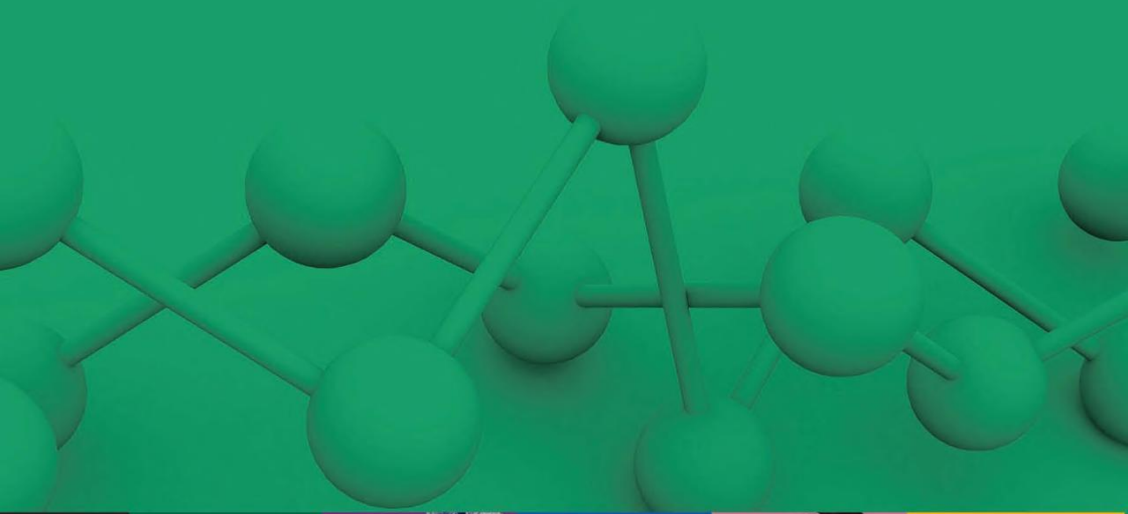
FUNDAMENTALS OF NATURAL COMPUTING

Basic Concepts, Algorithms, and Applications



Leandro Nunes de Castro

Chapman & Hall/CRC
Taylor & Francis Group

FUNDAMENTALS OF NATURAL COMPUTING
Basic Concepts, Algorithms, and Applications

Leandro Nunes de Castro
Catholic University of Santos (UniSantos)
Brazil

Chapman & Hall/CRC
Taylor & Francis Group
Boca Raton London New York

Chapman & Hall/CRC is an imprint of the
Taylor & Francis Group, an informa business

Cover designed by Sandro de Castro.

CRC Press
Taylor & Francis Group
6000 Broken Sound Parkway NW, Suite 300
Boca Raton, FL 33487-2742

Printed in the United States of America on acid-free paper
Version Date: 20110713

International Standard Book Number: 978-1-58488-643-3 (Hardback)

Library of Congress Cataloging-in-Publication Data

DeCastro, Leandro N., 1974-
Fundamentals of natural computing : basic concepts, algorithms, and applications / author, Leandro Nunes de Castro.
p. cm. -- (Chapman & Hall/CRC Computer and information science series)
Includes bibliographical references and index.
ISBN 1-58488-643-9 (9781584886433 : alk. paper)
1. Evolutionary programming (Computer science) 2. Neural networks (Computer science) 3. Quantum computers. 4. Molecular computers. I. Title. II. Series.

QA76.618.D42 2006
005.1--dc22 2006006350

CONTENTS

CHAPTER 1

FROM NATURE TO NATURAL COMPUTING

"... science has often made progress by studying simple abstractions when more realistic models are too complicated and confusing."
(I. Stewart, Does God Play Dice, Penguin Books, 1997, p. 65)

"Often the most profound insights in science come when we develop a method for probing a new regime of Nature."
(M. Nielsen and I. Chuang, Quantum Computation and Quantum Information, Cambridge University Press, 2000, p. 3)

1.1 INTRODUCTION

During the early days of humanity natural resources were used to provide shelter and food. We soon learned to modify and manage nature so as to breed crops and animals, build artifacts, control fire, etc. We then started to observe and study biological, chemical, and physical phenomena and patterns in order to better understand and explain how nature works. As examples, by learning about the physical laws of motion and gravity it became possible to design aircrafts; and by understanding some basic principles of life it is now possible to manage nature in various levels, from the creation of transgenic food to the control of diseases.

With the advent of computers, the way human beings interact with nature changed drastically. Nature is now being used as a source of inspiration or metaphor for the development of new techniques for solving complex problems in various domains, from engineering to biology; computers can simulate and emulate biological life and processes; and new material and means with which to compute are currently being investigated. *Natural computing* is the terminology introduced to encompass these three types of approaches, named, respectively: 1) *computing inspired by nature*; 2) *the simulation and emulation of natural phenomena in computers*; and 3) *computing with natural materials*. This book provides an introduction to the broad field of natural computing. It constitutes a textbook-style treatment of the central ideas of natural computing, integrated with a number of exercises, pseudocode, theoretical and philosophical discussions, and references to the relevant literature in which to gather further information, support, selected websites, and algorithms involving the topics covered here. This introductory chapter provides some motivations to study natural computing, challenges the student with some sample ideas, discusses its philosophy and when natural computing approaches are necessary, provides a taxonomy and makes a brief overview of the three branches of the proposed taxonomy for natural computing.

1.1.1. Motivation

Why should we study natural computing and why should research in this broad area be supported? There are many reasons for doing so; from the engineering of new computational tools for solving complex problems whose solutions are so far unavailable or unsatisfactory; to the design of systems presenting nature-like patterns, behaviors and even the design of new forms of life; and finally to the possibility of developing and using new technologies for computing (new computing paradigms). Although still very young in most of its forms, the many products of natural computing are already available in various forms nowadays, in washing machines, trains, toys, air conditioning devices, motion pictures, inside computers as virtual life, and so forth. Some of these applications will be reviewed throughout this book with varying levels of details.

Natural phenomena (e.g., processes, substances, organisms, etc.) have long inspired and motivated people to mimic, design, and build novel systems and artifacts. For many centuries, the observation of the natural world has allowed people to devise theories about how nature works. For example, physics is abounded with laws describing electromagnetism (Maxwell's equations), thermodynamics (first law: conservation, second law: entropy, and third law: absolute zero), motion (Newton's laws), and so forth. Artifacts, such as sonar echolocation, chemical substances used for pharmaceutical purposes, infrared imaging systems, airplanes, submarines, etc., were all developed by taking inspiration from nature, from animals (bats, birds, etc.) to chemical substances.

Natural computing is the computational version of this process of extracting ideas from nature to develop 'artificial' (computational) systems, or using natural media (e.g., molecules) to perform computation. The word artificial here means only that the systems developed are human-made instead of made by nature. While not the rule, in some cases, the products of natural computing may turn out to be so life-like that it becomes difficult to tell them apart from natural phenomena. Natural computing can be divided into three main branches (Figure 1.1) (de Castro and Von Zuben, 2004; de Castro, 2005):

1) *Computing inspired by nature*: it makes use of nature as inspiration for the development of problem solving techniques. The main idea of this branch is to develop computational tools (algorithms) by taking inspiration from nature for the solution of complex problems.
2) *The simulation and emulation of nature by means of computing*: it is basically a synthetic process aimed at creating patterns, forms, behaviors, and organisms that (do not necessarily) resemble 'life-as-we-know-it'. Its products can be used to mimic various natural phenomena, thus increasing our understanding of nature and insights about computer models.
3) *Computing with natural materials*: it corresponds to the use of natural materials to perform computation, thus constituting a true novel computing paradigm that comes to substitute or supplement the current silicon-based computers.

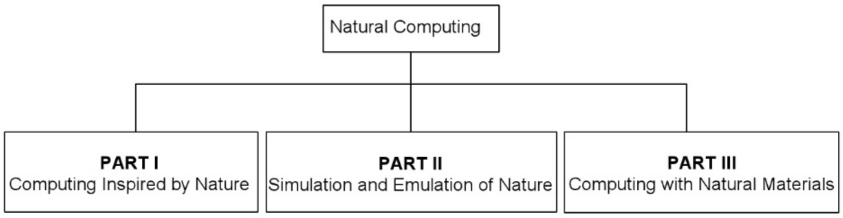

Natural Computing
PART I
Computing Inspired by Nature
PART II
Simulation and Emulation of Nature
PART III
Computing with Natural Materials

Figure 1.1: The three main branches of natural computing and their order of appearance in the book.

Therefore, natural computing can be defined as the field of research that, based on or inspired by nature, allows the development of new computational tools (in software, hardware or 'wetware') for problem solving, leads to the synthesis of natural patterns, behaviors, and organisms, and may result in the design of novel computing systems that use natural media to compute.

