it.

A Theory of the Individual

The technical apparatus for conceptualizing information processing systems leads first of all to constructing particular programs that accomplish particular tasks. When applied to psychology, this procedure leads naturally to constructing information processing systems that model the behavior of a single individual in a single task situation. Full particularity is the rule, not the exception. Thus, it becomes a problem to get back from this particularity to theories that describe a class of humans, or to processes and mechanisms that are general to all humans.

This situation is just the reverse of the one faced in earlier psychological theorizing. Indeed, a terminological line is usually drawn just to distinguish clinical efforts that deal with the individual in all his uniqueness (*idiographic*) from efforts of experimental psychology to deal with an individual only as an intersection of statistically defined populations (*nomothetic*).

The reversal of the usual emphasis gives the present theory a quite distinct flavor. Thus, individual differences is not a topic that is tacked on to the main body of our theory. On the contrary, we never use grouped data to test the theory if we can help it. The models describe individuals, so that the hard part is to say with precision what is common to all human information processors. With this approach it does not seem natural to assume that human behavior is fundamentally stochastic, its regularities showing up only with averaging (as in statistical learning theory); rather, Freud's dictum that all behavior is caused seems the natural one, and only reluctantly do we assign some aspects of behavior to probabilistic generators.

This aspect of the theory, highly visible against the contrasting background of experimental psychology, is really just a consequence of viewing the human as a complex mechanism (of whatever kind) whose parts and connections can ultimately be deciphered. This point of view is accepted in most of science outside psychology without question or comment.²

A Content-Oriented Theory

In 1955 McClelland wrote an article decrying psychology's abstraction from content. The opposition then was between content and process: by attempting only to describe basic universal processes used by humans (say, in learning), psychology was ignoring a major part of its domain.

The present theory is oriented strongly to content. This is dramatized in the peculiarity that the theory performs the tasks it explains. That is, a good informa-

² Cf. the relation of the general theory of geological processes to the specific hypothesis of continental drift, or to the character of the Moon's surface.

tion processing theory of a good human chess player can play good chess; a good theory of how humans create novels will create novels; a good theory of how children read will likewise read and understand. There is nothing mysterious in this. The theories explain behavior in a task by describing the manipulation of information down to a level where a simple interpreter (such as a digital computer) can turn the description into an effective process for performing the task. Not all versions of the theory are carried so far, of course. Nevertheless, in general, the theory can deal with the full content of a task.

The attention paid to content in information processing theories should not be contrasted with concern for process, which is how the matter appeared to McClelland. The contrast fails because it is of the nature of information processes to deal with content. It is almost tautological that one can talk of something as content only if processes exist that treat it as information—that is, discriminate on the basis of it.

Of course, there is no compelling evidence that the kinds of content that information processing theories demonstrably handle typify the entire range of content handled by humans. No information processing systems now exist that understand, say, a poem—that is, that can fully assimilate its content. Faith that this might be done goes well beyond the evidence presented in this book and elsewhere in the artificial intelligence literature. Each reader must make his own extrapolation of the evidence.

The importance of the orientation toward content is twofold. On the one hand, it removes a barrier toward extension of the theory. On the other hand, if content is a substantial determinant of human behavior—if in fact the message is a lot more message than medium—then information processing theories have opportunities for describing human behavior veridically that are foreclosed to theories unable to cope with content.

A Dynamically Oriented Theory

An information processing theory is dynamic, not in the sense in which that term is used in depth psychology, but in the sense of describing the change in a system through time. Such a theory describes the time course of behavior, characterizing each new act as a function of the immediately preceding state of the organism and of its environment.

The natural formalism of the theory is the program, which plays a role directly analogous to systems of differential equations in theories with continuous state spaces (for example, classical physics). In information processing systems the state is a collection of symbolic structures in a memory, rather than the set of values of position and momentum of a physical system in some coordinate system. Furthermore, a program generally specifies a discrete change in a single component of the state (that is, in just one symbol structure) at a moment in time, whereas a differential equation system specifies infinitesimal changes in all coordinates simultaneously. But these are details of the formalism, dictated by the underlying nature of the system under study.

All dynamic theories pose problems of similar sorts for the theorist. Funda-

mentally, he wants to infer the behavior of the system over long periods of time, given only the differential laws of motion. Several strategies of analysis are used, more or less, in the scientific work on dynamic theory. The most basic is taking a completely specific initial state and tracing out the time course of the system by applying iteratively the given laws that say what happens in the next instant of time. This is often, but not always, called simulation, and is one of the chief uses of computers throughout engineering and science. It is also a mainstay of the present work.

A second strategy is to solve the differential laws to yield an expression describing the state of the system at each point in time as a function of the initial state, represented symbolically. This is the classical act of integrating the laws of motion. It is only rarely successful for systems of any complexity—even for differential equation systems, where analytic techniques are most developed. It cannot be used as yet with theories like those in this book.

Complete detailed solutions are not always necessary (or even illuminating). The theorist of dynamic theories often spends his time trying to circumvent solving the system explicitly. One standard approach is to explore the steady state solutions of the system—the places where the laws of motion say that nothing changes. Another is to explore the asymptotic behavior—what happens in the long run. A more general tack is to prove that some property of the system is invariant over time—that it holds despite variation in other aspects of the system. In the systems of classical physics, conservation of energy, momentum, and angular momentum provide examples of such invariants.

All the above strategies show up in the study of information processing systems. Limitations on mathematical technique keep most of the analyses informal and heavily empirical,³ but the underlying motivation is dictated by the basic dynamic character of the theory.

An Empirical, Not Experimental, Theory

There is no lack of orientation towards the data of human behavior in the theory presented in this book. Yet we employ little experimental design using control groups of the sort so familiar in psychology. Because of the strong history-dependence of the phenomena under study, the focus on the individual, and the fact that much goes on within a single problem solving encounter, experiments of the classical sort are only rarely useful. Instead, it becomes essential to get enough data about each individual subject to identify what information he has and how he is processing it. This method leads, in conjunction with the content orientation, to emphasizing the use of verbal behavior as data, because of its high output rate. Thus, the analysis of verbal protocols is a typical technique for verifying the theory, and in fact has become a sort of hallmark of the information processing approach. The nature of the theory leads also to a continuing search for new

³ But see, for example, Ernst, 1969; King, 1969; Minsky and Papert, 1969; Pohl, 1969; and Robinson, 1965, for a few indications that formal analysis can be applied successfully to the kinds of IPS's considered in this book.

sources of data that can be conjoined to existing data to ease the problem of identification. The use of data on gross eye movements, briefly reported in Chapter 7, is a case in point.

A Nonstatistical Theory

It is difficult to test theories of dynamic, history-dependent systems. The saturation with content—with diverse meaningful symbolic structures—only makes matters worse. There is not even a well-behaved Euclidean space of numerical measurements in which to plot and compare human behavior with theory. Thus, this book makes very little use of the standard statistical apparatus. Theory and data are compared, and some attempts are made to measure and tabulate such comparisons. But our data analysis techniques resemble those of the biochemist or archaeologist more than those of the agricultural experimenter.

Sufficiency Analysis

The theory tends to put a high premium on discovering and describing systems of mechanisms that are *sufficient* to perform the cognitive task under study. Producing a system capable of performing provides a first approximation, taking into account, of course, gross limitations on the human's ability to process information—processing rate, immediate memory, and so on. If an information processing system meeting these constraints can be devised that does the task, one then attempts to develop a revised system that has higher fidelity to specific data on human processing.

The emphasis on sufficiency is still rather foreign to psychology. Almost never has it been asked of a psychological theory whether it could explain why man was capable of performing the behaviors in question. Concern with sufficiency arises, of course, not just in the present theory, but within the whole development of which this theory is a part. Thus, it is a current lively question in psycholinguistics whether the mechanisms of classical S-R learning theory are sufficient to account for the child's learning of language (it being concluded, not surprisingly, that they are not).

To take sufficiency as a first requirement of a theory is simply to adopt a particular approximating sequence in science's progress (a choice not without consequences, however). Since not all things can be done first, a particular theoretical orientation gets some of its flavor from what it puts first.

These characteristics of the theory, although neither systematically nor completely described, should help the reader assimilate the development of the theory through the book without being too surprised at the emergence of certain features and the (otherwise curious) absence of others. It remains to provide a plan of the book before beginning in earnest.

PLAN OF THE BOOK

The book is divided into five parts. The first lays the technical groundwork for the remainder. Each of the subsequent three parts takes up a different task environment: first a type of puzzle called cryptarithmic, then a form of elementary symbolic logic, and finally chess. The fifth and last part states the theory in comprehensive form.

Chapter 2, the first of the preparatory chapters, defines an information processing system (IPS). This is done in a somewhat axiomatic way in order to make clear what concepts are involved.

Chapter 3 considers the nature of the task environment and its role in a psychological theory. It takes up the major issue of whether a theory of problem solving (or thinking, or learning) is really a theory about human beings or a theory about the nature of the task environment. Of course, the total system always includes both environment and organism. But in a highly adaptive and intelligent organism the boundary becomes obscured. Most of the chapter is devoted to clarifying this question. A part of the chapter deals with more detailed questions of how task environments and problems are represented in information processing terms.

Chapter 4 discusses that species of information processing system that solves problems, especially problems stated in the representations introduced in Chapter 3. The notions of generate-and-test, of selective search, and of heuristics (devices that aid discovery) are introduced. We consider the assessment of task difficulty and introduce various tasks as concrete examples. A substantial fraction of the chapter is devoted to a detailed examination of the Logic Theorist (LT), a venerable program that was the first theorem-proving program. LT incorporates many of the lessons we wish to draw, and serves to consolidate the notions of the three chapters of Part I by providing a specific example of an information processing system operating in a task environment to solve problems.

Chapter 5 starts Part II, on cryptarithmic, with an analysis of the task. It exhibits various representations of the task and analyzes the behavior of various problem solvers.

Chapter 6 provides the first analysis of human problem solving. A particular subject, S3, tackles a single cryptarithmic problem, DONALD + GERALD = ROBERT. From a protocol of his verbal behavior we make an extensive analysis; then we describe a method for plotting his search behavior (the Problem Behavior Graph), and a way of representing his program as a production system (a kind of information processing system to be defined in Chapter 2). Since this is the first chapter that deals with protocol data, it contains a good deal of methodological discussion.

Chapter 7 continues the analysis of human behavior in the cryptarithmic task. We examine the behavior of four other subjects, and to do so we introduce another tool of analysis: the episode. As a final piece of evidence for the theory, we study yet one more subject, for whom we have eye movement data as well as verbal data.

Chapters 8, 9, and 10, comprising Part III, deal with a particular task in


elementary symbolic logic in the same manner as Part II dealt with cryptarithmic. Chapter 8 discusses the task environment, including a program called the General Problem Solver (GPS) that has long figured in work on simulation. Chapter 9 discusses attempts at simulating with GPS two fragments of protocols. Chapter 10 moves back to a broader base, giving analyses of various subjects working on various logic tasks, and examining a number of aspects of their behavior.

Chapters 11, 12, and 13, comprising Part IV, deal with chess. As in the two previous empirical parts, the first chapter (11) provides a task analysis; the second chapter (12) concentrates on the empirical analysis of an individual protocol; and the third chapter (13) provides a broader view of human problem solving in chess.

Chapter 14 constitutes the whole of Part V. It attempts to state/a theory of human problem solving that encompasses the phenomena that have emerged in the three separate task areas. Although the investigations in the book have been strictly limited to the areas of cryptarithmic, logic, and chess, the theory has a more substantial scope.

We have made no attempt in the body of this book to provide the historical context for our work beyond references to research on which we have drawn directly. Instead we have recorded in an historical appendix our picture of the relation of the work reported here to the broader trend of events during this century in psychology and other relevant disciplines.

