Chapter 1. The Game

1.1. Systematic Problem Solving

Let's begin with an attempt to define our subject. **Science** (Latin *scientia* - knowledge) seeks to understand the what, how and why of things. Answers are sought to questions, solutions to problems.[1] We are all familiar with problems - life is full of difficulties, setbacks, failures and conflicts. But we are thinking of those situations which can be resolved, that is, problems that have solutions. A **problem** may be thought of as *an unresolved situation*. We will be interested only in resolvable problems, not because unsolvable problems don't exist, but because solvable problems are more in harmony with the image of absolute truth popularly associated with science. Hopefully in introductory classes, teachers and textbook writers are careful to present problems which can be resolved unambiguously (according to the rules and conventions of the game of science). Nevertheless, humans are not perfect and it would be naive to assume that every published statement and every printed answer is without flaws.[2]

Problems can be presented in the form of questions with implied answers. The **solution** to the problem may be thought of as the *path* that connects the question to the answer. The term solution is sometimes used for the answer itself. This ambiguous usage may be a little more acceptable if one recognizes that if the path to the answer is known, the answer can always be found. Fig. 1.1 illustrates the relationship between the problem, its resolution and the path leading to resolution. Note that there does not necessarily have to be an unique solution path to a given problem, only that there has to be an unique answer. The fact that an unique answer can be reproduced is fundamental to rational science. That some problems may exist for which no answer can be obtained is also possible, and much more difficult to deal with.

---

[1] Terms introduced for the first time are **emboldened** and are defined in the Glossary for reference.

[2] Recently a physics student gained notoriety for finding a calculational error which had lain undiscovered for 300 years in the venerable Isaac Newton's great classic *Principia Mathematica*.

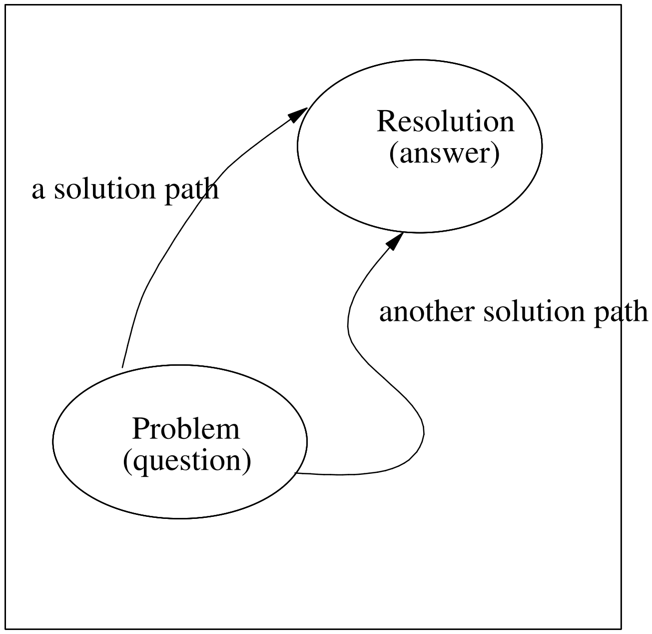

Fig. 1.1 Solution Space

With the *what* out of the way, let us turn to the *how* (and leave the *why* to another discussion). How do we solve problems? There may be as many different answers as there are problems to solve and people to solve them. One does not get out of debt necessarily the same way one plans a trip to Europe (although there may be a connection between the two!). It is our contention, however, that many problems can be grouped into classes with common methods for solution. These include performing arithmetic calculations (addition, subtraction, multiplication and division), filling out income tax forms, and most of the chemistry exercises students encounter in college courses. Our specific purpose here is to help you develop and use methods for solving chemistry problems, although the ideas apply to a much broader spectrum of experience.

Chemistry deals with the study of material objects, or matter. Many questions regarding material properties and behavior can be answered systematically by following logical processes. This does not mean that the guesswork is totally removed from chemical investigation, but rather that it is shifted from deducing a statement regarding a *particular* situation to understanding the *processes* for obtaining *general* information, and then applying those processes to solving particular problems.