Natural computing is thus a field of research that testimonies against the specialization of disciplines in science. It shows, with its three main areas of investigation - *computing inspired by nature*, *the simulation and emulation of nature by means of computing*, and *computing with natural materials* - that knowledge from various fields of research are necessary for a better understanding of life, for the study and simulation of natural systems and processes, and for the proposal of novel computing paradigms. Physicists, chemists, engineers, biologists, computer scientists, among others, all have to act together or at least share ideas and knowledge in order to make natural computing feasible.

It is also important to appreciate that the development and advancement of natural computing leads to great benefits to the natural sciences, like biology, as well. Many computational tools developed using ideas from nature and the biological sciences are applied to create models and solve problems within the biosciences. This application domain is becoming even more important over the last few years with the emergence of new fields of investigation like computational biology and bioinformatics (Attwood and Parry-Smith, 1999; Baldi and Brunak, 2001; Waterman, 1995). Natural computing has also proven to be useful for a better understanding of nature and life processes through the development of highly abstract models of nature. Sometimes natural computing techniques can be directly aimed at being theoretical models of nature, providing novel insights into how nature works.

1.2 A SMALL SAMPLE OF IDEAS

The history of science is marked by several periods of almost stagnation, intertwined with times of major breakthroughs. The discoveries of Galileo, Newtonian mechanics, Darwin's theory of evolution, Mendel's genetics, the development of quantum physics, and the design of computers are just a small sample of the scientific revolutions over the past centuries. We are in the midst of another technological revolution - the *natural computing age*; a time when the interaction and the similarity between computing and nature is becoming each day greater. The transformation may be revolutionary for all those involved in the development of natural computing devices, but, if they do their job well, it will not necessarily make much difference for the end users. We may notice our spreadsheets recalculating faster, our grammar checker finally working, several complex problems being solved, robots talking naturally to humans, cars driving themselves, new forms of life, and patterns emerging in a computer screen in front of us, computers based on biomolecules, etc. But we will all be dealing with the end results of natural computing, not with the process itself. However, will ordinary people and end-users get a chance to experiment and play with natural computing? In fact, we can get our hands dirty already. And we will start doing this just by testing our ability to look at nature and computing in different ways.

Below are discussions about some natural phenomena and processes involving natural means: 1) clustering of dead bodies in ant colonies; 2) bird flocking; and 3) manipulating DNA strands. All of them have already served as inspiration or media for the development of natural computing techniques and will be presented here as a first challenge and motivation for the study of natural computing. Read the descriptions provided and try to answer the following questions.

To clean up their nests, some ant species group together corpses of ants or parts of dead bodies, as illustrated in Figure 1.2. The basic mechanism behind this type of clustering or grouping phenomenon is an attraction between dead items mediated by the ants. Small clusters of items grow by attracting more workers to deposit more dead bodies. This grouping phenomenon can be modeled using two simple rules:

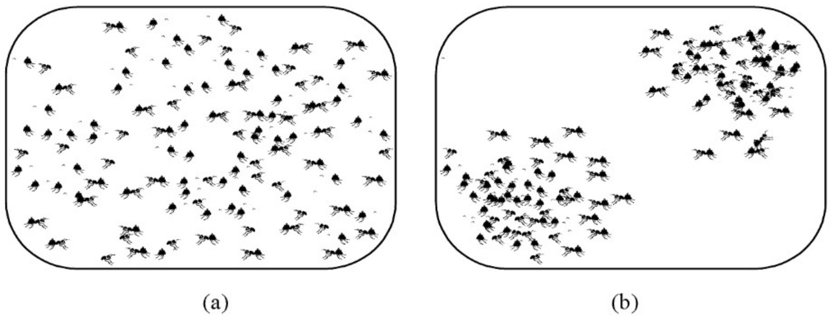


Figure 1.2: Clustering of dead bodies in an ant colony. (a) Initial distribution of ants. (b) Clustered bodies.

Pick up rule: if an ant finds a dead body, it picks it up and wanders around the arena until it finds another dead body. The probability or likelihood that an ant picks up a dead body is inversely proportional to the number of items in that portion of the arena; that is, the more dead bodies around, the smaller the probability it is picked up, and vice-versa.

Dropping rule: while wandering around, the loaded ant eventually finds more dead bodies in its way. The more dead bodies are found in a given region of the arena, the higher the probability the ant drops the dead body it is carrying at that location of the arena, and vice-versa.

As a result of these very simple behavioral rules, all dead items will eventually be brought together into a single group, depending on the initial configuration of the arena and how the rules are set up.

Question 1: what kind of problem could be solved inspired by this simple model of a natural phenomenon?

Question 2: how would you use these ideas to develop a computing tool (e.g., an algorithm) for solving the problem you specified above? ■

Figure 1.3 illustrates a bird flock. When we see birds flocking in the sky, it is most natural to assume that the birds 'follow a leader'; in this picture, the one in front of the flock. However, it is now believed (and there are some good evidences to support it) that the birds in a flock do not follow any leader.

There is no 'global rule' that can be defined so as to simulate a bird flock. It is possible, however, to generate scripts for each bird in a simulated flock so as to create a more realistic group behavior (for example, in a computer simulation). Another approach, one that is currently used in many motion pictures, is based on the derivation of generic behavioral rules for individual birds. The specification of some simple individual rules allows realistic simulation of birds flocking. The resultant flock is a result of many birds following the same simple rules.

Question 1: describe (some of) these behavioral rules that, when applied to each bird in the flock, result in an emergent group behavior that is not specifically defined by the individual rules. It means that such rules, together with the interactions among individual birds, result in a global behavior that cannot often be predicted by simply looking at the rules.

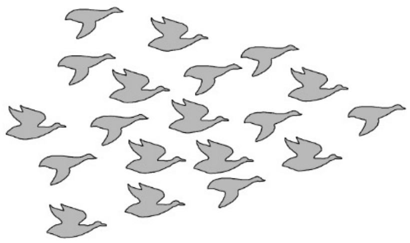


Figure 1.3: Illustration of a flock of birds.

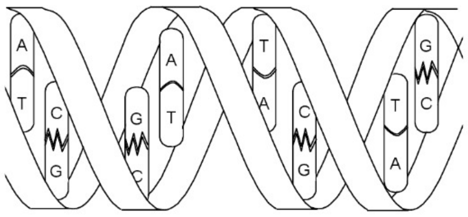


Figure 1.4: Double strand of DNA.

Question 2: can you extend these rules to herds of land animals and schools of fish? That is, is there a significant qualitative difference between these various types of group behavior? ■

Figure 1.4 depicts a double strand of DNA. The DNA molecules contain the genetic information of all living beings on earth. It is known that this genetic information, together with the environmental influences, determines the phenotype (expressed physical characteristics) of an individual.

Roughly, DNA molecules are composed of four bases which bind to each other exclusively in a complementary fashion: A binds with T, and C binds with G. Genetic engineering techniques can nowadays be used to artificially manipulate DNA so as to alter the genetic information encoded in these molecules. For instance, DNA molecules can be *denatured* (separated into single strands), *annealed* (single strands can be 'glued' together to form double strands of DNA), *shortened* (reduced in length), *cut* (separated in two), *multiplied* (copied), *modified* (e.g., new sequences inserted), etc.

Question 1: if the information that encodes an organism is contained in DNA molecules and these can be manipulated, then life can be seen as information processing. Based on your knowledge of how standard computers (PCs) work, propose a new model of computer based on DNA strands and suggest a number of DNA manipulation techniques that can be used to compute with molecules.

Question 2: what would be the advantages and disadvantages of your proposed DNA computer over the standard computers? ■

If you have not tried to answer these questions yet, please take your time. They may give you some flavor of what researchers on some branches of natural computing do. If you want to check possible answers to these questions, please refer to Chapters 5, 8, and 9 respectively

. . .

So, how did you get on?

If your answers were too different from the ones presented in this volume, do not worry; they may constitute a potentially new algorithm or computing paradigm!

1.3 THE PHILOSOPHY OF NATURAL COMPUTING

One important question this book tries to answer is how researchers discover the laws and mechanisms that are so effective in uncovering how nature functions and how these can be used within and for computing. A natural result of this line of investigation is the proposal of novel ways of computing, solving real-world problems, and synthesizing nature. Scientific explanations have been dominated by the formulation of principles and rules governing systems' behaviors. Researchers usually assume that natural systems and processes are governed by finite sets of rules. The search for these basic rules or fundamental laws is one of the central issues covered in this book. It is not easy to find such rules or laws, but enormous progress has been made. Some examples were provided in Section 1.2.