This, then, is the plan we shall follow: first, technical apparatus for analyzing problem solving behavior; next, application of the apparatus to three task environments and to empirical data of human problem solving behavior in each; finally, a comprehensive statement of the theory of human problem solving.



TASK ENVIRONMENTS

The behaviors commonly elicited when people (or animals) are placed in problem solving situations (and are motivated toward a goal) are called adaptive, or rational. These terms denote that the behavior is appropriate to the goal in the light of the problem environment; it is the behavior demanded by the situation.

Now if there is such a thing as behavior demanded by a situation, and if a subject exhibits it, then his behavior tells more about the task environment than about him. We learn about the subject only that he is in fact motivated toward the goal, and that he is in fact capable of discovering and executing the behavior called for by the situation. If we put him in a different situation, he would behave differently.

Problem solving situations in which motivation is not in question and emotion is not aroused would therefore seem austere surroundings in which to study psychology. They would appear rather more suitable for investigating the structure of task environments than the nature of behaving organisms. Indeed, one modern social science, economics, has erected an impressive structure for predicting certain human behavior on the foundation of a single psychological postulate. By assuming

that economic man is always motivated to maximize his utility (or, in some applications, his profit), and that he is always able to discover and execute the maximizing behavior, we can infer the actions of economic man from the structure of the choices with which he is confronted.

By this stratagem, economics is able to avoid a concern with the psychology of decision making, and to turn, instead, to an analysis of the environment of choice—markets, cost functions, and the like. Likewise, economics finesses questions of value and preference by placing the economic actor's utility function among the given data of the problem, rather than among the behavior variables to be explained and predicted.

Now consider a standard paradigm for an experiment in concept attainment. A sequence of stimuli is presented to a subject, who is asked to classify each as an instance, or noninstance, of a concept (as yet unknown to him). He is informed whether each reply is right or wrong. The conditions of the experiment are carefully arranged so that the subject will be motivated to perform the task well—say, to guess the concept from as few stimuli as possible. If the conditions of the experiment do not achieve appropriate motivation—if the subject refuses to try the task, is obviously inattentive, or undertakes to spoof the experimenter—we would not regard it as an experiment in concept attainment. We would also reclassify the experiment if it turned out that the subject could not discriminate among the stimuli because the light was too dim or because his eyes could not resolve the differences among them. In these cases, we would call it an experiment in visual sensation or, possibly, in perception.

For the properly motivated subject, however, there presumably exists an optimal strategy—a strategy that will disclose the concept in the minimum expected number of trials. (We presuppose that a set of possible concepts is given—this is usually defined by the experimental instructions and the situation.) If the situation is not too complicated, the optimal strategy can sometimes be deduced by means of principles drawn from information theory (Hovland, 1952). It would be perfectly possible for the psychologist to follow the route of the economist: to construct a theory of concept formation that depended on no characteristic of the subject other than his being adequately motivated to perform well. It would be a theory of how perfectly rational man would behave in that task environment—hence, not a psychological theory but a theory of the structure of the task environment.

In a few cases psychology has in fact begun to explore this route. The applications of statistical decision theory to experiments on choice under uncertainty fit precisely into the mold of formal economic thinking (Edwards, Lindman, and Phillips, 1965). In psychology, however, the deductions from premises of rationality about how rational man *would* behave are almost always accompanied by experiments on how laboratory man *does* behave. The discrepancies between behavior and predictions from the rational model are often large; they tend to vanish only when the rational behavior is transparently obvious, and not always even then. (We may well ask, "Obvious to whom?" If the subject does not behave rationally, we may take this fact as the definition of the rational behavior's not being obvious to him!)

It is precisely when we begin to ask *why* the properly motivated subject does not behave in the manner predicted by the rational model that we recross the

boundary again from a theory of the task environment to a psychological theory of human rationality. The explanation must lie inside the subject: in limits of his ability to determine what the optimal behavior is, or to execute it if he can determine it. In simple concept attainment experiments, for example, the most important mechanism that prevents the subject from adopting an efficient strategy is usually the limit on the number of symbols he can retain and manipulate in short-term memory. To the extent that this is true, such experiments are experiments to reveal the structure of human short-term memory, and it is for this reason that *cognitive strain* plays such a central role in the Bruner-Goodnow-Austin (1956) explanation of concept attainment phenomena.

In summary, then, when we study a properly motivated subject confronted with an intellectual task, we are observing *intendedly rational behavior* or behavior of *limited rationality* (Simon, 1947). Our examination of such behavior will lead to two kinds of knowledge:

1. To the extent that the behavior is precisely what is called for by the situation, it will give us information about the task environment. By observing the behavior of a grandmaster over a chessboard, we gain information about the structure of the problem space associated with the game of chess.
2. To the extent that the behavior departs from perfect rationality, we gain information about the psychology of the subject, about the nature of the internal mechanisms that are limiting his performance.

Just as a scissors cannot cut paper without two blades, a theory of thinking and problem solving cannot predict behavior unless it encompasses both an analysis of the structure of task environments and an analysis of the limits of rational adaptation to task requirements. We shall attempt such a two-bladed theory here, but we shall have to place bounds on the undertaking. A complete theory of task environments would have to cover all of human knowledge—natural science, practical arts, games, fine arts, and what not. We have no stomach (yet) for such a quixotic undertaking. Nevertheless, we will devote much more attention throughout the book to the analysis of the task environment than is customary in psychological works.

The term *task environment*, as we shall use it, refers to an environment coupled with a goal, problem, or task—the one for which the motivation of the subject is assumed.¹ It is the task that defines a point of view about an environment, and that, in fact, allows an environment to be delimited. Also, throughout the book, we shall often distinguish the two aspects of the theory of problem solving as (1) demands of the task environment and (2) psychology of the subject. These shorthand expressions should never seduce the reader into thinking that as a psychologist he should be interested only in the psychology of the subject. The two aspects are in fact like figure and ground—although which is which depends on the momentary viewpoint.

¹ There could of course be several such goals; the complication can be ignored here.

Because the task environment plays such a large role in the theory of problem solving, this chapter examines in detail a number of issues relating to it. The first section discusses how the external environment is to be represented. The second section considers how the subject represents a problem internally, and introduces the concept of a problem space, which we will use throughout the book. These two sections, then, focus on very general issues of the relations between experimenter and environment (i.e., the task environment) and the subject (i.e., his problem space).

The chapter's third and fourth sections carry further the treatment of problems and problem spaces. The third section draws upon the description of an information processing system provided in Chapter 2 to show, by example and in some detail, how a problem space is to be represented in an IPS, and how the processes of the IPS operate on such a representation to solve problems. The fourth section moves from the detail of the examples to a more general characterization of problem spaces, introducing two very broad classes of problem representations: set representations and search representations. This classification will provide the foundation for a discussion, in the next chapter, of the dependence of the structure of problem solving methods upon the structure of the problem space.

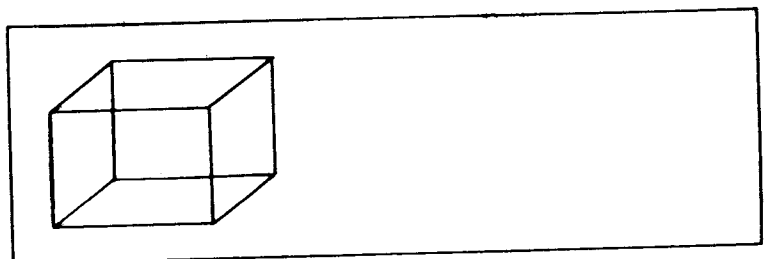
The chapter's fifth, and final, section returns from viewing internal representations—problem spaces—to considering again what it means for a task environment to impose demands upon a problem solver. The conclusions from this discussion will be important in justifying the techniques we use in later chapters to abstract from detail when we describe task environments and when we compare subjects' behavior with the behavior demanded by the environment.

REPRESENTATION OF THE EXTERNAL ENVIRONMENT

In talking about the task environment we must maintain clear distinctions among the environment itself (the Kantian *Ding an sich*, as it were), the internal representation of the task environment used by the subject (the problem space), and the theorist's "objective" description of that environment. This is the classical problem in psychology of defining the effective stimulus.

Consider a typical demonstration in perception—the Necker cube. A two-dimensional geometric figure drawn on a sheet of paper (Figure 3.1) is shown to the

FIGURE 3.1
Necker cube



subject, and he is asked, essentially, to interpret it in three dimensions (translate: to construct an internal representation of the stimulus as three-dimensional). For almost all subjects, two such representations are possible—there are two ways of interpreting the stimulus on the retina as if it were produced by a three-dimensional object “out there in the real world.” In fact, of course, neither interpretation is veridical, for the experimenter knows (as does the subject) that the geometric object producing the retinal stimulation is “really” two-dimensional. Now the distinction we wish to maintain is the distinction between the way or ways the subject interprets the Necker cube, and the actual stimulus provided by the experimenter.²

In the case at hand, and many like it, there is no serious difficulty in describing the stimulus itself as it impinges on the sense organs of the subject. In the case of the Necker cube, all that is required is to describe (or better, to exhibit) the two-dimensional figure, to state (or exhibit) the conditions under which it is presented to the subject (size, distance, illumination, and so on), and to quote (or exhibit) the verbal instructions that are given to the subject.

Even in this instance, the amount and fineness of detail required for veridicality can in the last analysis only be determined empirically. For example, small imperfections in the figure—slight departures from linearity, or nonuniform color of the surface—might, under some circumstances, turn out to be important. The experimenter’s intonations in reading the instructions to the subject could sometimes be relevant. Any description of the stimulus—short of exhibiting the actual experimental setup—involves abstraction, hence the possibility of omitting detail that affects the subject’s behavior. (Even total exhibition involves abstraction—it just pushes the responsibility off on the observer.) Of course this difficulty, while present in principle, is not always of practical importance. In practice, the Necker cube stimulus can easily be described without leaving out aspects of the situation that will turn out to affect significantly the responses of the subject.³

If the stimulus had been, say, a van Gogh painting instead of a Necker cube, no veridical verbal description would have been possible. Here, exhibiting the actual stimulus, under the actual lighting and other conditions of the experiment, would be the only way of guaranteeing that it was properly characterized. For purposes of some experiments a good color photographic reproduction of the painting might suffice, but it is easy to conceive of experiments where such a reproduction would be a wholly different stimulus from the painting itself. And even if the color photograph were the actual stimulus, we would still have no satisfactory way of describing the stimulus except by reproducing it accurately. It could not be described in words.

It is instructive to try to explain *why* we can describe some stimuli verbally, but not others—why we can describe the Necker cube, but not a van Gogh painting

² We are using the Necker cube in the service of a methodological discussion. There is also a large literature on the psychology of the Necker cube, some of which contributes directly to the points made here (e.g., Gregory, 1966; Simon, 1967b).

³ We can use this criterion as the basic test for the veridicality of the description of a stimulus: it is veridical if a stimulus built from the description (by one skilled in the art) produces the same behavior in subjects as the original stimulus did.

or even a reproduction of one. A necessary condition for a stimulus to be describable is that it can and will be encoded in highly simplified form by any subject exposed to it. If we know the language that will be used in the encoding, then we also have sufficient conditions for a description, for we can describe the stimulus in that same language. But if different subjects encode the stimulus in different languages, we cannot construct a single description that is guaranteed to be veridical for all; for a stimulus reconstructed from the description may alter properties of the original stimulus that are ignored by the language of the description but are not ignored by other encoding languages subjects may use.

In the case of the Necker cube, for example, we can describe the stimulus adequately on the assumption that each subject will handle the figure as an arrangement of line segments and plane figures bounded by line segments. We must assume that if there are very slight irregularities in the heaviness or directions of the lines, he will ignore them, retaining and using only the information that a certain distribution of ink is a straight line joining points A and B.