Do systematic procedures for solving problems exist? Some time ago a woman was taking beginning chemistry. Her husband was not taking the course, but would take her exams after she brought copies home. He never read the textbook, he never went to lecture, he never took a chemistry course, didn't know the definitions or terminology, yet would have received a B letter grade in the class based on his performance on the exams. When asked how he was able to do so well, he replied that he was a philosophy major, and applied only logic and reasoning to the questions on the exams. He must have been acquainted with some basic mathematics, but beyond that he must have had some skill in decoding information and general problem solving as well (possibly coupled with an understanding of the psychology of exam taking).

An **heuristic** (Gr. *heuriskein*, to discover) is a *strategy* for accomplishing some purpose, such as solving a problem. There is a hope but no guarantee that a given heuristic will lead to an acceptable solution because heuristics characteristically have no formal proofs that they accomplish their intended purpose. They are akin to hypotheses. Examples of heuristics are battle plans (including logistics and reconnaissance), business practices (including risk estimation and marketing), game strategies (including opening moves and bluffing), and scientific experiments (including design and analysis). An **algorithm**, on the other hand, is a *guaranteed procedure* for accomplishing some purpose, such as solving a problem. The term algorithm is derived from the name of the Arabian mathematician Mohammed A-Khowârizmî (Latin, Algorismus), one of the major contributors to the rules (procedures) of arithmetic in the West. (The term *algebra* is also derived from the same name.) While there is no single simple prescription for solving every problem one is ever going to encounter, here are some *heuristics* which can help one to decode information and develop solutions.[3]

[3] A popular book by George Polya, *How to Solve It*, (Doubleday & Company, 1957), presents the subject in the context of teaching and learning mathematics.

To solve a problem:

1. **Extract the Essential Information**. The statement of the problem should specify an unambiguous goal (answer, output, product) and often (but not always) provides *input* (givens). The information could be numerical, verbal or pictorial. The statement of the problem may include hints for directions to take along a solution path. Sometimes *irrelevant* information for a particular solution is given; this needs to be identified and disregarded.
2. **Categorize**. Try to identify the class of problem. What is the topic? Is this particular problem similar to anything you have seen before? Is there a familiar example? Can the problem be broken down into a set of smaller problems?[4]
3. **Use Visual Aids**. It usually helps to write something down, or at least try to visualize something. Chemists drew pictures of molecules as tiny connected spheres of atoms a hundred years before electron microscopes confirmed their existence. Data, equations, graphs, tables all help to organize one's thinking. Some of the procedures we will consider in this book will be in mathematical equation form, some in graphical form, some in tabular form. It is amazing how useful a picture of a chemical process, such as mixing two solutions, can be to the understanding of a calculation.
4. **Work in Two Directions**. Do not be afraid to work backwards (that is, from the point of view of the desired result, *not* from the answer at the back of the book). It is difficult to get where you are going if you don't know where it is. It is entirely acceptable to construct the solution path through the forest in initially unconnected parts, so long as they eventually link together.
5. **Try Something**. Don't be afraid to *guess*. Try to construct a simpler example of the problem and work that through to discover the process of solution. While there may be only one correct answer, sometimes we get the feeling there is only one way to obtain that answer. Most situations have several approaches to understanding (just note the variety of opinions on almost any subject). Most problems have a number of solution methods. Yours may be unique, yet equally valid to some other method.[5] Usually solutions are arrived at after several false starts. The unconscious

[4] "Life by the yard is hard, life by the inch is a cinch."

[5] One of President Teddy Rosevelt's contributions to humanity was a novel proof of the Pythagorean Theo-

mind can be as valid a tool as the conscious mind.