Most of the computational approaches natural computing deals with are based on highly simplified versions of the mechanisms and processes present in the corresponding natural phenomena. The reasons for such simplifications and abstractions are manifold. First of all, most simplifications are necessary to make the computation with a large number of entities tractable. Also, it can be advantageous to highlight the minimal features necessary to enable some particular aspects of a system to be reproduced and to observe some emergent properties. A common question that may arise is: "if it is possible to do something using simple techniques, why use more complicated ones?"

This book focuses on the extraction of ideas and design aspects of natural computing, in particular the teaching of modeling, how to make useful abstractions, and how to develop and use computer tools or algorithms based on nature. In contrast to some books on the technological and advanced aspects of specific topics, this text outlines the relations of theoretical concepts or particular technological solutions inspired by nature. It is therefore important to learn how to create and understand abstractions, thus making a suitable simplification of a system without abolishing the important features that are to be reproduced.

Which level is most appropriate for the investigation and abstraction depends on the scientific question asked, what type of problem one wants to solve, or the life phenomenon to be synthesized. As will be further discussed, simple behavioral rules for some insects are sufficient for the development of computational tools for solving combinatorial problems and coordinate collective robotic systems. These are also useful for the development of computer simulations of biological systems in artificial life, and the creation of abstract models of evolution, and the nervous and immune systems, all aimed at solving complex problems in various domains.

Natural computing usually integrates experimental and theoretical biology, physics and chemistry, empirical observations from nature and several other sciences, facts and processes from different levels of investigation into nature so as to design new problem solving techniques, new forms of mimicking natural phenomena, and new ways of computing, as summarized in Figure 1.5.

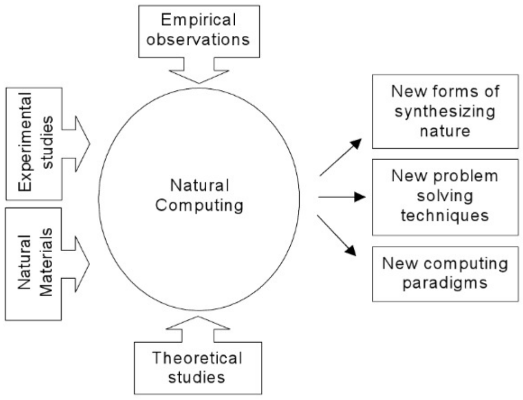


Figure 1.5: Many fields of investigation have to be integrated for the study and development of natural computing. As outcomes, new ways of computing, new problem solving techniques, and possible forms of synthesizing nature result.

1.4 THE THREE BRANCHES: A BRIEF OVERVIEW

This section provides a very brief overview of the three branches of natural computing and their main approaches. The bibliography cited is basically composed of references from pioneer works and books where further and didactic information can be found on all topics discussed. Instead of trying to cover the areas reviewed in detail, general comments about most natural computing tools, algorithms, techniques and their potential application areas are provided, together with a discussion of how the resultant computational tools or systems relate (interact) with nature and the natural sciences. The reader will be pointed to the chapters that deal specifically with each of the approaches discussed.

1.4.1. Computing Inspired by Nature

The main motivation for this part of the book is that nature has greatly enriched computing. More importantly, nature has been very successful in solving highly complex problems. In a very low level, there is an urge for survival in living organisms: they have to search for food, hide from predators and weather conditions, they need to mate, organize their homes, etc. All this requires complex strategies and structures not usually directly modeled or understood. But, for instance, viewing a colony of ants foraging for food as an 'intelligent behavior' is not always very intuitive for us, used to attribute 'intelligent behaviors' to 'intelligent beings'. What if I tell you that the way ants forage for food has inspired algorithms to solve routing problems in communication networks? Can you imagine how this is done?

Among all natural computing approaches, computational algorithms and systems inspired by nature are the oldest and most popular ones. They arose with two main objectives in mind. First, researchers were interested in the modeling of natural phenomena and their simulation in computers. The common goal in this direction is to devise theoretical models, which can be implemented in computers, faithful enough to the natural mechanisms investigated so as to reproduce qualitatively or quantitatively some of their functioning. Theoretical models are supposed to provide a deeper insight and better understanding of the natural phenomena being modeled, to aid in the critical analysis and design of experiments, and to facilitate the recovery of results from laboratory experimentation or empirical observations. There is a vast number of theoretical models available in the literature concerning all natural sciences, including biology, ethology, ecology, pharmacology, nutrition and health care, medicine, geophysics, and so forth.

However, the focus of computing inspired by nature, under the umbrella of natural computing, is most often on problem solving instead of on theoretical modeling, and this leads to the second objective of computing based on nature. The second objective, thus, involves the study of natural phenomena, processes and even theoretical models for the development of computational systems and algorithms capable of solving complex problems. The motivation, in this case, is to provide (alternative) solution techniques to problems that could not be (satisfactorily) resolved by other more traditional techniques, such as linear, non-linear, and dynamic programming. In such cases, the computational techniques developed can also be termed *bio-inspired computing* or *biologically motivated computing* (Mange and Tomassini, 1998; de Castro and Von Zuben, 2004), or *computing with biological metaphors* (Paton, 1994).

As computing inspired by nature is mostly aimed at solving problems, almost all approaches are not concerned with the creation of accurate or theoretical models of the natural phenomena being modeled. In many situations highly abstract models, sometimes called metaphors (Paton, 1992), are proposed mimicking particular features and mechanisms from biology. What usually happens is that a natural phenomenon, or a theoretical model of it, gives rise to one particular computational tool and this is then algorithmically or mathematically improved to the extent that, in the end, it bears a far resemblance with the natural phenomenon that originally motivated the approach. Well-known examples of these can be found in the fields of artificial neural networks and evolutionary algorithms, which will be briefly discussed in the following.

A landmark work in the branch of bio-inspired computing was the paper by McCulloch and Pitts (1943), which introduced the first mathematical model of a neuron. This neuronal model, also known as artificial neuron, gave rise to a field of investigation of its own, the so-called *artificial neural networks* (Fausett, 1994; Bishop, 1996; Haykin, 1999; Kohonen, 2000). Artificial neural networks, to be discussed in Chapter 4, can be defined as information processing systems designed with inspiration taken from the nervous system, in most cases the human brain, and with particular emphasis on problem solving. There are several types of artificial neural networks (ANNs) and learning algorithms used to set up (train) these networks. ANNs are distinct from what is currently known as *computational neuroscience* (O'Reilly and Munakata, 2000; Dayan and Abbot, 2001; Trappenberg, 2002), which is mainly concerned with the development of biologically-based computational models of the nervous system.

Another computing approach motivated by biology arose in the mid 1960's with the works of I. Rechenberg (1973), H. P. Schwefel (1965), L. Fogel, A. Owens and M. Walsh (Fogel et al., 1966), and J. Holland (1975). These works gave rise to the field of *evolutionary computing* (Chapter 3), which uses ideas from evolutionary biology to develop *evolutionary algorithms* for search and optimization (Bäck et al., 2000a,b; Fogel, 1998; Michalewicz, 1996). Most evolutionary algorithms are rooted on the neo-Darwinian theory of evolution, which proposes that a population of individuals capable of reproducing and subject to genetic variation followed by natural selection result in new populations of individuals increasingly more fit to their environment. These simple three processes, when implemented in computers, result in evolutionary algorithms (EAs). The main types of EAs are the *genetic algorithms* (Mitchell, 1998; Goldberg, 1989), *evolution strategies* (Schwefel, 1995; Beyer, 2001), *evolutionary programming* (Fogel, 1999), *genetic programming* (Koza, 1992, 1994; Bahnzaf et al., 1997), and *classifier systems* (Booker et al., 1989; Holmes et al., 2002).

The term *swarm intelligence* (Chapter 5) was coined in the late 1980's to refer to cellular robotic systems in which a collection of simple agents in an environment interact based on local rules (Beni, 1988; Beni and Wang, 1989). Nowadays, the term is being used to describe any attempt to design algorithms or problem-solving devices inspired by the collective behavior of social organisms, from insect colonies to human societies. Swarm intelligence has two main frontlines: algorithms based on the collective behavior of social insects (Bonabeau et al., 1999), and algorithms based on cultures or sociocognition (Reynolds, 1994; Kennedy et al., 2001). In the first case, the collective behavior of ants and other insects has led to the development of algorithms for solving combinatorial optimization, clustering problems, and the design of autonomous robotic systems. Algorithms based on cultures and sociocognition demonstrated effectiveness in performing search and optimization on continuous and discrete spaces.