The vast body of experience—in the laboratory and in everyday life—with subjects' interpretations of line drawings gives the experimenter assurance that his descriptions of such stimuli (i.e., describing them as line drawings) are veridical. If he were to experiment with subjects from other species than his own (or perhaps even from other cultures), he would have less reason for assurance. If the subject were a frog, for instance, the experimenter would be rash to describe the stimulus in terms of lines (Lettvin, Maturana, McCulloch, and Pitts, 1959). All sorts of alternative encodings are conceivable. The difficulties that have been encountered in designing pattern-recognizing devices for two-dimensional stimuli provide concrete evidence that representation of the stimuli in terms of lines requires a nontrivial encoding scheme. Only a subject whose inborn physiological mechanisms provide such a scheme, or who has acquired one by learning, will perceive a Necker cube as made up of a few line segments.

What we have said about visual stimuli applies equally well to verbal stimuli, including experimental instructions. A printed report of verbal instructions will be a veridical description of the stimulus only if the sound waves striking the subjects' ears when the instructions are read to them will be encoded as words in the language of the printed description. (Of course, veridicality requires more than this—in particular, that inflections and intonations omitted from the printed version do not affect the subjects' behaviors significantly.)

To say, then, that the stimulus can be described adequately means that we can predict the vocabulary of elements and relations the subject will use in his initial encoding of it. The description no longer refers exclusively to the stimulus; it postulates something also about the first stages of processing that the stimulus will undergo as it is perceived by the subject. This does not mean, of course, that the subjects' encodings are verbal—that the word "line" appears in a subject's encoding of the Necker cube. It does mean that the subject's initial encoding of the Necker cube stimulus must be composed of elements, however represented, that correspond to the lines in the drawing.

It need not be assumed, however, that the initial encoding of the stimulus will bear any close relation to the way in which it is represented internally in the subject's processing of it at some later stage. The stimulus may—and usually will—be

subjected to further transformations as the subject seeks a convenient internal representation—one that he can process relatively easily. (E.g., the subject transforms the Necker cube internally into a representation of a 3-dimensional object.) Our next section will examine these further transformations into the problem space, the way in which the subject represents the problem internally once he has perceived the stimulus.

INTERNAL REPRESENTATIONS: THE PROBLEM SPACE

Even more difficult than describing the stimulus is finding a neutral and objective way of talking about the responses of the subject, including his internal thinking responses, as he goes about dealing with the stimulus situation. Our description of his behavior will necessarily be molar rather than molecular—we will not be tempted to refer to the space-time coordinates of his body parts and the sound waves emanating from his mouth. But that is not the main source of difficulty. We shall find it necessary to describe not only his actual behaviors, but the set of possible behaviors from which these are drawn; and not only his overt behaviors, but also the behaviors he considers in his thinking that don't correspond to possible overt behaviors. In sum, we need to describe the space in which his problem solving activities take place. We will call it the problem space.

This is not a space that can yet be pointed to and described as an objective fact for a human subject. An attempt at describing it amounts, again, to constructing a representation of the task environment—the subject's representation in this case. The subject in an experiment is presented with a set of instructions and a sequence of stimuli. He must encode these problem components—defining goals, rules, and other aspects of the situation—in some kind of space that represents the initial situation presented to him, the desired goal situation, various intermediate states, imagined or experienced, as well as any concepts he uses to describe these situations to himself.

Example of an Internal Representation

A specific example will help delineate the exact nature of the difficulty, and will provide a starting point for our discussion of how to obviate it. We consider, as our problem situation, a two-person game (*number scrabble*) having the following rules: A set of nine cardboard squares (pieces), like those used in the game of Scrabble, is placed, face-up, between the two players. Each piece bears a different integer, from 1 to 9, so that all nine digits are represented. The players draw pieces alternately from the set. The first player who holds any subset of exactly *three* pieces, from among those he has drawn, with digits summing to 15, wins. If all the pieces are drawn from the set without either player obtaining three whose digits sum to 15, the game is a draw. Thus, if the alternate draws are 2, 7; 5, 8; 4, 6; 9, the first player wins, since $2 + 4 + 9 = 15$. If the alternate draws are 5, 2; 8, 6; 7, 3; 1, 9; 4, neither player wins, since no combination of three digits selected from

the set (5, 8, 7, 1, 4) sums to 15, nor does any combination of three selected from the set (2, 6, 3, 9).

The description, just given, of number scrabble constitutes a description of the stimulus confronting each player that is adequate for all practical purposes. It is objective in the sense that it simply (if only grossly) describes the visual and aural stimuli actually presented to the subjects when the game is explained to them. But the subject's task environment contains more than just this stimulus. It contains also the various possibilities that exist for playing the game—that is, for changing the stimulus. How can we represent these possibilities objectively?

One possible representation is provided by the game tree. Every possible play of the number scrabble game can be represented by a permutation of the nine digits. We can represent the set of permutations as a tree, arranged according to the first digit selected; then, among those branches having the same first digit, arranged according to the second digit; and so on. The tree contains a total of $9! = 352,880$ terminal branches. A few initial branches are shown in Figure 3.2. (Some of these branches are inessential, in cases where the game would end before the ninth move.) Since the game is a game of perfect information with a finite (large but not enormous) game tree, an optimal strategy can in fact be devised with only a finite number of calculations. The strategy is based on the well-known minimax principle of game theory. The player simply examines all the branches of the game tree, and assigns to each terminal the value, to him—win, draw, or lose. He then works backward, assigning to each branch the best of the outcomes (win, draw, or lose) for the subbranches at nodes where it is his turn to move, and the worst of the outcomes for the subbranches at nodes where it is his opponent's turn to move.

But in examining internal representations are we safe in limiting ourselves to the alternatives contained in the game tree? Consider the following situation. The players have made the initial moves 3, 5; 9, 7. The first player now calculates what his next move shall be. He has already drawn the digits 3 and 9, which together total 12. A winning strategy would be to draw the number which, when added to this total, yields 15. The appropriate move for this strategy is readily calculated: it is $15 - 12 = 3$. So the first player wishes to consider the winning sequence 3, 5; 9, 7; 3.

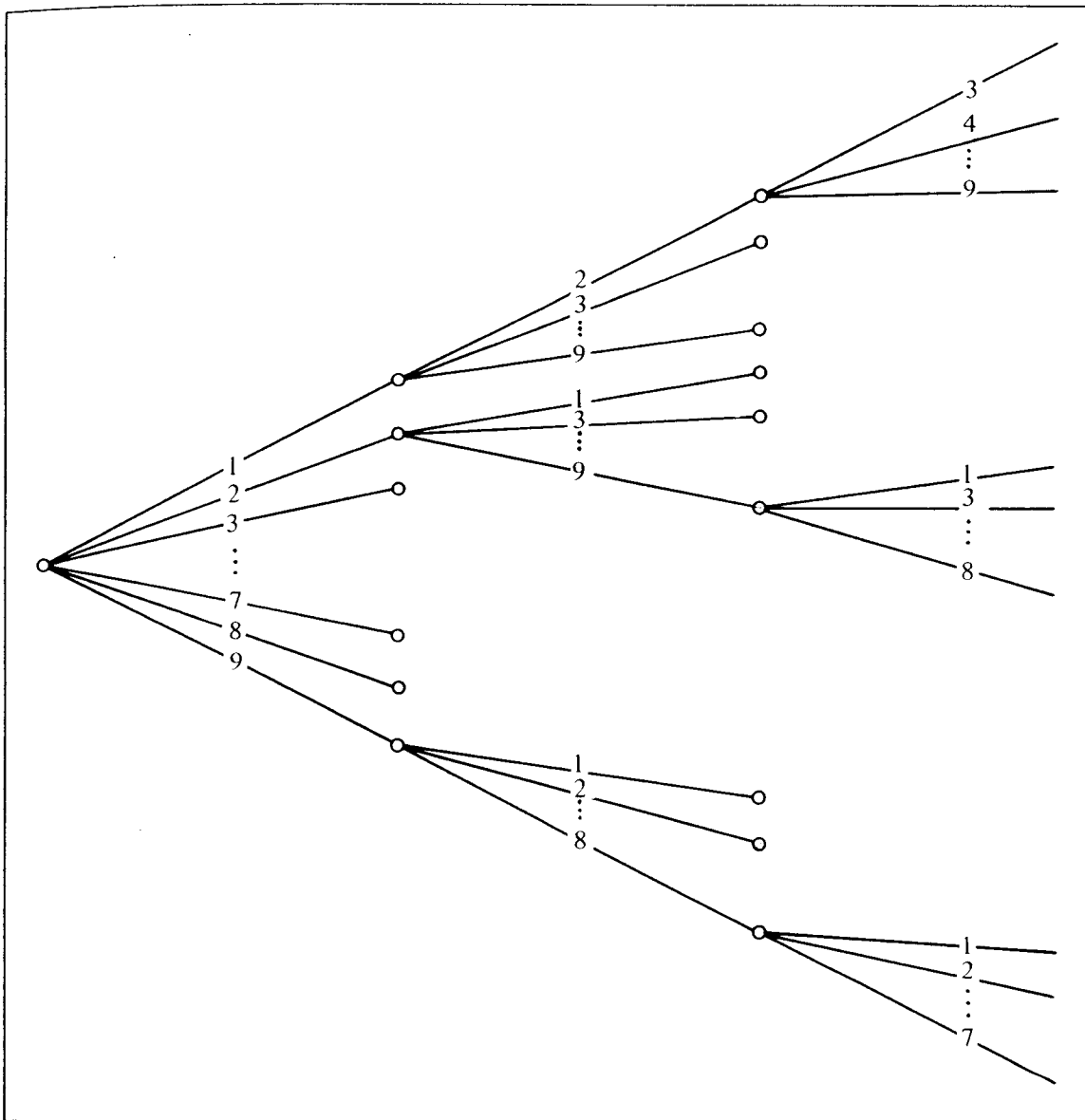
It will be objected immediately (and correctly) that 3 is not a legal move here, because a 3 has been drawn previously, and each digit appears on only one piece. Presumably the player will check for the legality of his move, and will discover that 3 is not available. (If actual physical pieces are used, he will discover it, if in no other way, from his failure to find 3 among the pieces remaining in the center.)

Nevertheless, we must not exclude the possibility that the player will actually carry through the calculation of subtracting 12 from 15, get 3 as his result, and only then discover that the move 3 is not available. If we are to construct a theory of the subject's thought processes, and to refer in that theory to the set of behaviors he considers, we must include in that set even behaviors that, though considered, prove infeasible, illegal, or in some other way impossible. We must, so to speak, represent the subject's wishes and dreams as well as his more realistic thoughts. There is no essential reason, of course, why such considerations should remain covert. If, in our example, the player said, "Oh, there is no 3," we would thereby have concrete evidence of his having gone outside the game tree.

TASK ENVIRON
includes:
1) stimuli = objects
2) actions!

A: No
↓

FIGURE 3.2
game tree for number scrabble



Recoding of Representations

The situation where the subject considers alternatives beyond those included in the game tree or other representation of the genuine behavior possibilities is not the only one that makes it difficult to describe the task environment objectively. Another whole set of difficulties arises out of the fact that the subject may recode the situation completely.

Consider the familiar game of tic-tac-toe. There is a three-by-three array of blank squares. Players occupy squares alternately, marking the square occupied by a \bigcirc or \times , respectively. The first player who attains a horizontal, vertical, or diagonal sequence of three of his symbols wins. If the whole array is filled without either player's attaining such a sequence, the game is a draw.

Now it is true, although not obvious, that the game of tic-tac-toe is formally

equivalent to the game of number scrabble that we described earlier. (We wonder how many readers, almost all of whom have surely played tic-tac-toe, were aware of this equivalence before we mentioned it.) To see this, we construct the magic square whose rows, columns, and diagonals each sum to 15, as shown in Figure 3.3.