6. **Check Your Work**. Are you sure you answered the correct question? Is your answer reasonable? If your checkbook balances to something larger than the national debt, it may be time to hire an accountant (or perhaps an attorney!). Repeat the process leading to solution, with refinement where possible.
7. **Derive Defensively**. Textbook answers may not be given, or may be hidden among false and partially-correct possibilities, or may even be incorrect. In the real world, answers are usually neither given nor are they known. How can you be sure you have the right answer? As you work through a solution or derivation, imagine a little imp on your shoulder which nags at you constantly with questions like, "Are you sure that's right?", or "Can you prove it?". Get in the habit of proving to yourself that your work is correct.
8. **Practice**. There is no substitute for experience. Practice makes perfect. Repeat the process leading to the answer on different examples until you are confident it is correct. Patience can be a virtue. A useful strategy is to make up your own problems. You may be surprised how many times something similar or identical appears on examinations (since exams are supposed to find out if you have arrived at a certain level of understanding and competence).

1.2. Artificial Intelligence and the Algorithmic Approach

The development of computer technology has revolutionized many aspects of our society, not the least of which are the ways we think about problems and the ways we go about solving them. The field of computer science addresses issues in logic, language, game theory, and artificial intelligence, among others. All of these are areas of human thinking and processing and we have learned new approachs to problem solving from building and using computers.

---

rem (of which there are dozens). When children were given the problem of connecting nine dots arranged in a square matrix with as few straight lines as possible without lifting the pencil from the paper, one creative individual folded the paper with the dots and pushed a pencil through the paper connecting the dots with a single line! (Four connected straight lines on flat paper is the standard expected answer.)

The 1968 Nobel Prize in Economics was awarded to Herbert Simon, who pioneered the application of using computers to discover natural laws. Simon was able to "teach" a computer to "discover" the inverse proportional relationship between volume and pressure of a gas, known as Boyle's Law, and to find the family behavior of the chemical elements, known as the Periodic Law.[6] Discovering scientific laws using the computer reflects non-trivial human thought processes, and this accomplishment was a scientific milestone.[7]

Much remains to be learned about creativity and thought processes. Much remains to be learned about creativity and thought processes. Humans exhibit such a wide variety of thinking styles that it would be inappropriate at this stage to declare any particular one as better than any other. Superficially, some of this textbook can be thought of as a "cookbook" contains "recipes" for solving a variety of problems one might encounter in an introductory chemistry course. If your style is to "plug and chug," you are welcome to go directly to the boxed procedures and follow the recipes. On the other hand, if you want to learn *how* to catch fish, you will want to learn how to develop your own recipes for success. Developing the skill to invent solutions to problems is termed an *algorithmic approach* to problem solving.

Algorithms are planned procedures for producing guaranteed (consistent and reproducible) results. **Heuristics** are planned procedures for producing results as well; however, they lack the component of *proof*. Assembly instructions, operation handbooks, diagnostic and repair manuals all employ algorithms to achieve their stated purpose. Heuristics include cooking recipes, surgical procedures and legal processes; hopefully they produce their desired purposes. Algorithms and heuristics are somewhat like road maps; they tell which route to take to get from one place to another.[8] We are all familiar with algorithms; we use them to calculate change, tie our shoes, cook meals, and change automobile tires. For

---

[6] Simon was awarded the prize specifically for his work on economic theory, but the computer science community claims him as the "first Nobel laureate in computer science," a field for which there has been no specific recognition by the Nobel committee.

[7] Current alternatives to digital electronic computers include analogue computers that work on mechanical principles, and neural network machines which attempt to mimic biological learning and memory processes.

[8] Printed road maps, however, don't give all the information needed to make a successful journey. Hazards and repairs are not given for example. Because of this road maps may be more like heuristics than algorithms since they show a possible but not guaranteed strategy.

example, an algorithm to escape a maze follows a path which follows walls along just one side (right or left) of the path. It may not be the shortest route to an exit, but it can be shown to be a certain route.

In problem solving, the starting place is the statement of the problem, the destination is the final answer, and a *solution* is a route to take to get from the question to the answer. Algorithms and heuristics are courses of action which require something to act on something else to produce a result. A recipe without a cook and some ingredients by itself will not produce a cake. Humans are trained and computers programmed to assemble and maneuver automobiles, write fugues, follow dress patterns, call square dances, play chess, yes, and even solve chemistry problems. A cooking recipe gives a specific procedure to follow to produce a specific product. The basic ingredients of scientific problems are the input data, the recipe is the algorithm which manipulates the input to produce the answer, and the final dish is the answer. If the answer isn't correct, perhaps the wrong ingredients (*data*) were used, or perhaps the wrong recipe (*algorithm or heuristic*) was used, or perhaps the recipe (*procedure*) wasn't followed (implemented) correctly.