Artificial immune systems (AIS) or *immunocomputing* (Chapter 6), borrow ideas from the immune system and its corresponding models to design computational systems for solving complex problems (Dasgupta, 1999; de Castro and Timmis, 2002; Timmis et al., 2003). This is also a young field of research that emerged around the mid 1980's. Its application areas range from biology to robotics. Similarly to ANNs, EAs and swarm intelligence, different phenomena, processes, theories and models resulted in different types of immune algorithms, from evolutionary-like algorithms to network-like systems. Several other (emerging) types of algorithms inspired by nature can be found in the literature. For instance, it is possible to list the *simulated annealing* algorithm, the systems based on *growth* and *development*, and the *cells and tissues* models (Kirkpatrick et al., 1983; Aarts and Korst, 1989; Paton et al., 2004; Kumar and Bentley, 2003; Glover and Kochenberger, 2003; de Castro and Von Zuben, 2004).

Figure 1.6 summarizes the main components of computing inspired by nature to be discussed in this book and the respective chapters.

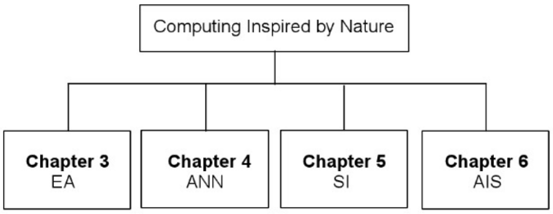


Figure 1.6: The main components of computing inspired by nature to be discussed in this book: ANN: artificial neural networks (neurocomputing); EA: evolutionary algorithms (evolutionary computing); SI: swarm intelligence; AIS: artificial immune systems (immunocomputing).

1.4.2. The Simulation and Emulation of Nature in Computers

While biologically inspired computing is basically aimed at solving complex problems, the second branch of natural computing provides new tools for the synthesis and study of natural phenomena that can be used to test biological theories usually not passive of testing via the traditional experimental and analytic techniques. It is in most cases a synthetic approach aimed at synthesizing natural phenomena or known patterns and behaviors. There is also a complementary relationship between biological theory and the synthetic processes of the simulation and emulation of nature by computers. Theoretical studies suggest how the synthesis can be achieved, while the application of the theory in the synthesis may be a test for the theory. There are basically two main approaches to the simulation and emulation of nature in computers: by using *artificial life* techniques or by using tools for studying the *fractal geometry of nature* (Figure 1.7).

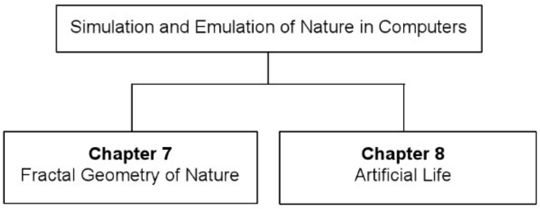

Simulation and Emulation of Nature in Computers
Chapter 7 Fractal Geometry of Nature
Chapter 8 Artificial Life

Figure 1.7: The two main approaches for the simulation and emulation of nature in computers and the chapters in which they are going to be presented.

Recent advances in computer graphics have made it possible to visualize mathematical models of natural structures and processes with unprecedented realism. The resulting images, animations, and interactive systems are useful as scientific, research, and educational tools in computer science, engineering, biosciences, and many other domains. One major breakthrough in the modeling and synthesis of natural patterns and structures is the recognition that nature is fractal in the sense that it can be successfully emulated by *fractal geometry* (Mandelbrot, 1983; Peitgen et al., 1992; Flake, 2000; Lesmoir-Gordon et al., 2000). In a simplified form, *fractal geometry* (Chapter 7) is the geometry of nature, with all its irregular, fragmented, and complex structures. In general, fractals are characterized by infinite details, infinite length, self-similarity, *fractal dimensions*, and the absence of smoothness or derivative. Nature provides many examples of fractals, for instance, ferns, coastlines, mountains, cauliflowers and broccoli, and many other plants and trees are fractals. Moreover, organisms are fractals; our lungs, our circulatory system, our brains, our kidneys, and many other body systems and organs are fractal.

There are a number of techniques for modeling fractal patterns and structures, such as *cellular automata* (Ilachinski, 2001; Wolfram, 1994), *L-systems* or *Lindenmayer systems* (Lindenmayer, 1968; Prusinkiewicz and Lindenmayer, 1990), *iterated function systems* (Hutchinson, 1981; Barnsley and Demko, 1985; Barnsley, 1988), *particle systems* (Reeves, 1983), *Brownian motion* (Fournier et al., 1982; Voss, 1985), and others. Their applications include computer-assisted landscape design, the study of developmental and growth processes, and the modeling and synthesis (and corresponding analysis) of an innumerable amount of natural patterns and phenomena. But the scope and importance of fractals and fractal geometry go far beyond these. Forest fires have fractal boundaries; deposits built up in electro-plating processes and the spreading of some liquids in viscous fluids have fractal patterns; complex protein surfaces fold up and wrinkle around toward three-dimensional space in a fractal dimension; antibodies bind to antigens through complementary fractal dimensions of their surfaces; fractals have been used to model the dynamics of the AIDS virus; cancer cells can be identified based on their fractal dimension; and the list goes on.

Artificial life (ALife) will be discussed in Chapter 8. It corresponds to a research field that complements traditional biological sciences concerned with the analysis of living organisms by trying to synthesize life-like behaviors and creatures in computers and other artificial media (Langton, 1988; Adami, 1998; Levy, 1992). Differently from nature-inspired computing, approaches in the ALife field are usually not concerned with solving any particular problem. ALife has, as major goals, to increase the understanding of nature (life-as-it-is), enhance our insight into artificial models and possibly new forms of life (life-as-it-could-be), and to develop new technologies such as software evolution, sophisticated robots, ecological monitoring tools, educational systems, computer graphics, etc.

ALife systems have, thus, been designed to simulate and emulate behaviors or organisms in order to allow the study or simulation of natural phenomena or processes. In most cases it emphasizes the understanding of nature, and applications as a problem solver are left in second plan. For instance, ALife systems have been created to study traffic jams (Resnick, 1994); the behavior of synthetic biological systems (Ray, 1994); the evolution of organisms in virtual environments (Komosinski and Ulatowski, 1999); the simulation of collective behaviors (Reynolds, 1987); the study and characterization of computer viruses (Spafford, 1991); and others. Its major ambition is to build living systems out of non-living parts; that is, to accomplish what is known as 'strong ALife' (Sober, 1996; Rennard, 2004).

1.4.3. Computing with Natural Materials

Computing with natural materials is concerned with new computing methods based on other natural material than silicon. These methods result in a non-standard computation that overcomes some of the limitations of standard, sequential John von Neumann computers. As any mathematical operation can be broken down into bits, and any logical function can be built using an AND and a NOT gate, any computable 'thing' can be worked out by appropriately wired AND and NOT gates. This independence of a specific representation makes it possible to use new concepts for the computational process based on natural materials, such as, chemical reactions, DNA molecules, and quantum mechanical devices.

The history of computer technology has involved a sequence of changes from one type of realization to another; from gears to relays to valves to transistors to integrated circuits. Nowadays, a single silicon chip can contain millions of logic gates. This miniaturization of the most basic information processing elements is inevitably going to reach a state where logic gates will be so small so as to be made of atoms. In 1965 G. Moore (1965) observed that there is an exponential growth in the number of transistors that can be placed in an integrated circuit. According to what is now known as the "Moore's law", there is a doubling of transistors in a chip every couple of years. If this scale remains valid, by the end of this decade, silicon-based computers will have reached their limits in terms of processing power. One question that remains thus, is "What other materials or media can be used to perform computation in place of silicon?". Put in another form, after certain level of miniaturization of the computing devices, the standard physical laws will be no longer applicable, because quantum effects will begin to take place. Under this perspective, the question that arises refers to "How should we compute under quantum effects?".

Computing with natural materials is the approach that promises to bring a major change in the current computing technology in order to answer the questions above. Motivated by the need to identify alternative media for computing, researchers are now trying to design new computers based on molecules, such as membranes, DNA and RNA, or quantum theory. These ideas resulted in what is now known as *molecular computing* (Păun et al., 1998; Gramß et al., 2001; Calude and Păun, 2001; Păun and Cutkosky, 2002; Sienko et al., 2003) and *quantum computing* or *quantum computation* (Hirvensalo, 2000; Nielsen and Chuang, 2000; Pittenger, 2000), respectively. Figure 1.8 summarizes the main components of computing with natural materials.

Molecular computing is based upon the use of biological molecules (biomolecules) to store information together with genetic engineering (biomolecular) techniques to manipulate these molecules so as to perform computation. It constitutes a powerful combination between computer science and molecular biology. The field can be said to have emerged due to the work of L. Adleman who, in 1994, solved an NP-complete problem using DNA molecules and biomolecular techniques for manipulating DNA (Adleman, 1994). Since then, much has happened: several other 'molecular solutions' to complex problems have been proposed and molecular computers have been shown to perform universal computation. The main advantages of molecular computing are its high speed, energy efficiency, and economical information storage. An overall, striking observation about molecular computing is that, at least theoretically, there seem to be many diverse ways of constructing molecular-based universal computers. There are, of course, the possibility of errors and difficulties in implementing real molecular computers. When compared with the currently known silicon-based computers, molecular computers offer some unique features, such as the use of molecules as data structures and the possibility of performing massively parallel computations. In Chapter 9, this book reviews one particular molecular computing approach, namely, *DNA computing*.