FIGURE 3.3
magic square for tic-tac-toe

2	7	6
9	5	1
4	3	8

If we played tic-tac-toe on this magic square, whenever a player attained a horizontal, vertical, or diagonal row, he would possess a set of three numbers adding to 15. The construction of the magic square assures this. The converse is also true, as can be verified by enumeration: any player occupying three squares that add to 15 will find that these squares lie in a horizontal, vertical, or diagonal row. Thus, the formal equivalence of the two games has been proved.

To say that the two games are formally equivalent does not say that a person who knew a good strategy for playing one of them would be capable of transferring the strategy immediately to the other. Tic-tac-toe experts will readily persuade themselves that this is not at all easy.

The following is generally regarded as a good strategy for the first player in tic-tac-toe. Since the game is a draw when viewed from a game-theoretic standpoint, *good* means here a strategy that will guarantee a draw and that will give the opponent as many opportunities as possible of making a losing mistake. We outline the strategy as a production system (Figure 3.4). We do not show details of the patterns to be detected on the board, which are familiar to all tic-tac-toe players. $\langle \text{own-winning-pattern} \rangle$ is a line with two of the player's marks and one blank. $\langle \text{opponent-winning-pattern} \rangle$ is the same, but with the opponent's marks rather

FIGURE 3.4
production system for playing tic-tac-toe

tic-tac-toe-strategy:

1. side-to-move = opponent \rightarrow stop.
2. $\langle \text{own-winning-pattern} \rangle (\Rightarrow \text{blank-square}) \rightarrow \text{play (blank-square)}$.
3. $\langle \text{opponent-winning-pattern} \rangle (\Rightarrow \text{blank-square}) \rightarrow \text{play (blank-square)}$.
4. $\langle \text{own-forking-pattern} \rangle (\Rightarrow \text{intersection-square}) \rightarrow \text{play (intersection-square)}$.
5. center = blank \rightarrow play (center).
6. $\langle \text{opponent-on-side} \rangle \rightarrow \text{find corner} = \text{blank}; \text{play (corner)}$.
7. $\langle \text{opponent-on-corner} \rangle \rightarrow \text{find opposite of corner}; \text{play (opposite)}$.

than the player's. \langle own-forking-pattern \rangle contains two lines, each with one of player's mark and two blanks, intersecting in a single blank square. Playing in the intersecting square creates two \langle own-winning-pattern \rangle 's, thus forking the opponent.

How would a player, knowing this tic-tac-toe strategy, play number scrabble? We assume the players are not allowed pencil and paper to assist their memories. If the player is gifted with a certain amount of visual imagery, he can proceed as follows. He memorizes the array of numbers in the magic square. Then as he, or his opponent, draws a numbered piece in the number scrabble game, he visualizes the corresponding \bigcirc or \times on the square. In this way, he can detect two-in-a-row situations, and can distinguish center, side, and corner locations as required in order to apply the strategy. We have verified that this is a feasible method for at least some subjects.

An alternative method would be to translate the tic-tac-toe strategy into the language of the number scrabble game, and learn to apply the new strategy directly in that game. The translation of the strategy is fairly straightforward. *Winning move* is redefined as *triplet adding to 15*, *blocking* as *taking the number needed by the opponent to complete such a triplet*, *forking move* as *move creating two pairs each of which belongs to a triplet still completable*. For *center square*, we read 5; for *corner square*, even number; for *side square*, odd number other than 5. Finally, we translate *square opposite* as *ten's complement of*—i.e., the number, x , such that $\text{square} + x = 10$. Strategy for tic-tac-toe can be translated word-for-word into the equivalent strategy for number scrabble. For example: "If the opponent then occupies a corner square, occupy the opposite corner" becomes "If the opponent then takes an even number, take its ten's complement." We leave the rest of the translation as an exercise for the reader.

Finding an Objective Representation

The games of number scrabble and tic-tac-toe provide us with an example of a very simple problem situation where subjects can (and occasionally do) represent the task internally in quite different ways. How shall the experimenter talk about the task? Since the two games are formally equivalent, it is not clear which, if either, of the descriptions we have presented should be chosen to represent the problem space. Any player could use either representation (or some other) when playing either game. He could use a combination of them—for example, representing number scrabble as tic-tac-toe on a magic square, as we outlined earlier. And, as we have also seen, he can use a representation that permits him to consider moves that are not legal moves in the actual game. It would seem that the experimenter's description of the task environment, whatever representation he uses for that description, would be no less subjective than the player's. No particular description appears to have claim to complete or exclusive veridicality.

Rather than try to untie this Gordian knot, we might ask whether we can cut it. Perhaps it is not necessary, in constructing and testing a theory of problem solving, for the experimenter to use an explicit representation of the task environment. What are some of the other possibilities?

often do as task becomes more complex.

One possibility is to omit a description of the objective problem space, but to incorporate in the theory hypotheses about the internal representation the subject will himself use for the problem space. This possibility exists because the behavior of the subject cannot depend on what the problem really is—neither he nor we know that—but only on what the stimulus is. Hence it is the stimulus, including the feedback the subject receives from the environment as a result of his behavior in the problem situation, not the real problem, that should enter as an independent variable if we wish to predict the representation. Even in situations where we wish to predict how differences in problem statements affect ability to solve the problem, we need not describe the external task environment, but only the stimulus and feedback rules.

A somewhat different possibility, a pragmatic one in this case, is to construct a hypothetical problem space that is objective only in the sense that all of the representations that human subjects in fact use for handling the problem can be imbedded as specializations in this larger space. This solution operates as follows: Consider a large number of subjects placed in a task environment and allowed to study and analyze that environment over a long period of time. Presumably, they will devise a number of representations for characterizing the environment, and the representations they use will change over time. If we allow enough time, the rate at which they generate new representations will gradually decrease, and presently additional ones will appear rarely if at all. We shall equate the real environment with the whole set of representations generated up to the time when this asymptote is reached, together with the mappings that define the relations of each of the representations to the others. With this strategy the experimenter knows, for all practical purposes, this entire set of representations and mappings.⁴

What does this latter requirement amount to from a practical standpoint? First, it is important only in situations where subjects do, in fact, employ a number of internal representations, and where the theory is concerned with which representation they employ, or how they acquire a representation. Of course, most situations that are interesting for a theory of thinking and problem solving have these characteristics.

Second, the requirement can be satisfied approximately by studying situations where the complexity is great relative to the time available to subjects for analyzing it. Then the experimenter, even if he is no more intelligent than his subjects, can meet the requirement by devoting much more time to the analysis of the situation than is available to any subject (or alternatively, by withholding from them some of the information that is available to him about the structure of the environment).

In the studies in this book, we conform reasonably with this strategy for two of our tasks, cryptarithmic and elementary logic, but fall short with the third, chess. We are, in fact, somewhat handicapped in studying the behavior of masters and grandmasters in chess, since we cannot attain a better understanding of the task environment than such subjects.

⁴ It is conceivable that for a formally given IPS one could develop a formal theory of all representations of a given task. However, no such theory has yet come forth for any interesting task environment.

Since the theories that will be discussed in this book include in their structure explicit descriptions of the central processes and memory contents of the subjects, these theories, in effect, assert what the subjects' representations will be. To the extent that different problem spaces would lead to different behavior in the face of the experimental task, these assertions are empirically testable.

Let us consider a simple example from the number scrabble game. Suppose we were to hypothesize that a particular subject was using the magic square representation in choosing his moves—that is, that he visualized his choices, and his opponent's, as lying in rows and columns in the square. Now suppose that the subject, having already drawn the numbers 9 and 3, said something like: "That makes 12. I need 3 to make 15," and then searched for a 3 among the pieces remaining to be chosen. This behavior would provide rather conclusive evidence that the hypothesis was wrong—that the subject was not using the magic square representation, but some kind of arithmetic problem space. For 9 and 3 do not lie on a row, column, or central diagonal of the magic square; hence a subject using that representation would not treat those numbers as a possible basis for a winning triad.

In later chapters of this book we shall present various examples of inferences that predict the behavior of the subject from his (hypothesized) representation, and then test the prediction by comparing it with the actual behavior. Hypotheses about internal representations will, in fact, be a major source of the predictive power and parsimony of our theories. For the assumption that a particular problem space is being used generally has many consequences for behavior, hence permits many predictions. Moreover, if we are willing to assume (1) that the number of representations available to human subjects is not enormous—is, in fact, quite modest—and (2) that the same representations are used in performing a considerable range of tasks, then the parsimony of the theories is greatly enhanced.

Representations and Linguistic Deep Structure

We are now in a position to see more clearly the meaning of the assertion in the previous chapter that the internal representations employed by the IPS are to be regarded as the deep structures postulated by modern structural linguists. Language is usually discussed in a context of interpersonal symbol processing—speaking (writing) and listening (reading). But there has also been much inconclusive discussion in psychology about the role of language in internal symbol processing—in thinking. John Dewey (1910, p. 170) has described the alternative views as well as anyone:

Three typical views have been maintained regarding the relation of thought and language: first, that they are identical; second, that words are the garb or clothing of thought, necessary not for thought but only for conveying it; and third . . . that while language is not thought it is necessary for thinking as well as for its communication. When it is said, however, that

thinking is impossible without language, we must recall that language includes much more than oral and written speech. Gestures, pictures, monuments, visual images, finger movements—anything consciously employed as a *sign* is, logically, language.

The difficulties, we submit, in choosing among these views, stem from ambiguity of the term “language.” The position implicit in the analysis of this book can be summed up as follows:

1. The generation or processing of symbol structures that are isomorphic with the strings of natural language or with their surface structures (in the linguist’s meaning of that phrase) is inessential to human problem solving of the kinds we examine.
2. The internal symbol structures that represent problems and information about problems are synonymous with the linguist’s deep structure. If “language” means deep structure, then language is essential to thinking and problem solving.

In sum, paraphrasing Dewey: (1) The surface structure of language, and language strings, are the garb or clothing of thought, necessary not for thought but only for conveying it. (2) While linguistic deep structure is not thought, it is necessary for thinking.

To the extent that the problem representations we shall postulate do, in fact, explain the problem solving behavior of human beings, the claim that the structures embodying these representations are linguistic deep structures gains strong support. For if this were not so, *and* if language is implicated in thinking, then we would have to postulate a distinct set of deep structures, holding the meanings of the surface structures, carried along by the IPS in parallel with the “nonlinguistic” problem representations. But this is an unnecessary and unparsimonious multiplication of hypothetical entities that has no evidential support. We prefer the simpler course that identifies deep structures closely with semantics, hence with the internal symbol structures that are the media of thought.

The theories of deep structure that have been sketched by linguists (up to the present time no more than sketches are available) generally associate individual deep structures with individual sentences of the surface strings. In contrast, the internal structures we shall postulate for problem solving situations generally constitute large, complex, interrelated contexts that do not factor in any simple way into components that are isomorphic with single sentences. These complex internal structures, extending far beyond sentence boundaries, provide a promising route toward explaining how context can determine the meanings of otherwise ambiguous natural language strings. The exploration of this route, however tempting, cannot be undertaken in this book. The early work by Lindsay (1961, 1963) gives some general indication of what we mean by this brief allusion, and provides the reader with a starting point if he wishes to undertake the exploration himself.

THE CONTENT OF A REPRESENTATION

In developing our tic-tac-toe and number scrabble examples of possible problem spaces, we have already been drawing implicitly on the assumption that the subjects in our experiments behave like information processing systems of the sort that we described in Chapter 2. We need now to consider in more detail how an IPS can store and process problem space representations. The game of number scrabble again provides a simple example of how this can be done.

Our concern at present is not with hypothesizing specifically how human subjects in fact represent this problem, but with providing a concrete illustration of how an IPS can represent it—how it can encode the elements of the problem space into symbol structures, and how it can operate on these structures to accomplish the processing required to play the game.

Objects will be represented by symbols; *sets of objects* will be represented by lists. The *relations* between various objects and sets of objects will be represented by associations. *Moves* will be represented by processes, which modify the internal symbol structures in correspondence with the way the moves would modify the external sets of objects or their relations. In addition, for the IPS to play, other processes are needed to test the representation for certain features and to embody various strategies of play.