1.3. Science and Algorithms

There is a subtle difference between algorithms and heuristics, related to truth. In the sense that nothing can be proven absolutely true, the best that can be hoped for is that a given conclusion is *consistent* within the context of the system used to derive it. In science no conclusion can be better than the premises on which it is based. Mathematical equations are used to describe physical systems. These are referred to as mathematical *models*. It may be possible to use mathematical logic to derive a certain result from a mathematical model of a physical system, but although it is consistent with the model, the result may not describe a measurable property of the system with the desired degree of accuracy. In this case, the derivation is algorithmic, but the model is heuristic. Thus science uses both algorithms and heuristics. Confusion about scientific statements often results from not making a clear distinction between the two.

Distinctions between algorithms and heuristics are particularly important to scientific endeavors such as chemistry. Until the major premise that gross matter is made up of tiny components became established in the Twentieth Century, most of chemical insights were heuristic in nature. That is, chemical models were useful tools for predicting and controlling the behavior of matter, but they were not provable in the sense that logic is used in mathematics. To the degree that chemistry is still an open subject, not fully derivable from basic principles, it can be challenging to discern the difference between fact and fiction, reality and wishful thinking.

Where do heuristics and algorithms come from? They are the products of intuition (e.g. it feels like rain, perhaps I'll take an umbrella), invention (e.g. a better mouse trap), innovation (e.g. the model T Ford automobile plant - the first assembly line using interchangeable parts), insight (e.g. I wonder what things would look like if I traveled on a light beam?), in short, some form of intellectual activity. Ingenuity plays a role, and developing new algorithms and heuristics takes practice. In this book we will attempt to describe the concepts of chemistry so that algorithms and heuristics used by chemists can be appreciated and applied to problem solving.

1.4. How to Develop an Algorithm

Charles L. Dodgson was a Nineteenth Century mathematical lecturer at Oxford University who entertained the Dean's daughters with manufactured stories that were published as *Alice's Adventures in Wonderland*, under the pen name Lewis Carroll. Carroll the story-teller couldn't entirely ignore Dodgson the mathematician, and numerous logical amusements accompany Alice's fantastic adventures. Witness Alice's encounter with the Cheshire-Cat:[9]

"Would you tell me, please, which way I ought to go from here?"

"That depends a good deal on where you want to get to," said the Cat.

"I don't much care where---" said Alice.

"Then it doesn't matter which way you go," said the Cat.

"---so long as I get *somewhere*," Alice added as an explanation.

"Oh, you're sure to do that," said the Cat, "if you only walk long enough."

If this situation sounds hypothetical, it is not very different from what happens the first time a human tries to communicate with a computer, or, for that matter, the first time a student tries to solve a new type of problem.

---

[9] *The Complete Works of Lewis Carroll*, Random House, Inc., New York, nd, pp 71-72.

Developing an algorithm to solve a scientific problem requires a clear understanding of the *principles* involved, whether they be mathematical, physical or chemical. In developing our algorithms, we will start with the underlying principles. Part of the fun of developing algorithms is *reducing principles to simple, precise and clear statements*. This usually translates to *understanding* the principles and how they may be applied to obtain information. Once the principles are understood, various applications should become apparent.

In addition to understanding principles, algorithm development requires logic skills, intuition, and experience. Life would be easier if we could give you a universal algorithm to develop algorithms (the "algorithm of algorithms," if you will). What we can do here is to illustrate possible approaches. One approach that may be useful is to pretend that you are explaining the problem and its solution to someone else. This works best if you imagine you are talking to the person on the phone, or writing a letter to them. You may even find it useful to talk to your calculator. "Now this is what I want you to do ..." As you give the directions, you will see what questions are relevant, how to organize the information and your thinking, and what steps you may have overlooked in the solution process.