When the atomic scale of logic gates is reached, the rules that will prevail are those of *quantum mechanics*, which are quite different from the classical rules that determine the properties of conventional logic gates. Thus, if computers are to become even smaller in the (not so far) future, quantum technology must complement or supplement the current technology. *Quantum computation* and *quantum information*, introduced in Chapter 10, is the study of the information processing tasks that can be accomplished using quantum mechanical systems (Nielsen and Chuang, 2000). In quantum computers information is stored at the microphysical level where quantum mechanisms prevail.

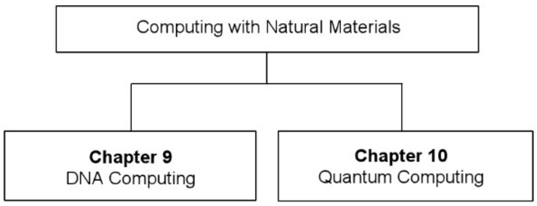


Figure 1.8: The two main branches of computing with natural materials or media.

In such cases, a bit could represent both zero and one simultaneously, and measurements and manipulations of these quantum bits are modeled as matrix operations. The seminal paper by R. Feynman (1982) introduced a computer capable of simulating quantum physics. A little later on, D. Deutsch (1984) published a paper where he demonstrated the universal computing capability of such quantum computers. Another seminal work that served to boost the field was the paper by P. Shor (1994) introducing the first quantum algorithm capable of performing efficient factorization, something that only a quantum computer could do. What is important to remark about quantum computing, though, is that it can provide entirely novel types of computation with qualitatively new algorithms based on quantum mechanics; quantum technology thus offers much more than simply adding the capability of processing more bits using the current silicon-based computers. Therefore, quantum computing aims at nontraditional hardware that would allow quantum effects to take place.

1.5 WHEN TO USE NATURAL COMPUTING APPROACHES

While studying this book, the reader will be faced with a diverse range of problems to be solved, phenomena to be synthesized and questions to be answered. In all cases, one or more of the natural computing approaches briefly reviewed above will be used to solve the problem, synthesize the phenomenon or answer the question. However, it is important to acknowledge that natural computing is not the only field of investigation that provides solutions to these, nor is it always the most suitable and efficient approach. To clarify when natural computing should be used, let us present some examples and arguments in each of the three branches. Let us assume that you have just finished your undergraduation course and now have an engineering or science degree in hand. In your first interview for a job in a major company, you are posed with three problems and given some time to provide solutions to them.

Problem 1: the company is expanding rapidly and now wants to build a new factory in a country so far unattended. The site where the factory is to be built has already been chosen as long as the cities to be attended by this factory. Figure 1.9 depicts the scenario. The problem is: given all the cities in the map find the smallest route from the factory to all cities passing by each city exactly once and returning to the departure city (factory). This problem is well-known from the literature and is termed *traveling salesman problem* (TSP).

Such problems have several practical applications, from fast food delivery to printed circuit board design. Although this problem may be simple to state it is hard to solve, mainly when the number of cities involved is large. For example, if there is one factory plus three other cities, then there are 6 possible routes; if there is one factory plus four other cities, then there are 24 possible routes; if there is one factory plus five other cities, then there are 120 possible routes; and so on. This corresponds to a factorial growth in the number of possible routes in relation to the number of cities to be attended.

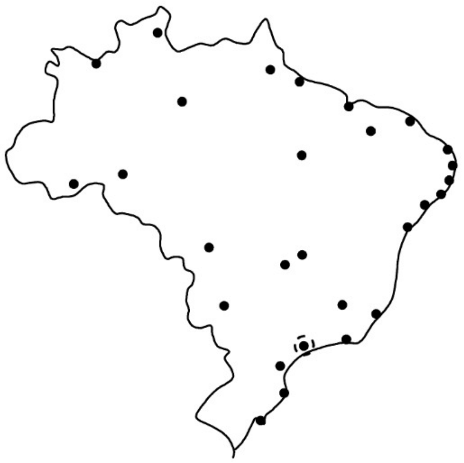


Figure 1.9: Map of the new country to be attended by the company. The city where the factory is going to be built is detached together with the 27 cities to be attended.

The most straightforward solution you could provide to this problem is to suggest the testing of all possible routes and the choice of the smallest one; an approach we usually call 'brute force' or 'exhaustive search'. Although simple, this approach is only efficient for a small number of cities. Assuming your computer is capable of analyzing 100 routes per second, it would take much less than a second to solve a three cities instance of this problem and a bit more than a second to solve a five cities instance of the problem. For the problem presented, however, the scenario is much different: there are 27! possible routes, and this corresponds to approximately 1.1×10^{28} possible routes to be tested. In your computer, the exhaustive search approach would take, approximately, 3.0×10^{23} hours, or 1.3×10^{22} days, or 3.5×10^{19} years of processing time to provide a solution. ■

Problem 2: the expansion plans of the company include the development of motion picture animations. The first animation to be designed involves a herd of zebras running away from a few hungry lions. Your task is to propose a means of realistically and effectively simulating the collective behavior of the zebras.

The first proposal you may come out with is to write a *script* for each zebra; that is, a computer program that fully describes the action of the zebras in the field. This, of course, would seem easier than writing a single script to coordinate the whole herd. Again, though simple, this approach has some drawbacks when the number of zebras involved is large and when more realistic scenarios are to be created. First, as the script is static, in the sense that the behaviors modeled do not change over time, the resultant collective behavior is the same every time the simulation is run. Second, the number of scripts to be written grows with the size of the herd and the complexity of the scripts grows in a much larger scale, because the more zebras, the more difficult it becomes to coordinate the behavior of all animals so as to avoid collisions, etc., and all aspects of the collective behavior have to be accounted for within the script.

In such cases, it is much more efficient and realistic to try and find a set of simple rules that will be responsible for guiding the individual behavior of the zebras across the field, similarly to the discussion presented in Section 1.2 for a bird flock. More details on how to solve this type of problem will be provided in Chapter 8.

To make your task more complete and challenging, the interviewer also asks how you would create the surrounding environment in which the animals will be placed. The simplest approach, in this case, would be to draw the scenarios or maybe photograph a real scenario and use it as a background for the animation, as sometimes done in cartoons. The drawing or pictures have to be passed, frame by frame, and the animals placed on this environment for running the simulation. A problem with this solution is the amount of memory required to store all these images (drawings or photographs). For instance, a digital photo of medium quality requires around 500KB of memory to be stored.

An efficient way of reproducing natural scenarios and phenomena is by using techniques from fractal geometry (Chapter 7). Brownian motion, cellular automata, L-systems, iterated function systems, and particle systems can be used to generate scenarios of natural environments; simulate fire, chemical reactions, etc., all with unprecedented realism and, in most cases, at the expense of a small amount of memory usage. These techniques usually employ a small number of rules or equations that are used to generate patterns or simulate phenomena *on line*, thus reducing the need for huge storage devices. ■

Problem 3: in the current computer technology, bits constitute the most basic unit of information. They are physically stored in integrated circuits or chips based on silicon technology. The need to increase the memory capacity and processing speed of computers forced the chips to be able to accommodate more and more bits, thus promoting a miniaturization of the electronic devices. To have an idea of the dramatic decrease in size of electronic devices, by the year 2000 approximately 10^9 atoms were necessary to represent a single bit of information. It is estimated that by the year 2020 a single atom will be used to represent one bit of information. The question is: if we will unavoidably reach the miniaturization limit of current computer technology, what are the other types (than silicon) of materials that could be used for computing?