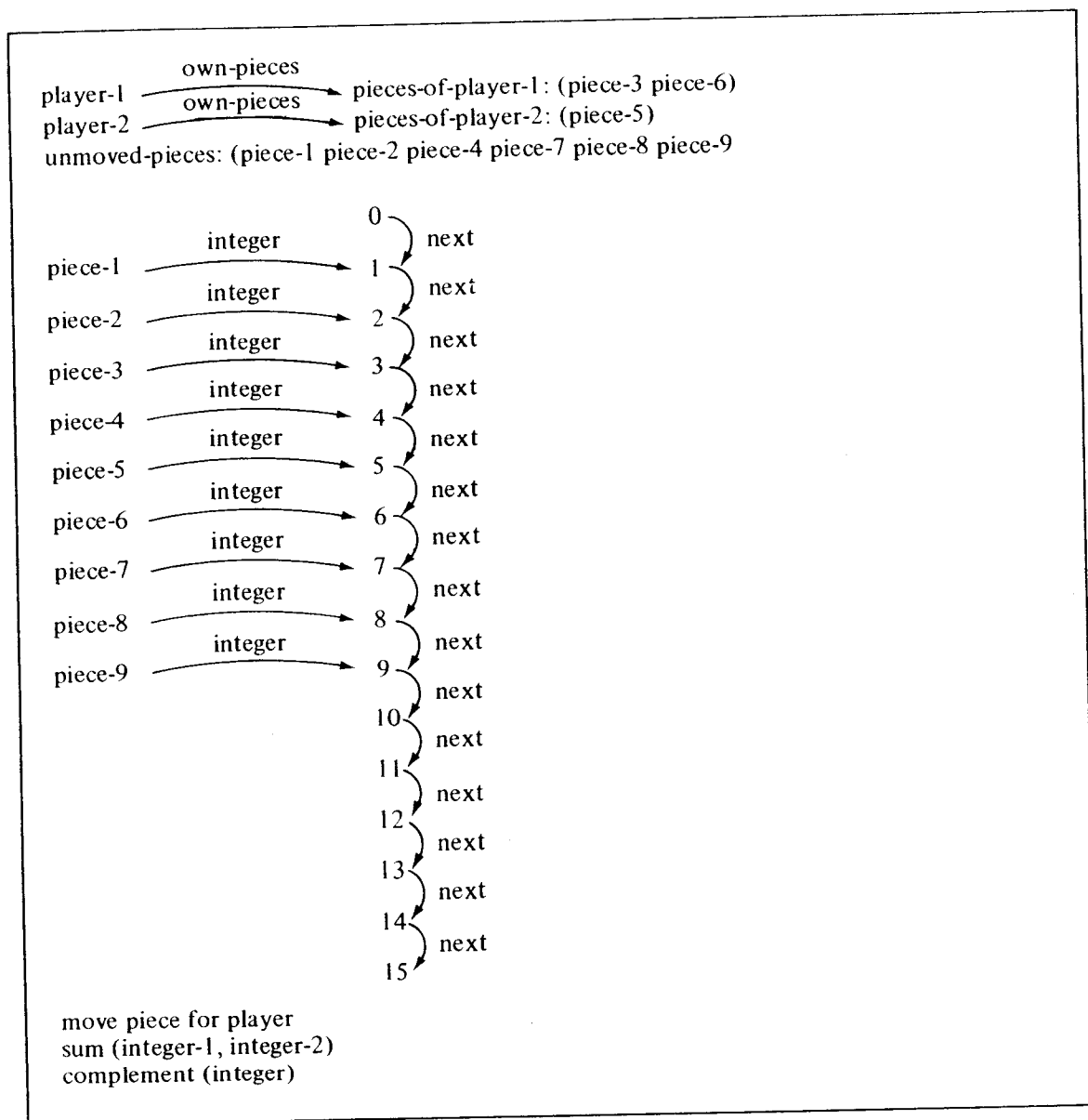
Figure 3.5 gives the details of the representation. The basic objects are players (two) and pieces (nine). There are three sets: *unmoved-pieces*, *pieces-of-player-1*, and *pieces-of-player-2*. Each player has an association (attribute: *own-pieces*) to the set of his pieces. Each piece has an association (attribute: *integer*) to the name of the integer (from 1 to 9) on that piece.

Thus, at the beginning of the game, the sets of *pieces-of-player-1* and *pieces-of-player-2* are empty, while the set of *unmoved-pieces* contains all nine of the pieces. A move is a process that removes one piece from the list of *unmoved-pieces* and adds it to the list of *own-pieces* of the player who is moving. Besides the basic process for moving, some processes are needed for manipulating integers: *sum*, which adds up two integers to produce a third (their sum); and *complement*, which takes an integer as input and produces the difference between it and 15 as output (thus, more accurately, *15's-complement*).

The three processes listed in Figure 3.5 are taken as primitive, but they can also be expressed as programs in terms of still more primitive processes for manipulating and testing lists and associations. In Figure 3.6 we give programs for all three. The arithmetic is of the “counting fingers” variety, since we do not wish to introduce a full array of arithmetic processes; still, it gets the job done. The processes used in Figure 3.6 are truly primitive, in that we have not introduced any other representation in terms of which we could discuss how they are realized.

With this representation of number scrabble we can express how a player could discover, for example, whether he had an immediately winning move. Figure 3.7 shows the program. We have used a generator of all pairs from a set. This produces, as element, a list of two symbols. Again, we could take this as primitive (as we have taken *generate* as primitive). However, Figure 3.8 does show a process for realizing *generate-pairs*.

FIGURE 3.5
symbol structures and primitive processes for tic-tac-toe



In Figure 3.7 it might appear costly, in terms of processing time, to have to repeat the subprocess (lines 2–6) for each pair on the list of own-pieces. However, this list would never have more than three members (if the player has already made four moves, his fifth move is forced); hence, there would never be more than six such pairs. Actually, a more efficient process would consider only pairs that contain the most recently moved piece, since the others would already have been considered on a previous move. With this added refinement, three pairs, at most, would have to be examined.

On the other hand, with this representation, somewhat more processing is required to determine if the player has a forking move—i.e., a move that sets up two possible winning moves for the next round, hence cannot be countered by the opponent. Figure 3.9 gives the program. The processing cost for finding forking moves in this way may be large. Three generators are nested: the first generates pairs of pieces from the list of own-pieces, the second and third generate pieces from

FIGURE 3.6
basic processes for number scrabble

```
move piece for player:
1. delete piece from unmoved-pieces,
   if fail stop move and report illegal;
2. find own-pieces of player;
3. insert piece on result.

sum (integer-1, integer-2):
1. initialize count to be 0;
2. initialize total to be integer-1;
3. test if count = integer-2,
   if true stop and report total;
4. find next of count ( $\Rightarrow$  count);
5. find next of total ( $\Rightarrow$  total),
   go to 3.

complement (integer):
1. initialize count to be 0;
2. initialize total to be integer;
3. test if total = 15,
   if true stop and report count;
4. find next of count ( $\Rightarrow$  count);
5. find next of total ( $\Rightarrow$  total),
   go to 3.
```

FIGURE 3.7
program for finding winning move

```
make-winning-move for player:
1. generate-pairs own-pieces of player ( $\Rightarrow$  (first second));
2. sum (integer of first, integer of second);
3. complement (sum);
4. generate unmoved-pieces:
5. test if integer of unmoved-piece = complement,
   if false continue generation;
6. move unmoved-piece for player,
   stop process.,
   stop and report no move.
```

the list of unmoved-pieces. On the first player's fourth move, for example, the three nested generators involve consideration of a maximum of $6 \times 3 \times 3 = 54$ cases, and if the branch is reached, yet another generator is called into operation.

Consider, on the other hand, how the forking moves could be discovered with the tic-tac-toe representation. We could associate with each piece the rows, columns, and diagonals (i.e., the lists of three pieces, or triads) to which it belongs.

FIGURE 3.8
program for generate-pairs

```

generate-pairs list: subprocess:
1.   generate list ( $\Rightarrow$  first-element):
2.       generate remainder-list of first-element ( $\Rightarrow$  second-element):
3.           construct (first-element second-element);
4.           apply subprocess to result,
               if stop-generate stop process and report stopped. . ,
               stop and report completed.

```

FIGURE 3.9
program for finding forking moves

```

make-forking-move for player:
1.   generate-pairs own-pieces of player ( $\Rightarrow$  (first second)):
2.       generate unmoved-pieces ( $\Rightarrow$  potential-fork):
3.           sum (integer of first, integer of potential-fork);
4.           complement (sum);
5.           generate unmoved-pieces:
6.               test if unmoved-piece = potential-fork,
                   if true stop;
7.               test if integer of unmoved-piece = complement,
                   if true stop generation.,
                   if completed stop;
8.           sum (integer of second, integer of potential-fork);
9.           complement (sum);
10.          generate unmoved-pieces:
11.              test if unmoved-piece = potential-fork,
                  if true continue generation;
12.              test if integer of unmoved-piece = complement,
                  if true stop generation.,
                  if completed stop;
13.          move potential-fork for player,
              stop process. . ,
              stop and report no move.

```

For each piece on the unmoved-piece list, we look for those rows, columns, and diagonals for which one of the remaining pieces is on the unmoved-piece list, the other on the player's own-pieces list. If there are two such triads, the piece gives a forking move. (In addition, triads already blocked could be marked, further cheapening the search.) This is clearly a much simpler process than the one previously described.

Now there is nothing that prevents us, in the number scrabble representation, from storing in memory the lists of triads associated with each move, and hence

making it possible to use essentially the same process for finding forking moves as we used in the tic-tac-toe representation. But the presence or absence of this information, in readily accessible form, is precisely the crux of the difference in representations. For the human being playing number scrabble, these lists are not in fact available, unless he explicitly goes to the trouble of writing them down or memorizing them. In tic-tac-toe, the lists are directly available in the visual diagram, which can be scanned rapidly along rows, columns, and diagonals.

In summary, let us return to a point made earlier about the requisite level of detail in describing information processing. As long as the processes being discussed are quite definite and elementary, it can often simply be assumed that there exists some encoding of the objects into symbol structures and some straightforward programs adapted to that encoding that will accomplish the processing and produce the indicated information as output (in an appropriate encoding). When the existence of such an encoding and such programs can be postulated, it is not necessary to specify the details of either encoding or programs.

We have given an explicit specification here for number scrabble to illustrate exactly what is involved in such a description of structure and processes. We will also provide such specifications at several other places throughout the book, where the specific encoding of objects or operations becomes important. But generally we will float above this level of detail. The fact that running computer programs exist for various tasks provides a continuing guarantee that the lower, more detailed levels of specification have been made explicit in many cases and could be in others.

The ability to abstract from detail partly reflects the simple hierarchical structure of information processing. However, it also reflects the power of lists and associations as general-purpose representations for encoding information. If, perchance, we had started with a linearly addressed memory, such as is found in a standard computer, we would have faced questions about how particular task environments could be encoded. The representation in terms of list structures and associations is particularly convenient in making it possible to add additional information to memory without destroying the information already encoded into the representation. It thereby greatly simplifies the representation task.

SOME CLASSES OF PROBLEM SPACES

So far we have proceeded almost entirely by example in characterizing representations of problem spaces and problems. It is time now to view matters more broadly and, abstracting from the detailed structure of either the IPS or its specific problem representations, characterize problem solving tasks in more general terms. Taking this broader view will give us a better picture of the whole area. It will also provide essential vantage points for the treatment of problem solving in the next chapter. Let us indicate briefly the nature of the connection.