Don't expect instant success. When Bertrand Russell began his *magnum opus* on the foundations of mathematics at the beginning of the Twentieth Century with Alfred North Whitehead, he describes how he would come to his office, day after day, and sit in front of a blank sheet of paper. Following several agonizing weeks with nothing to show on paper, the ideas began to flow. The resulting seminal work on the fundamental algorithms of mathematics was appropriately titled *Principia Mathematica*, the same title Isaac Newton had used two centuries earlier to announce the algorithms of fundamental physics to the world. Similarly, Newton himself described his first encounter with Euclid's *Elements* (of geometry). He states that he could barely understand the beginning of the treatise, so he laid it aside for some months. The next time he tackled the *Elements*, he was able to read it straight through, and went on to write his first mathematical paper (at age 13!). We may not all be Newtons, but there is a lesson for all of us here. Information that accumulates or builds on previous information, requires understanding of each point before the next can be understood. Before going to the next idea, it is wise to try to understand each concept on which it is based as well as possible.[10]

---

[10] Three R's of comprehending accumulative information: reread, review, refer (to another source for comparison, or explanation).

An example of a process that is often taken for granted, but is actually rather complicated, is arithmetic (addition, subtraction, multiplication, division, powers and roots of numbers). Familiarity, derived from much practice, makes the process seem automatic, but attempting to *deduce* precise, correct procedures from fundamental principles, which can be communicated and explained clearly to someone unfamiliar with the procedures, reveals the effort required to develop general, efficient algorithms.

Example 1.1 Develop an algorithm for multiplying two positive decimal integers.

One possible way to proceed is to recall memorizing multiplication "tables". Based on that experience, one could develop the following algorithm: to multiply two positive decimal integers, locate a table of all possible products of integers and look up the result. Needless to say, this becomes inefficient for arbitrarily large numbers. Another possibility, based on recalling the definition of multiplication as multiple additions, could proceed by adding one of the integers n times, where n is the other integer. This too becomes inefficient for large integers. Note that these are not *incorrect* algorithms, merely *impractical* algorithms.

A better way to proceed is to break the problem down into smaller problems from which the final result can be accumulated. "Long-hand" multiplication using a pencil and paper is based on the understanding of the notation for integers as expansions of products of digits times powers of the base (the base is 10 for decimal integers), and the distributive and associative laws of arithmetic. These properties allow the multiplication of arbitrarily large integers to be reduced to the sum of partial products obtained from the multiplication of just two digits at a time. Thus multiplication of decimal integers reduces to looking up products in a short table of products of single digits (containing 55 unique entries). (According to position notation, multiplication of a number by a power of the base simply appends a number of zeros to the right of the number, equal to the value of the power.) As example, consider the following expansion of the two digits and the accumulation of the products according to the rules of associativity and distributivity:

$$\begin{aligned}
123 \times 456 &= (1 \times 10^2 + 2 \times 10^1 + 3 \times 10^0) \times (4 \times 10^2 + 5 \times 10^1 + 6 \times 10^0) \\
&= (1 \times 10^2 \times 4 \times 10^2 + 1 \times 10^2 \times 5 \times 10^1 + 1 \times 10^2 \times 6 \times 10^0) \\
&\quad + (2 \times 10^1 \times 4 \times 10^2 + 2 \times 10^1 \times 5 \times 10^1 + 2 \times 10^1 \times 6 \times 10^0) \\
&\quad + (3 \times 10^0 \times 4 \times 10^2 + 3 \times 10^0 \times 5 \times 10^1 + 3 \times 10^0 \times 6 \times 10^0)
\end{aligned}$$

$$= (4 \times 10^4 + 5 \times 10^3 + 6 \times 10^2) + (8 \times 10^3 + 10 \times 10^2 + 12 \times 10^1) + (12 \times 10^2 + 15 \times 10^1 + 18 \times 10^0)$$

$$= (40000 + 5000 + 600) + (8000 + 1000 + 120) + (1200 + 150 + 18)$$

$$= 56088$$

Note that associativity has been used to isolate multiplication of single digits from multiplication of powers of the base (e.g. $1 \times 100 \times 4 \times 100 = 1 \times 4 \times 100 \times 100$).