Note that the answer to this question may involve a change of computing *paradigm*. That is, if we are to investigate the possibility of computing with a material that is different from silicon, then a completely new type of storing and manipulating information may be used. An example of a potential novel material for computing was given in Section 1.2 with the use of DNA. In that case, information was stored in DNA molecules and genetic engineering techniques were proposed as means to manipulate DNA for computing (Chapter 9). ■

To summarize, natural computing approaches almost invariably correspond to alternative solution techniques to all the problems discussed in this volume and many others to which they are applicable. In many situations there are other solution techniques for a given problem and these may even provide superior results. Thus, it is important to carefully investigate the problem to be solved before choosing a specific solution method, be it a natural computing tool or a different one. To give some hints on when natural computing should be used, consider the following list. Natural computing could be used when:

- The problem to be solved is complex, i.e., involves a large number of variables or potential solutions, is highly dynamic, nonlinear, etc.
- It is not possible to guarantee that a potential solution found is optimal (in the sense that there is no better solution), but it is possible to find a quality measure that allows the comparison of solutions among themselves.
- The problem to be solved cannot be (suitably) modeled, such as pattern recognition and classification (e.g., vision) tasks. In some cases, although it is not possible to model the problem, there are examples (samples) available that can be used to 'teach' the system how to solve the problem, and the system is somehow capable of 'learning from examples'.
- A single solution is not good enough; that is, when diversity is important. Most standard problem-solving techniques are able to provide a single solution to a given problem, but are not capable of providing more than one solution. One reason for that is because most standard techniques are deterministic, i.e., always use the same sequence of steps to find the solution, and natural computing is, in its majority, composed of probabilistic methods.
- Biological, physical, and chemical systems and processes have to be simulated or emulated with realism. Euclidean geometry is very good and efficient to create man-made forms, but has difficulty in reproducing natural patterns. This is because nature is fractal, and only fractal geometry provides the appropriate tools with which to model nature.
- Life behaviors and phenomena have to be synthesized in artificial media. No matter the artificial media (e.g., a computer or a robot), the essence of a given natural behavior or pattern is extracted and synthesized in a, usually, much simpler form in artificial life systems.
- The limits of current technology are reached or new computing materials have to be sought. Nature abounds with information storage and processing systems, and the scientific and engineering aspects of how to use these natural materials to compute are the main challenges of the third branch of natural computing: computing with natural materials.

These prerequisites may not sound so clear yet, but you will naturally have a better idea of why, when, and how natural computing should or have been used as you progress in the book.

1.6 SUMMARY

Natural computing is the terminology used to refer to three types of systems: 1) computational algorithms for problem-solving developed by taking inspiration from natural phenomena; 2) computational systems for the simulation and/or emulation of nature; and 3) novel computing devices or paradigms that use media other than silicon to store and process information.

Although all these branches are quite young from a scientific perspective, several of them are already being used in our everyday lives. For instance, we now have 'intelligent' washing machines, games, (virtual) pets, control systems, etc., all based on, or using, computing devices inspired by nature; research on ALife and the fractal geometry of nature has allowed the creation of realistic models of nature and the simulation and emulation of several plants and animal species, and has aided the study of developmental processes and many other natural phenomena; and computing with natural materials has given new insights into how to complement or supplement computer technology as known currently.

Natural computing is highly relevant for today's computer scientists, engineers, biologists, and other professionals because it offers alternative, and sometimes brand new, solutions to problems yet unsolved or poorly resolved. It has also provided new ways of seeing, understanding, and interacting with nature. There is still much to come and much to do in such a broad and young field of investigation as natural computing. However, we now know for sure that this is not only a promising field of research; its many applications, outcomes, and perspectives have been affecting our lives, even if this is not perceived by most people. And this is just the beginning. Welcome to the natural computing age!

1.7 QUESTIONS

1. Find evidences in the literature that support the idea that birds in a flock do not follow a leader.
2. The movie "A Bug's Life" by Disney/Pixar starts with an ant colony harvesting for food. But instead of harvesting food for the colony, the ants were harvesting for the grasshoppers, who obliged them to do so. The long line of ants carrying food is stopped when a leaf falls down on their way, more precisely on their *trail*. The ants in front of the line just interrupted by the leaf start despairing, claiming they do not know what to do; that is, where to go. Another 'more instructed' ant suggests that they might walk around the leaf thus keeping their harvesting. One ant even claims that such catastrophe was not nearly as bad as the "twig of 93".

   Based upon this brief summary of the beginning of the movie "A Bug's Life", what can you infer about the way ants forage and harvest food, more specifically, why did the ants loose their direction when the leaf fell in their trail?

3. What is the Occam's Razor (also spelled Ockham's Razor)? What is its relationship with the approach usually adopted in natural computing (Section 1.3)?
4. Name some limitations of the current silicon-based computing paradigm.
5. Name one natural phenomenon, process, or system that is a potential candidate to become a new natural computing technique. Include the biological and theoretical background, and the utility of the approach proposed.

1.8 REFERENCES

[1] Aarts, E. and Korst, J. (1989), *Simulated Annealing and Boltzman Machines – A Stochastic Approach to Combinatorial Optimization and Neural Computing*, John Wiley & Sons.

[2] Adami C. (1998), *An Introduction to Artificial Life*, Springer-Verlag/Telos.

[3] Adleman, L. M. (1994), "Molecular Computation of Solutions to Combinatorial Problems", *Science*, **226**, November, pp. 1021–1024.

[4] Attwood, T. K. and Parry-Smith, D. J. (1999), *Introduction to Bioinformatics*, Prentice Hall.

[5] Bäck, T., Fogel, D. B. and Michalewicz, Z. (2000), *Evolutionary Computation 1 Basic Algorithms and Operators*, Institute of Physics Publishing (IOP), Bristol and Philadelphia.

[6] Bäck, T., Fogel, D. B. and Michalewicz, Z. (2000), *Evolutionary Computation 2 Advanced Algorithms and Operators*, Institute of Physics Publishing (IOP), Bristol and Philadelphia.

[7] Bahnzaf, W., Nordin, P., Keller, R. E. and Frankone, F. D. (1997), *Genetic Programming: An Introduction*, Morgan Kaufmann.

[8] Baldi, P. and Brunak, S. (2001), *Bioinformatics: The Machine Learning Approach*, 2nd ed., Bradford Books.

[9] Barnsley, M. F. (1988), *Fractals Everywhere*, Academic Press.

[10] Barnsley, M. F. and Demko, S. (1985), "Iterated Function Systems and the Global Construction of Fractals", *Proc. of the Royal Soc. of London*, **A339**, pp. 243-275.

[11] Beni, G. (1988), "The Concept of Cellular Robotic Systems", *Proc. of the IEEE Int. Symp. on Intelligent Control*, pp. 57–62.

[12] Beni, G. and Wang, J. (1989), "Swarm Intelligence", *Proc. of the 7th Annual Meeting of the Robotics Society of Japan*, pp. 425–428.

[13] Bentley, P. J. (2001), *Digital Biology*, Headline.

[14] Beyer, H.-G. (2001), *Theory of Evolution Strategies*, Springer-Verlag.

[15] Bishop, C. M. (1996), *Neural Networks for Pattern Recognition*, Oxford University Press.

[16] Bonabeau E., Dorigo, M. and Theraulaz, T. (1999), *Swarm Intelligence: From Natural to Artificial Systems*, New York: Oxford University Press.

[17] Booker, L. B., Goldberg, D. E. and Holland, J. H. (1989), "Classifier Systems and Genetic Algorithms", *Artificial Intelligence*, **40**, pp. 235–282.

[18] Calude, C. S. and Păun, G. (2001), *Computing with Cells and Atoms: An Introduction to Quantum, DNA, and Membrane Computing*, Taylor & Francis.

[19] Dasgupta, D. (1999), *Artificial Immune Systems and Their Applications*, Springer-Verlag.

[20] Dayan, P. and Abbot, L. F. (2001), *Theoretical Neuroscience: Computational and Mathematical Modeling of Neural Systems*, The MIT Press.

[21] de Castro, L. N. (2005), "Natural Computing", In M. Khosrow-Pour, *Encyclopedia of Information Science and Technology*, Idea Group Inc.

[22] de Castro, L. N. and Timmis, J. I. (2002), *Artificial Immune Systems: A New Computational Intelligence Approach*, Springer-Verlag.

[23] de Castro, L. N. and Von Zuben, F. J. (2004), *Recent Developments in Biologically Inspired Computing*, Idea Group Publishing.

[24] Deutsch, D. (1985), "Quantum Theory, the Church-Turing Principle and the Universal Quantum Computer", *Proc. of the Royal Soc. of London*, **A 4000**, pp. 97–117.

[25] Fausett, L. (1994), *Fundamentals of Neural Networks: Architectures, Algorithms, and Applications*, Prentice Hall.

[26] Feynman, R. P. (1982), "Simulating Physics with Computers", *Int. Journal of Theor. Physics*, **21**(6/7), pp. 467–488.

[27] Flake, G. W. (2000), *The Computational Beauty of Nature*, MIT Press.

[28] Fogel, D. B. (1998), *Evolutionary Computation: Toward a New Philosophy of Machine Intelligence*, IEEE Press.

[29] Fogel, L. J. (1999), *Intelligence through Simulated Evolution: Forty Years of Evolutionary Programming*, Wiley-Interscience.

[30] Fogel, L. J., Owens, A. J. and Walsh, M. J. (1966), *Artificial Intelligence through Simulated Evolution*, Wiley, New York.