In our illustrations of problem spaces we have already introduced important abstractions. Implicitly, the problem spaces have all been defined in relation to task goals, and the representations have been restricted to aspects of the total environment that were task-relevant. For example, we did not specify the physical

dimensions of the pieces used in number scrabble, or their color. Reader and authors automatically assumed that such properties were irrelevant to the goal.

However, since the relation of problem space to goal has thus far been left implicit, we have not had to state explicitly what a problem is, and what the interdependence is between task environment and goal. This interdependence is crucial—as we have already seen—for separating the task-relevant from the task-irrelevant components of the task environment, so that only the former need be incorporated in the problem space. The interdependence is crucial, also, in enabling the processes that the IPS has available—which must, after all, be provided to it in advance of its meeting a *specific* problem—to be brought to bear upon the tasks posed by these specific task environments. By introducing a general typology of problem spaces, we take the first step toward showing how general methods, associated with these types, can be used to solve specific problems. The demonstration will be continued in Chapter 4.

What Is a Problem?

A person is confronted with a *problem* when he wants something and does not know immediately what series of actions he can perform to get it. The desired object may be very tangible (an apple to eat) or abstract (an elegant proof for a theorem). It may be specific (that particular apple over there) or quite general (something to appease hunger). It may be a physical object (an apple) or a set of symbols (the proof of a theorem). The actions involved in obtaining desired objects include physical actions (walking, reaching, writing), perceptual activities (looking, listening), and purely mental activities (judging the similarity of two symbols, remembering a scene, and so on).

In this book, the objects we shall mainly be concerned with are systems of symbols—for example, proofs of theorems or English sentences. The actions we shall examine are mainly manipulations of symbol structures. Restricting the discussion to symbolic entities and processes does not severely limit our analysis of problem solving, except at its physiological boundaries (e.g., the physiological aspects of sensory and motor skills, especially those requiring real-time action and coordination). For the crucial activities, at least in human problem solving of any complexity, are symbol-manipulating activities that take place centrally. This is true even when the desired object and the required activity are physical—e.g., going to the orchard to pick an apple.

Consider the problem of finding a proof for a theorem—say, a theorem in elementary symbolic logic. The objects in this problem, theorems and other logic expressions, are composed of symbols. We represent them internally as list structures, like those we introduced in the last chapter. A proof is a sequence of axioms and theorems, terminating in the theorem to be proved. Hence, a proof may be represented as a list of list structures—which is itself a list structure. The task of finding a proof for a theorem is, then, a task of finding a list structure with appropriate characteristics.

Instead of defining directly what it means most generally for a human to have a problem (or for any organism or device to have one), let us try the following

strategy. To have a problem implies (at least) that certain information is given to the problem solver: information about what is desired, under what conditions, by means of what tools and operations, starting with what initial information, and with access to what resources. The problem solver has an interpretation of this information—exactly that interpretation which lets us label some part of it as *goal*, another part as *side conditions*, and so on. Consequently, if we provide a representation for this information (in symbol structures), and assume that the interpretation of these structures is implicit in the program of the problem solving IPS, then we have defined a problem. Thus, we say with any textbook on Euclidean plane geometry:

Given: A general triangle, ABC.
 Prove: The three lines bisecting its angles
 intersect in a common point.

This is a problem, provided that the problem solver has (implicit) not only information defining triangle, bisecting, and so on, but also appropriate interpretations of Given and Prove. In the appropriate context (e.g., a homework assignment) the problem solver will come to “have” the problem stated above—and will be found diligently at work on it some time before the next class period (assuming he is appropriately motivated).

We need not make special provision for a separate representation for each particular problem. Instead, we can define representations that cover large classes of problems. Likewise, we need not assume that the problem is presented by means of English language statements. This would imply capabilities of understanding natural language that are beside the main issue. Instead, we can define our general representations directly in terms of the essential symbol structures (deep structures). We will introduce two such general representations: the *set representation* and the *search representation*. We have no guarantee that all problems can be represented in one of these two forms. It is always possible that some important classes of problems have not yet been represented.

Set Representation of a Problem

A problem proposed to an information processing system is well defined if a test exists, performable by the system, that will determine whether an object proposed as a solution is in fact a solution.⁵ By *performable* we mean, more specifically, performable with a relatively small amount of processing effort.

Consider the problem of constructing a symbolic expression having the characteristics p_1, p_2, \dots, p_n . If the system has a set of relatively simple tests for determining whether any expression does or does not have each of the characteristics, p_i , this is a well-defined problem.

In most cases it is possible to define a set of objects—the set of possible solutions—that contains the solution or solutions. Where the solution is a symbolic expression, we can take as this set the set of “all possible” symbolic expressions.

⁵ The notion of a *well-defined problem* was proposed by McCarthy (1956).

When we need to be more precise, we may define it as the set generated by a certain enumerative procedure from an alphabet of elementary expressions. Hence, in set-theoretical terms, a problem may be characterized as follows:

Given a set U , to find a member of a subset of U having specified properties (called the goal-set, G).

We will call this the *set representation* of a problem, although either of the longer terms *set-theoretic representation* or *set-predicate representation* would have been equally appropriate. We have already commented on the phrase *having specified properties*. In the sections that follow we shall have more to say about what is meant by *given* and *find*.

Three reasons (not the only ones) why problems are problems are that: (1) the original set, U , of possible solutions given to the problem solver can be very large (it is often immense⁶), (2) the actual solutions, G , can be dispersed very widely and rarely throughout it, and (3) the cost of obtaining each new element and testing it can be very high (e.g., the search for the Northwest Passage). Thus the problem solver is not really given the set of possible solutions; instead he is given some process for generating elements of that set (all or some of them) in some order. Such a generator has properties of its own, not usually specified in stating the problem—e.g., there is associated with it a certain cost per element produced, it may or may not be possible to change the order in which it produces the elements, and so on. The test that a given element has the desired properties also has costs and times associated with it. The problem can be solved if all of these costs are not too large relative to the time and computing power available for solution.

In some cases, a problem that is characterized set-theoretically can be decomposed into a number of subproblems, each of which can be characterized in the same way. Consider how we might represent the problem of solving a crossword puzzle. Take as U all possible arrays of letters of the English alphabet that will fill the white squares of the puzzle. The subset G comprises those arrays in which all consecutive linear horizontal and vertical sequences of letters form words that satisfy the specified definitions.

Decomposed into subproblems, the crossword puzzle may be described thus: There are n problems, $i = 1, \dots, n$. Take as U_i the set of all English words. The subset G_i comprises those words that (1) satisfy definition d_i , (2) are of length k_i , and (3) at each intersection with a word belonging to one of the other subproblems (say, at square s_{ij}) have the same English letter as that other word. The last-mentioned condition on the subset G_i , of course, introduces an interdependency among the solution sets for the different subproblems.

The respective responsibilities of generators and tests in obtaining and verifying solutions could be divided in different ways. For example, it might be possible to devise a generator that produced only words satisfying conditions (1) and (2),

⁶ The term *immense* was proposed by Elsasser (1958) to stand for numbers so large that their logarithms were themselves very large [e.g., N is immense if $\log(N) = 10^{40}$]. The numbers we deal with (e.g., 10^{120}) are not so large, but do satisfy the same functional criterion: that no revision of the number from more detailed investigations and assumptions can affect the argument at hand.

and then simply to test for satisfaction of (3). As expert crossword-puzzle solvers know, this is not the only—nor usually the best—procedure. An alternative is to find a generator that produces only words that satisfy (1) and (3), then test for satisfaction of (2).

In continuing our discussion of factorization it is time to introduce another class of problem representations where the role of factorization into subproblems is even more prominent. We turn now to the search representation.

Search Representation of a Problem

The set representation of a problem, while it is relatively general, is not always the most convenient characterization. As a first step toward constructing an alternative, let us describe in detail, in the set-theoretic framework, the problem of finding a proof for a theorem in Euclidean geometry or elementary symbolic logic. As in the crossword puzzle example, we will exploit the factorability of the problem into subproblems.

Discovering the Proof of a Theorem. A proof of a theorem, t , is a symbolic expression that consists of a sequence of subexpressions (the proof steps). Hence, the set U is the set of such sequences. A sequence, u , belongs to G , the set of proofs of t , if it has properties p_1 and p_2 , where p_1 means that each subexpression in u either is an axiom or is derivable by certain allowable transformations (the rules of inference) from previous subexpressions in u , and p_2 means that the final subexpression in u is t .

Consider the tests that determine whether p_1 and p_2 are satisfied. The tests can be factored into parts, each of which is concerned with characteristics of only a portion of u . (1) The test for p_2 concerns only the characteristics of the final subexpression. (2) Let u_i be the object comprised of the first i subexpressions of the sequence u . Then the test for p_1 can be factored into tests p_{11} , p_{12} , and so on, where p_{1i} is a test on the final subexpression, $last(u_i)$, in the sequence u_i .

Moreover, these latter tests are relational—they are satisfied if $last(u_i)$ stands in a certain relation to one or two prior subexpressions in u_i , the subexpressions from which it is directly derived. We can regard the sequence u_i as having been generated from the sequence u_{i-1} by the application of an operator, Q_i , to the latter sequence. We can then write:

$$u_t = Q_t(Q_{t-1}(\dots Q_1(u_0)) \dots) \quad (1)$$

where $u_i = Q_i(u_{i-1})$ means the expression (sequence) obtained from u_{i-1} by applying to it the operator Q_i , and where u_0 is the sequence of axioms and theorems given initially.

Thus, we can designate the theorem, t , in several ways. It is the final subexpression in the sequence u_t . From the right-hand side of equation (1) we see that it is also the subexpression $last(u_t)$, generated by applying the sequence of operators, Q_i , to u_0 . In order to talk about these different ways of referring to objects, it is convenient to introduce the terms *state language* and *process language*. Symbols

that designate expressions, elements of expressions, characteristics of expressions, or differences between expressions belong to the state language for the problem. Symbols that designate operators for transforming expressions, sequences of operators, or characteristics of operators belong to the process language.

A proof is most simply characterized by using both state and process language. Thus, u_i is a proof of t if $\text{last}(u_i) = t$, and if it can be represented in the form of equation (1), where all the Q_i are *admissible* operators—i.e., operators corresponding to the rules of inference—and where the subexpressions of u_0 are axioms and previously proved theorems.

State Space and Action Space. This dual aspect of problem situations is not peculiar to theorem proving. Problem solutions in most domains are defined by (1) characteristics of a terminal state, (2) an initial state, (3) conditions on the admissible transformations from one state into another, and sometimes (4) characteristics of the intermediate states. The state-process distinction is fundamental to understanding the behavior of an organism when it seeks a goal object. The internal representation of the goal object symbolizes it in the state language. What is required is a series of signals (or symbols) in the process language that will transmit the proper instructions through the motor system to the effectors in order to enable the organism to attain the goal (i.e., to change the state to the desired one). Hence, the state language is related to the operation of the *afferent* system, the process language to the operation of the *efferent* system. The problem is solved by an appropriate translation of afferent stimuli into efferent responses; and tests of the solution are applied to new stimuli received through the afferent channels.

In everyday language, the term *solution* is used in several different ways. Sometimes (e.g., in theorem proving) we call the sequence of expressions, u_i , the solution. Sometimes (e.g., in the organism's attempt to reach a goal object), we call the sequence of operators, Q_1, \dots, Q_i , the solution. Sometimes (e.g., in a crossword puzzle) we call the terminal state, $\text{last}(u_i)$, the solution. Where it is necessary to distinguish them, we may call these the *solution-path*, *solution-action-sequence*, and *solution-object* (or *goal-object*), respectively. When we use the term solution without qualification, it will be clear from the context which one of these is intended.

In the course of solving a problem, more than one potential solution path is sometimes generated. In working forward, for example, more than one admissible operator may be applicable to u_0 . Applying each, we obtain paths u_{01} , u_{02} , and so on. Again, in general, more than one admissible operator may be applicable to each of these, giving u_{011} , u_{012} , \dots , u_{021} , u_{022} , and so on. In this way a branching tree of paths (*search tree* or *discovery tree*) is obtained, and the terminal branches are compared with the conditions on $\text{last}(u_i)$ until one is found that satisfies these conditions. The path that leads to this branch is a solution path.