Generalizing the process outlined in the example yields an algorithm for multiplying arbitrary integers in an arbitrary base:

Integer Multiplication Algorithm

Purpose: To obtain the product of two arbitrary decimal integer numbers.

Procedure:

1) Expand the digits of each of the given numbers into sums according to the place notation of integers. That is, for each digit of the number scanned from right to left, repeatedly multiply the digit by the base raised to the power of the position of the digit minus one.
2) Expand the product of the sums according to the distributive property.
3) Replace the products of digits according to the multiplication table for digits.
4) Replace the products of powers of the base by appending a number of zeros to the right of each product in the previous step, equal to the sum of the powers of the base of each multiplicand.
5) Sum the sub-products to obtain the result.
6) Assign the sign of the product to be the product of the signs of the numbers.

This procedure may appear rather complicated, but what it lacks in simplicity it gains in generality, what it appears to lack in clarity it gains in precision. Some effort may be required to understand the steps, but once mastered, it can be applied with confidence to any situation involving integer multiplication.

The Integer Multiplication Algorithm algorithm illustrates a number of features of algorithms in general, such as the use of logical operations and mathematics, and following a sequence of steps to a final result. Notice the use of nested operations in step 1; this happens with repetitive processes. It is not uncommon for one algorithm to apply sub-algorithms in its steps. In fact, reducing a process to its most fundamental steps could involve many layers of subprocesses.

Digital computers can do the equivalent of long-hand decimal integer multiplication efficiently by working in a binary base representation, which has only two digits, 0 and 1 (called *bi*nary digi*ts*, or *bits*). Arbitrary integers are represented as binary numbers by a string of binary digits (0 or 1), representing a sum of binary digits (0 or 1) times powers of the base (2), similar to decimal number place notation. Scanning the digits of one number (step 1) is effected by shifting the number one place (bit) to the left and reading the last digit (bit). Multiplication of a given number by a binary digit (step 3) simply produces 0 (if the binary digit is 0), or reproduces the given number (if the binary digit = 1). Multiplication of powers of the base (step 4) is effected by shifting to the left (and filling in a zero). Step 5 becomes binary addition. The process is very efficient because binary operations (the subprocesses) are very basic, efficient process in digital computers, and therefore among the fastest operations.

Summary

Problems ask questions, solutions find the answers. There is no universal way to solve all problems; heuristics are guesses to solutions and algorithms are systematic procedures for obtaining answers to selected classes of problems.

Problem solving requires a clear understanding of what is given and what is asked. There may be many bridges from question to answer. Clearly stated procedures for correct, efficient solution paths usually requires introspection, creativity and experience.

PROBLEM SOLVING EXERCISES

1. What are some possibilities if your answer to an exercise doesn't agree with the published answer?
2. Analyze the statement: chemistry should be reasonable.
3. Can you think of some other logical traps Alice could fall into during her conversation with the Cheshire-Cat?
4. How would a binary digital computer multiply 123×456?
5. Write out an algorithm for balancing a checkbook.
6. Develop a strategy for never loosing at *tick-tack-toe*.
7. Develop a strategy for passing Chemistry.

PROBLEM SOLVING EXERCISE HINTS

1. There are at least three reasons two people may not agree on the implications of a given set of facts.
2. Consider the meanings of the words and the logic of the statement.
3. Are there any assumptions about existence and/or uniqueness?
4. First convert the multiplicands from decimal (base 10) to binary (base 2).
5. What is the input and what is the desired output? It may help to purchase a calculator that does arithmetic accurately.
6. With the proper algorithm, if you are the first player you need never lose; if your opponent is the first player the best they can do is draw against you. To explore all the possible plays and counter plays (called exhaustive search[11]) may be asking a bit much, even though there are only three possible first moves (center, side and corner), so concentrate on the case of being the first player.

[11] *Exhausting* may be more appropriate.

7. First register for the course?