[31] Fournier, A., Fussell, D. and Carpenter, L. (1982), "Computer Rendering of Stochastic Models", *Comm. of the ACM*, **25**, pp. 371–384.

[32] Glover, F. W. and Kochenberger, G. A., (2003), *Handbook of Metaheuristics*, Springer.

[33] Goldberg, D. E. (1989), *Genetic Algorithms in Search, Optimization, and Machine Learning*, Addison-Wesley Pub Co.

[34] Gramß, T., Bornholdt, S., Groß, M., Mitchell, M. and Pellizzari, T. (2001), *Non-Standard Computation*, Wiley-VCH.

[35] Haykin, S. (1999), *Neural Networks: A Comprehensive Foundation*, 2nd ed., Prentice Hall.

[36] Hirvensalo, M. (2000), *Quantum Computing*, Springer-Verlag.

[37] Holland, J. H. (1975), *Adaptation in Natural and Artificial Systems*, MIT Press.

[38] Holmes, J. H., Lanzi, P. L., Stolzmann, W. and Wilson, S. W. (2002). "Learning Classifier Systems: New Models, Successful Applications", *Information Processing Letters*, **82**(1), pp. 23–30.

[39] Hutchinson, J. (1981), "Fractals and Self-Similarity", *Indiana Journal of Mathematics*, **30**, pp. 713–747.

[40] Ilachinski, A. (2001), *Cellular Automata: A Discrete Universe*, World Scientific.

[41] Kennedy, J., Eberhart, R. and Shi. Y. (2001), *Swarm Intelligence*, Morgan Kaufmann Publishers.

[42] Kirkpatrick, S., Gerlatt, C. D. Jr. and Vecchi, M. P. (1983), "Optimization by Simulated Annealing", *Science*, **220**, 671–680.

[43] Kohonen, T. (2000), *Self-Organizing Maps*, Springer-Verlag.

[44] Komosinski, M. and Ulatowski, S. (1999), "Framsticks: Towards a Simulation of a Nature-Like World, Creatures and Evolution", In D. Floreano and F. Mondada, *Lecture Notes in Artificial Intelligence 1674* (Proc. of the 5th European Conf. on Artificial Life), pp. 261–265.

[45] Koza, J. R. (1992), *Genetic Programming: On the Programming of Computers by Means of Natural Selection*, MIT Press.
[46] Koza, J. R. (1994), *Genetic Programming II: Automatic Discovery of Reusable Programs*, MIT Press.
[47] Kumar, S. and Bentley, P. J. (2003), *On Growth, Form and Computers*, Academic Press.
[48] Langton, C. (1988), "Artificial Life", In C. Langton (ed.), *Artificial Life*, Addison-Wesley, pp. 1-47.
[49] Lesmoir-Gordon, N., Rood, W. and Edney, R. (2000), *Introducing Fractal Geometry*, ICON Books UK.
[50] Levy, S. (1992), *Artificial Life*, Vintage Books.
[51] Lindenmayer, A. (1968), "Mathematical Models for Cellular Interaction in Development, Parts I and II", *Journal of Theoretical Biology*, **18**, pp. 280–315.
[52] Mandelbrot, B. (1983), *The Fractal Geometry of Nature*, W. H. Freeman and Company.
[53] Mange, D. and Tomassini, M. (1998), *Bio-Inspired Computing Machines: Towards Novel Computational Architecture*, Presses Polytechniques et Universitaires Romandes.
[54] McCulloch, W. and Pitts, W. H. (1943), "A Logical Calculus of the Ideas Immanent in Nervous Activity", *Bulletin of Mathematical Biophysics*, **5**, pp. 115–133.
[55] Michalewicz, Z. (1996), *Genetic Algorithms + Data Structures = Evolution Programs*, Springer-Verlag, 3rd Ed.
[56] Mitchell, M. (1998), *An Introduction to Genetic Algorithms*, The MIT Press.
[57] Moore, G. E. (1965), "Cramming More Components into Integrated Circuits", *Electronics*, **38**(8).
[58] Nielsen, M. A. and Chuang, I. L. (2000), *Quantum Computation and Quantum Information*, Cambridge University Press.
[59] O'Reilly, R. C. and Munakata, Y. (2000), *Computational Explorations in Cognitive Neuroscience: Understanding the Mind by Simulating the Brain*, The MIT Press.
[60] Paton, R. (1992), "Towards a Metaphorical Biology", *Biology and Philosophy*, **7**, pp. 279–294.
[61] Paton, R. (Ed.) (1994), *Computing with Biological Metaphors*, Chapman & Hall.
[62] Paton, R., Bolouri, H. and Holcombe, M. (2004), *Computing in Cells and Tissues: Perspectives and Tools of Thought*, Springer-Verlag.
[63] Păun, G. and Cutkosky, S. D. (2002), *Membrane Computing*, Springer-Verlag.
[64] Păun, G., Rozenberg, G. and Saloma, A. (1998), *DNA Computing*, Springer-Verlag.
[65] Peitgen, H.-O, Jürgens, H. and Saupe, D. (1992), *Chaos and Fractals: New Frontiers of Science*, Springer-Verlag.
[66] Pittenger, A. O. (2000), *An Introduction to Quantum Computing Algorithms*, Birkhäuser.
[67] Prusinkiewicz, P. and Lindenmayer, A. (1990), *The Algorithmic Beauty of Plants*, Springer-Verlag.
[68] Ray, T. S. (1994), "An Evolutionary Approach to Synthetic Biology", *Artificial Life*, **1**(1/2), pp. 179–209.
[69] Rechenberg, I. (1973), *Evolutionsstrategie: Optimierung Technischer Systeme Nach Prinzipien der Biologischen Evolution*, Frommann-Holzboog, Stuttgart.
[70] Reeves, W. T. (1983), "Particle Systems – A Technique for Modeling a Class of Fuzzy Objects", *ACM Transactions on Graphics*, **2**(2), pp. 91–108.

[71] Rennard, J.-P. (2004), "Perspectives for Strong Artificial Life", In L. N. de Castro and F. J. Von Zuben (Eds.), *Recent Developments in Biologically Inspired Computing, Idea Group Publishing*, Chapter 12, pp. 301–318.

[72] Resnick, M. (1994), *Turtles, Termites, and Traffic Jams: Explorations in Massively Parallel Microworlds*, MIT Press.

[73] Reynolds, C. W. (1987), "Flocks, Herds, and Schools: A Distributed Behavioral Model", *Computer Graphics*, **21**(4), pp. 25–34.

[74] Reynolds, R. G. (1994), "An Introduction to Cultural Algorithms", In A. V. Sebald and L. J. Fogel (Eds.), *Proceedings of the Third Annual Conference on Evolutionary Programming*, World Scientific, River Edge, New Jersey, pp. 131–139.

[75] Schwefel, H. –P. (1965), *Kybernetische Evolutionals Strategie der Experimentellen Forschung in der Stromungstechnik*, Diploma Thesis, Technical University of Berlin.

[76] Schwefel, H. –P. (1995), *Evolution and Optimum Seeking*, John Wiley & Sons.

[77] Shor, P. W. (1994), "Algorithms for Quantum Computation: Discrete Logarithms and Factoring", *Proc. of the 35th Annual Symposium on Foundations of Computer Science*, Santa Fe, NM, IEEE Computer Society Press, pp. 124–134.

[78] Sienko, T., Adamatzky, A. and Rambidi, N. (2003), *Molecular Computing*, MIT Press.

[79] Sober, E. (1996), "Learning from Functionalism – Prospects for Strong Artificial Life". In M. A. Boden (Ed.), *The Philosophy of Artificial Life*, Oxford: Oxford University Press, pp. 361–378.

[80] Spafford, E. H. (1991), "Computer Viruses – A Form of Artificial Life?", in C. G. Langton, C. Taylor, J. D. Farmer (eds.), *Artificial Life II*, Addison-Wesley, pp. 727–746.

[81] Timmis, J., Bentley, P. J. and Hart, E. (Eds.) (2003), *Artificial Immune Systems*, Proc. of the 2nd International Conference on Artificial Immune Systems (ICARIS 2003), Springer-Verlag.

[82] Trappenberg, T. (2002), *Fundamentals of Computational Neuroscience*, Oxford University Press.

[83] Voss, R. F. (1985), "Random Fractals Forgeries", In R. A. Earnshaw (ed.), Fundamental Algorithms for Computer Graphics, Springer-Verlag: Berlin, pp. 805–835.

[84] Waterman, M. S. (1995), *Introduction to Computational Biology: Maps, Sequences, and Genomes*, Chapman & Hall.

[85] Wolfram, S. (1994), *Cellular Automata and Complexity*, Perseus Books.