Alternative Spaces for Search. In our description of the search representation, the nodes of the search space have been realizable states of affairs, and the links between nodes have been the actions that changed one state into another. But in problem solving the actual search for a solution need not, and usually does not, take place in the external environment. Accordingly, the states represented by nodes of the search space need not correspond with realizable states of the outside world, but can be imaginable states—literally so, since they are internal to the problem

solver. These states may be generated, in turn, by operators that do not satisfy all the conditions of admissibility. For example, operators may be employed in the problem space that generate plausible sequences of states, and those sequences that do not satisfy the admissibility tests may be modified or rejected at later stages of the problem solving process.

Putting the matter in different terms, the tree of solution action sequences is only one of a number of ~~trees that may be associated with transformations on a problem space.~~ The structure of the search space that is used is not a property solely of the task environment—its states and admissible actions—but depends also on the problem space and program the problem solver employs. He may, for example, work backward from the desired state, applying operators that are, roughly, the inverses of the admissible actions, in order to see whether he can discover a path that leads back to the initial state.

Operators that are not admissible in the external environment may take the form of conjectures of intermediate states of affairs which, if they could be attained, would be likely steps toward the final goal. Thus, a chess player might conceive through search a specific mating configuration, then initiate a new search to find a sequence of moves to attain this particular checkmate.

Moreover, the solution to the problem of a chess player making a move is not to discover a whole sequence of future moves, but simply to discover a good *next* move. The search tree of subsequent moves that he explores is not a part of a solution action sequence, but is simply a structure he creates to predict future consequences, and to evaluate, thereby, the move directly before him.

Still another possibility—search in a planning space—is illustrated by the crossword puzzle example. At each step of solving a crossword puzzle, a skillful problem solver may first pose the subproblem of finding a good next problem to attack. For example, he may look for a particular word, j , whose initial letter or letters are already known, using this information to simplify the task of the generator seeking to produce a word satisfying the definition, d_j . Thus, at each stage in constructing the solution object, a subsidiary search is evolved to determine which generator is to be activated.

In spite of these qualifications and complications, it is true that in many classes of problems the solution is defined by (1) properties of a solution object, conjoined with (2) the requirement that the solution object be reached by a path that can be generated by application of a sequence of operators drawn from a set of admissible operators (legal moves). In these cases, an elementary problem solving program might generate possible solution paths, using the admissible operators to produce sequences of one kind or another. But we must keep in mind that the problem solver may also make searches in which the states and operators of the problem space do not correspond with the states and admissible operators of the external environment.

Further Examples of Problems

To be a bit more concrete about both set representation and search representation, let us characterize some other common problems in the terms we have been using:

Finding the Combination of a Safe. Take as U all possible settings of the dials of the safe, and as G those particular settings that open the safe. As safes are usually constructed, G consists of a single element. If the safe has several dials, a particular search tree might be produced by trying, for each setting of the first dial, all possible settings of the others, and so on. As in the crossword puzzle example, different problem solving techniques might define vastly different search spaces.

Designing a Machine. Take as U the set of all possible parameter values for a machine design; take as G the subset of parameter values that: (1) satisfy the design specifications, and (2) meet certain criteria of cost minimization. For cases of practical interest, the set U will be immense and hence will have to be explored in somewhat systematic fashion. In early stages of the search, for example, particular design variables may be bounded, or even fixed, prior to establishing limits on the other variables. If the priorities are fairly definite, then the search tree will have hierarchic properties—the branchings at different stages referring to different classes of design variables.

Programming a Computer to Invert a Matrix. Take as U the set of all possible sequences of computer instructions, and as G a particular sequence that will perform the specified matrix inversion. The branching tree of sequences in U also defines a search tree.

Translating a German Article into English. Take as U the set of all possible sequences of English words; take as G the subset of sequences that: (1) satisfy certain criteria of English syntax and style, and (2) have the same meaning as the German original. Again, the interpretation in terms of the search representation can be carried a step further by identifying the elements of the sequences mentioned with the successive segments of the maze that constitutes a path. However, in all these examples, the formulation in terms of the sets, U and G , is more general, in the sense that there are often numerous alternative ways of representing the problem, so stated, by a maze.

ANALYSIS OF THE TASK ENVIRONMENT

What does it mean to analyze a task environment? The topics so far discussed in this chapter and the presentation of an IPS in the prior chapter show what it means to present a theory of a specific segment of human behavior. A representation of the problem space is given, comprised of a representation of the environment and one of the problem. (If needed, a representation of the actual environment, independent of the subject's perception of it in his problem space, may also be given—say, if the situation involves feedback from the actual environment during the process of solution.) Then the IPS is specified. It incorporates the problem space, not in the sense of spanning its whole extent, but in possessing symbol structures and programs that provide access to that space via the system's processes for generation, testing, and so on. In addition, the IPS incorporates the goal or task, either explicitly in a symbol structure (with an appropriate interpreter) or implicitly in the

behavior of some of its programs, as we saw earlier in the chapter. The IPS, when set in the environment and given the goal, becomes a determinate system that produces a stream of behavior. This is the theoretical behavior to be compared with the human's actual behavior.

All the above is familiar in outline, and we understand generally in what sense such a theory would explain the empirical events (or would fail to, if the IPS or task environment were incorrectly posited). But there is an alternative route to explaining the behavior. Remembering that a problem solver is an adaptive system, we might postulate that the human problem solver produced the behavior that he did because he had to—because the behavior was demanded by the task environment. This is also an explanation of the behavior. There is no necessary conflict between the two explanations, assuming that the IPS did in fact solve the problem. Then the various symbol manipulations that occurred under the control of the program constituted an effective means for responding to the demands of the task environment, with the production of appropriate behavior.

It is a premise of the second kind of explanation—that the behavior corresponds to the demands of the situation—not only that the goal was held by the problem solver and that he was adequately motivated, but also that he was able to attain the goal. It might seem, then, that this form of explanation would make few predictions in cases where the problem solver was *unable* to solve the problem—unable to meet the demands of the situation. But this is not entirely so, for one may explain (or predict) behavior by observing what are the “obvious” things to do to attain the goal (even though they may be insufficient), and therefore what are the things that the problem solver, being a bear of little brain, *will* do.

It is a cautionary note to (rather than a premise of) the classical kind of process explanation that the program and symbol structures that are inferred to produce the problem solving behavior may themselves be idiosyncratic to the task and ephemeral, being simply what is demanded by the task environment. Thus, quite different structures and programs would be generated by different tasks. In that case, the process explanation would show only that indeed the problem solver could adapt sufficiently to the situation.

We will employ both kinds of explanation in this book. Indeed, as we observed earlier, one of our main tasks will be to understand what is demanded by the task environment, so that we can understand—by elimination—what aspects of behavior are determined by the psychology of the problem solver. The purpose of this section is to be sure we understand what it means to analyze a task environment in order to reveal its demands.

It is easy to give an essentially correct definition of a demand of the task environment: it is a constraint on the behavior of the problem solver that must be satisfied in order that the goal be attained. Thus, the environment per se does not make demands: rather the problem or goal makes them via the problem solver's commitment to attain it. (The features of the environment that give rise to these demands constitute the relevant structure or texture of the environment) (Tolman and Brunswick, 1935).

Difficulties arise with this definition from the question of the independence of the demands and task structures from the nature of the problem solver. The whole point of viewing the environment as making demands is to provide a frame-

work outside the problem solver to which his behavior can then be related. If this can be done, it seems natural to make an analysis of the task environment before looking at behavior. We have organized each of the main empirical parts of this book in exactly this fashion. Each main task starts with a chapter on task analysis, followed by chapters on behavior.

But if something's being a demand is only apparently independent of the problem solver, and *in fact* depends on intimate features of his program, then the factorization into task environment and problem solver is a mirage. The possibility of such interaction arises from two sources: first, that the problem solver has a goal; second, and most important, that a bond of necessity underlies the constraints upon modes of goal attainment.

With respect to the first, the goal, we have eliminated the difficulty by fiat—by conditioning the concept of demand both on the goal's being held and on the problem solver's being adequately motivated. Both of these side conditions clearly do depend on the nature of the problem solver, but in the experiments reported throughout the book they are always satisfied. Furthermore, it is often easy to know that they are satisfied, for in our culture (at least) people are willing to declare publicly (either verbally or behaviorally) many of their goals.

Thus the remaining difficulty is to clarify what it means for behavior to be necessary for a goal. This will require some discussion. We start with an extreme case where the issues are clear, and then move to the complexities.

Task Invariants

We are given three things: ¹ an environment, a ² problem solver, and a ³ goal or task. Suppose it is possible to demonstrate that *all* paths in the environment that lead to attainment of the goal have a certain property. For instance, any way of winning tic-tac-toe involves having three-in-a-row. Or, all proofs of a given theorem from a set of axioms involve the use of a specific axiom. Clearly the behaviors corresponding to these features—making three-in-a-row or using the specified axiom—are demands of the task environment. These elements cannot depend on the nature of the problem solver, because the demands are invariants of the situation. Their invariance is quite independent of the amount of analysis that is required to reveal them. In the case of tic-tac-toe the invariance is obvious; in the case of the theorem it may require a difficult proof to establish it. For example, the classic problems of squaring a circle, duplicating a cube, and trisecting an angle are all impossible in a given task environment (when the only operators admitted are constructions with straight-edge and compass), but proof of their impossibility took many centuries.

These examples raise the issue of whether the set of admissible operators—and consequently the tree of reachable states in the problem space—can always be regarded as a property of the task environment, independent of the properties of the problem solving IPS. Consider the task of crossing a deep chasm with only a single bridge across it. Going over the bridge would seem to be a demand that the environment places on any problem solver. But suppose a problem solver arrives who is provided with a chartered airplane, or who is equipped with wings, is capable

of teleportation, or whatever. For such a problem solver, the environment no longer demands going over the bridge. In the original statement of the problem, we have smuggled in, it would seem, some assumptions about the nature of the problem solvers—that they lack wings, chartered planes, or the capacity for teleportation.

In the case of a mathematical proof this difficulty does not arise, because the set of admissible operators is defined formally, and the definition is accepted by both experimenter and subject. In problems involving actions upon the real world, possibility and the admissibility of operators are empirical, not definitional matters. Pragmatically, however, we can usually take care of the difficulty by drawing the boundary between IPS and environment a little closer to the center of the IPS, so that capabilities for actually altering the external environment are treated as properties of that environment and not of the problem solver. Then it becomes part of the specification of the task environment whether the problem solver has an airplane, or whether he can acquire one by an action sequence that is open to him.

We must exercise caution, however, in shifting the boundary between problem solver and environment. If we remove particular operators and classify them with the task environment, there is a danger that the problem solver will disappear entirely, and that there will be no room at all for a theory of him. For example, how shall we treat the problem solver's capabilities (and inabilities) for doing arithmetic? Is it a description of the problem solver that he can do mental multiplications at a certain speed in solving the problem? Or is this a specification of the environment (as we might want to regard it if there were a question of the availability of paper and pencil or desk calculators)? And how shall we treat the problem solver's capacity for attempting goals? If we follow the path of assigning all means to the environment, there will be nothing left of the problem solver: he will do what he does because all that he is—being means—is specified by the environment.

These examples suggest that a suitable way to fix the boundary is to regard possibilities of actual physical actions as part of the description of the environment, but to regard the information processing activities of the problem solver—the processes for searching through his internal problem space—as describing him. We must now consider more carefully whether such a strategy provides at least a usable and pragmatically tenable boundary between IPS and environment.

Adaptivity and Task Demands

The success that an information processing system will experience with problems of a certain kind can be used to construct a measure of its adaptivity or intelligence in environments resembling those in which the test problems were set. Such a measure will be highly predictive of performance in a very similar environment, less predictive of performance in environments with different characteristics.

To the extent that such a measure is predictive at all over diverse environments, we might want to call it a measure of general intelligence, but such a measure is not needed for our purposes. We can be content with measures that are specific

to certain classes of environments; and that consequently permit us to predict performance in tasks in the appropriate environment, to scale such tasks according to difficulty, and to scale IPS's according to adaptivity, ability, or intelligence in performing these tasks. All of this corresponds to the common practice in psychometrics.

* Now the demands that a particular environment places on an IPS may depend on intelligence. In our example of the chasm to be crossed, going over the bridge may be a demand on an IPS of ordinary intelligence. A very much more intelligent IPS might solve the problem by inventing the airplane (or, less dramatically, by thinking of the still unusual possibility of chartering one). For him, there is no demand that he cross the bridge. If, as we proposed earlier, we include *all* external action possibilities as part of the description of the environment, then crossing the bridge is not a demand on either problem solver. If we include only thinkable external actions in the description of the environment, then the environments are not the same for the two problem solvers.

A pragmatic solution to the dilemma, which preserves for all practical purposes the concept of demands of the environment, is to assume that all the problem solvers whose behavior is to be studied are drawn from a reasonably narrow range on the scale of intelligence. If two populations of widely differing intelligence are to be studied, then the environment (for the same task) must be described differently for them. In the description of the environment for any given, relatively homogeneous, group of problem solvers, only external actions that are thinkable by them need be included.

* In the search representation of a problem the task environment can be identified with the whole space of search paths that are accessible to IPS's of a specified level of intelligence. To the extent that intelligence in the problem domain is scalable along a single dimension, the search space describing the task environment of a highly intelligent problem solver can be assumed to include, as a subpart, the smaller and sparser space describing the environment of a less intelligent problem solver. (Thus, the space for the airplane inventor of the previous example would contain branches leading to the invention of that alternative; while these branches would be missing from the search spaces of noninventive problem solvers.)

Once a search space of this kind—a network of possible wanderings, so to speak—has been specified, the demands of the environment simply become properties of the space that are invariant over all paths that lead to a problem solution. It is easy to see that an analogous notion of demands of the environment can be introduced into the set-theoretical representation of a problem. For present purposes, we can restrict our discussion to the search representation.

It is worth noting that the ability measures that are relevant here reflect the problem solver's knowledge of the problem domain in addition to whatever problem solving aptitudes he may possess. Consider the game of chess, for example. Although there are no hard data to prove it, there probably is some correlation between skill in chess—for those who play the game at all—and standard measures of intelligence. There is undoubtedly a much *higher* correlation between skill in chess and chess knowledge (which, in turn, is correlated to some degree with amount of experience). Chess skill is sufficiently unidimensional so that official chess organizations assign to players scalar numerical rankings. In the American

system, for example, a class A player has a rating between 1800 and 2000, an expert between 2000 and 2200, a master between 2200 and 2400, and a senior master over 2400. In a game between two players whose ratings differ by, say, 300 points, the outcome is predictable to a high degree: the stronger player would probably not lose more than one game in twenty or more, and not draw more than one in five.

Players of about the same strength see roughly the same things in a given game situation. Hence the task environment can be described in about the same way for them, and the demands of the situation can be inferred. Indeed, the idea that a move is demanded by the situation is a standard part of the chess culture. At master and grandmaster levels, it is also taken for granted that if enough time is allowed for the analysis of a position (in difficult cases this may mean hours or days of analysis), the move that is demanded by the situation can be detected more or less unequivocally. (We will have more to say about these matters as they apply specifically to chess in later chapters.)

We can now summarize our account of what comprises an analysis of the task environment. An analysis of the task environment produces a description of the constraints on behavior that must be satisfied to attain the problem goal at a specified level of intelligence or adaptivity. For a given level of intelligence, certain paths in the task environment are not available as paths to the goal—they are too difficult. Removing these paths, one can then search for the invariant features of the remaining paths that do lead to the goal.

Simplification of the Problem Space

Our approach to task description provides us with at least one valuable by-product—a technique for vastly simplifying the description of search paths through the problem space. We consider, again, a task environment described as a search space, relative to an IPS of a given level of intelligence. As we shall see in the next chapter, we can measure the difficulty of search through such a space, for an IPS of given intelligence, by the expected size of his search tree. We can measure in this same way not only the difficulty for the IPS of an entire search path, but also the difficulty of any segment.

An IPS, let us suppose, solves a problem by finding a path from A to G, via B, C, D, E, and F. We can speak of the difficulty of getting from A to G, but also the difficulty of getting from C to D. Perhaps getting from C to D was easy, but getting from D to E was very hard. With any particular node, say C, in such a search space, we can associate all the nearby nodes whose attainment is *obvious*, once they have reached C, to problem solvers of the specified level of intelligence. How they traverse these obvious subsegments is often of little interest in explaining their overall problem solving behavior. Hence, in describing a task domain, we need not include the detail of the links connecting a set of nodes that are all obviously reachable from each other. Instead, each such set can be treated as a single node and detail retained only for the interesting and difficult passageways that separate one such cluster from another.

In terms of a different metaphor, in describing the terrain of the problem environment, the detailed topography of flat or slightly rolling plains will have

little import. What will be interesting and significant is the location and detailed configuration of the mountain passes. In our later discussions, we will be assuming, for instance, that our (adult) subjects can simply follow rules at the level of executing chess moves or carrying out simple arithmetic processes. We will not consider how these component acts are done—they amount to moving in a plain. If our subjects had been young children, such assumptions would not have been viable. Making a legal move in chess can be a problem for a child or for any person just learning the game; but for our subjects it is not problematic but obvious. We incorporate such assumptions of obviousness by declaring the legal moves to be processes that are immediately available to the IPS without calling on its problem solving apparatus.

Thus, we will ~~eliminate the obvious~~ in describing our subjects' behavior and focus on the difficult and uncertain aspects of their behavior—their progress over the high passes and ridges. We will often describe the behavior as a sequence of *episodes*, each of which is a succinctly describable segment of behavior associated with attaining a goal. If the goal is sufficiently easy to attain (or to identify as not attainable), then we need not be concerned with the detailed behavior of the problem solver in getting to it. We can treat each episode as a unit, paying attention primarily to what determines which episodes are initiated and how they are terminated.

Specificity of Environments and Adaptability

This approach to defining the demands of the task environment, while committing us to specifying the intelligence of the problem solver, does not commit us to the assumption that intelligence is generalizable over different task domains. Its generality remains an empirical question, to be settled by evidence, not definition. We are thus free to restrict and normalize the domains that we wish to analyze in any given instance.

Perhaps the most important restriction that we can impose is to choose a series of problems for a definite class of task environments. Then we only assume a unidimensional ability for, say chess or cryptarithmic or elementary logic. In contrast with the task domains covered by intelligence test batteries, these are extremely homogeneous. Indeed, all three domains put together are still a relatively homogeneous area. Yet, we will not need even to postulate that the ordering of problem solvers in chess ability agrees with that in, say, logic. We also use a restricted class of subjects—subjects who are intelligent enough to understand and accept the tasks, yet not so smart or knowledgeable that the tasks are trivial for them.

A second important restriction we can accept is to adapt our analysis to the subject's internal representation of the task. Much of the variation—especially the extreme variation—of response to a task by different problem solvers comes about from differences among the representations they use. Our analysis of a task will assume that the problem solvers are using a problem space of a specific character. If two subjects use radically different problem spaces, we would expect to make two separate analyses of the task, and perhaps then arrive at correspond-

ingly different measures of difficulty and different invariant features of the task environment. Our theory of problem solving will in fact give us a good deal of basis for understanding the nature of intelligence and its relation to the structure of the problem space.

A third restriction, already noted, is to treat some of the informational resources as aspects of the task environment instead of aspects of the problem solvers. For example, in analyzing the cryptarithmic task, we may assume that all subjects know that a number of the form $B = A + A$ is an even number; hence, we may regard this fact as a feature of the task environment rather than a property of the subject. This again reduces the variance among the problem solvers. In the tasks we shall examine, our most important technique for reducing intersubject variance in representation of the task environment is to select subjects who are naive in certain dimensions—e.g., who have not previously seen the particular problems we give them, or the specific kind of task we ask them to perform. (In the chapters on chess we clearly relax this latter requirement of naiveté, though not the former one. The subjects must be chess players, but the positions given them must be new to them.)

With these restrictions, as well as the fact that we only need approximate validity, the assumption of scalable intelligence becomes tenable. The final demonstration of this, of course, comes from the analyses that we undertake in later chapters of the book. All we have done here is to establish the methodological basis for those analyses.

SUMMARY

This chapter has focused on some major issues that arise in defining a task environment for a human problem solver, viewed as an IPS. The first question is 1. how to represent the external environment. We, as experimenters, have no privileged access to the real world that constitutes the external environment for another human. Even when we create a laboratory situation, we still must describe the aspects of that environment that are relevant for our subjects' behavior. We saw, in effect, that our choice of representation amounts to a set of hypotheses about what encoding will be provided by the subject IPS.

The second question is 2. how the subject will represent the environment internally. Here we introduced the key concept of the problem space, which not only represents the current situation, but also the possibilities for change and transformation of that situation. We noted especially that one must not assume automatically that the subject's problem space will lie inside a legal problem space as understood by the experimenter.

We made these two issues concrete by detailing an explicit representation in an IPS for both the external environment and the problem space of tic-tac-toe.

The third question is 3. how to represent problems and their goals. We provided representations for two large classes of problems: the set-predicate representation and the search representation. In each representation, the problem solver is viewed as having certain information, and the definition of the desired solution may be

viewed as a request for specified additional information. Although these two representations—the set-predicate and the search—cover a vast territory, we left open the possibility that there are other problem representations that cannot usefully be subsumed under either of these paradigms.

The final question (returning to the one that opened the chapter) concerns the relation of task environment to problem solver, and the extent to which either can be analyzed separately. We posed the issue by observing that the behavior of an adaptive organism can often be explained as behavior demanded by the task environment, rather than behavior that flows from the operation of mechanisms internal to the organism (i.e., from the program of the IPS). At the core of this environment-centric view is the notion of invariants of the environment—aspects of behavior that are constant along all paths that lead to a goal, hence are necessarily exhibited by all problem solvers that are successful. We found that this environment-centric mode of analysis could be extended considerably if a measure of adaptivity or intelligence could be assumed to exist, even approximately. We showed how such an assumption was in fact implicit in the concept of problem space.

We also introduced the concept of episode, defined as a segment of behavior devoted to attainment of a goal. If the goal is easy enough (relative to the power of the problem solver), then the episodes can be derived from analysis of the task environment and treated as units.

In general, this chapter has had to be content with raising issues, introducing the concepts appropriate to understanding the issues (e.g., problem space, task invariants), and illustrating these concepts by brief examples. For full understanding, task environments must be dealt with in their particularity, and to make any of these general concepts fully operational requires an analysis of an environment in detail. We used simple examples and metaphors, precisely because illustrating the concepts with a real task environment would obscure the broad issues in the technical particulars of that environment. In three later parts of the book, where we deal respectively with cryptarithmic, elementary logic, and chess, the concepts and issues introduced here will provide the framework for analysis.

However, there is still one more preliminary chapter. Chapters 2 and 3 have set the stage for a general description of the problem solving activity itself. We have described the two principal components—the information processing system in Chapter 2, the task environment in the present chapter. The goal, accepted by the problem solver, and to be attained in the given task environment, provides the interface between the two components. In the next chapter, Chapter 4, we consider in general terms how an IPS goes about solving a problem, then illustrate the general discussion by an extensive analysis of the behavior of one of the first problem solving systems that was programmed for a digital computer—the Logic Theorist.