[86] Zurada, J. M., Robinson, C. J. and Marks, R. J. (1994), *Computational Intelligence: Imitating Life*, IEEE Press.

CHAPTER 2

CONCEPTUALIZATION

"Adaptationism, the paradigm that views organisms as complex adaptive machines whose parts have adaptive functions subsidiary to the fitness-promoting function of the whole, is today about as basic to biology as the atomic theory is to chemistry. And about as controversial."
(D. Dennett, Darwin's Dangerous Idea: Evolution and the Meanings of Life, Penguin Books, 1995, p. 249)

"Nothing is wrong with good metaphors so long as we don't take them as reality."
(C. Emmeche, 1997, Aspects of complexity in life and science, Philosophica, 59(1), pp. 41-68.)

"The complexity of a model is not required to exceed the needs required of it."
(B. S. Silver, The Ascent of Science, 1998, Oxford University Press, p. 79)

2.1 INTRODUCTION

Perhaps the most remarkable characteristic of natural computing is the encompassing of a large variety of disciplines and fields of research. Ideas, principles, concepts, and theoretical models from biology, physics, and chemistry are most often required for a good understanding and development of natural computing. This interdisciplinarity and multidisciplinarity also involves the use of concepts that aid in the description and understanding of the underlying phenomenology. However, most of these concepts are made up of 'slippery' words; words whose meaning can be different under different domains and words whose definitions are not available or agreed upon yet.

Therefore, this chapter presents, in a descriptive manner, what is meant by words such as adaptation, complexity, agents, self-organization, emergence, and fractals. Very few closed definitions will be provided. Instead, descriptive explanations sufficient for the conceptualization and comprehension of the basic ideas incorporated in natural computing will be given. Some popular science books will be cited because they introduce important topics using a terminology with little formalism and mathematics. Some more technical literature may eventually be cited, but most of it can be readily reached with a quick web search.

For those readers anxious to put their hands at work with some computational and algorithmic description of natural computing, this chapter can be temporarily skipped. However, readers are strongly encouraged to go through this chapter thoroughly, because it will give him/her better philosophical basis and insights into concepts that will be widely used throughout the book. This chapter will also provide a better understanding of what is meant by and what is behind of many of the phenomena, processes, and systems studied in natural computing. Finally, this chapter makes use of the many concepts reviewed to start our discussion of nature, which led to the development of natural computing. Several natural systems and processes are used to illustrate the meaning of the concepts investigated here. Most of these examples from nature will be returned to in later chapters.

2.1.1. Natural Phenomena, Models, and Metaphors

All the approaches discussed in this volume are rooted on and sometimes enhanced by their natural plausibility and inspiration. The focus is on how nature has offered inspiration and motivation for their development. However, all these approaches are very appealing to us mainly for computational reasons, and they also may help us study and understand the world we inhabit, and even to create new worlds, new forms of life, and new computing paradigms. They hold out the hope of offering computationally sufficient accurate mechanistic accounts of the natural phenomena they model, mimic, or study, almost always with a view of computing, understanding, and problem solving. They have also radically altered the way we think of and see nature; the computational beauty and usefulness of nature.

Modeling is an integral part of many scientific disciplines and lies behind great human achievements and developments. Most often, the more complex a system, the more simplifications are embodied in its models. The term *model* can be found in many different contexts and disciplines meaning a variety of different things. Trappenberg (2002) has defined “models [as] abstractions of real world systems or implementations of a hypothesis in order to investigate particular questions or to demonstrate particular features of a system or a hypothesis.” (Trappenberg, 2002; p. 7) It corresponds to a (schematic) description of a system, theory, or phenomenon, which accounts for its known or inferred properties and that may be used for further study of its characteristics. Put in a different form, models can be used to represent some aspect of the world, some aspect of theories about the world, or both simultaneously. The representative usefulness of a model lies in its ability to teach us something about the phenomenon it represents. They mediate between the real world and the theories and suppositions about that world (Peck, 2004).

The critical steps in constructing a model are the selection of salient features and laws governing the behavior of the phenomena under investigation. These steps are guided by *metaphor* and knowledge transfer (Holland, 1998). However, for purely practical reasons, many details are usually discarded. In the particular case of natural computing, models are most often simple enough to understand, but rich enough to provide (*emergent*) behaviors that are surprising, interesting, useful, and significant. If all goes well, what may commonly be the case, the result may allow for the prediction and even reproduction of behaviors observed in nature, and the achievement of satisfactory performances when a given function is required from the model.

The word *metaphor* comes from the Greek for 'transference'. It corresponds to the use of language that assigns one thing to designate another, in order to characterize the latter in terms of the former. Metaphors have traditionally been viewed as implicit comparisons. According to this view, metaphors of the form *X is a Y* can be understood as *X is like Y*. Although metaphors can suggest a comparison, they are primarily attributive assertions, not merely comparisons (Wilson and Keil, 1997; p. 535-537). For example, to name a computational tool developed with inspiration in the human brain an 'artificial neural network' or a 'neurocomputing device' corresponds to attributing salient properties of the human brain to the artificial neural network or neurocomputing device. The first part of this book, dedicated to computing inspired by nature, is sometimes referred to as *computing with biological metaphors* (Paton, 1994) or *biologically inspired computing* (de Castro and Von Zuben, 2004).

The use of metaphors from nature as a means or inspiration to develop computational tools for problem solving can also be exemplified by the development of 'artificial immune systems' for computer protection against viruses. One might intuitively argue: "if the human immune system is capable of protecting us against viruses and bacteria, why can't I look into its basic functioning and try to extract some of these ideas and mechanisms to engineer a computer immune system?" Actually, this type of metaphor has already been extracted, not only by academic research institutions, but also by leading companies in the computing area, such as IBM (Chapter 6). There is, however, an important difference between a metaphor and a model. While models are more concerned with quantitatively reproducing some phenomena, metaphors are usually high-levels abstractions and inspirations taken from a system or process in order to develop another. Most metaphors are basically concerned with the extraction or reproduction of qualitative features.

A simple formula, a computer simulation, a physical system; all can be models of a given phenomenon or process. What is particularly important, though, is to bear in mind what are the purposes of the model being created. In theoretical biology and experimental studies, models may serve many purposes:

- Through modeling and identification it is possible to provide a deeper and more quantitative description of the system being modeled and its corresponding experimental results.
- Models can aid in the critical analysis of hypotheses and in the understanding of the underlying natural mechanisms.
- Models can assist in the prediction of behaviors and design of experiments.
- Models may be used to simulate and stimulate new and more satisfactory approaches to natural systems, such as the behavior of insect societies and immune systems.
- Models may allow the recovery of information from experimental results.

An *experiment* can be considered as a procedure performed in a controlled environment for the purpose of gathering observations, data, or facts, demonstrating known facts or theories, or testing hypotheses or theories. Most biological experiments are usually made *in vivo*, within a living organism (e.g., rats and mice), or *in vitro*, in an artificial environment outside the living organism (e.g., a test tube).

There is a significant conceptual difference between experiment, simulation, realization, and emulation. In contrast to experiments, *simulations* and *realizations* are different categories of models (Pattee, 1988). Simulations are metaphorical models that 'stand for' something else, and may cover different levels of fidelity or abstraction. They can be performed by physical modeling, by writing a special-purpose computer program, or by using a more general simulation package that is usually still aimed at a particular kind of simulation. They can be used, for instance, to explore theories about how the real-world functions based on a controlled medium (e.g., a computer). As an example, the simulation of a car accident can be performed by specifying the place and conditions in which the car was driven and then using a given medium (e.g., the computer) to run the simulation. Computer simulation is pervasive in natural computing. It has been used to design problem-solving techniques that mimic the behavior of several biological phenomena (Chapter 3 to Chapter 6), it has served to drive synthetic environments and virtual worlds (Chapter 7 and Chapter 8), and it has been used to simulate DNA computers (Chapter 9).

The *realization* of a system or organism corresponds to a literal, material model that implements certain functions of the original; it is a substantive functional device. Roughly speaking, a realization is evaluated primarily by how well it can function as an implementation of a design specification, and not in relation to the goodness of the measurements (mappings) they perform. A system or function is used to realize another when one performs in exactly the same way as another (Pattee, 1988; Mehler, 2003). To *emulate* a system is to imitate or reproduce its functions using another system or medium. The emulating system has to perform the same functions of the emulated system in the way the latter does. A typical example in computer science is the emulation of one computer by (a program running on) another computer. You may emulate a system as a replacement for the system, whereas you may simulate a system if the goal is, for instance, simply to analyze or study it.

Natural computing approaches are aimed at simulating, emulating, and sometimes realizing natural phenomena, organisms, and processes with distinct goals. The metaphorical representation of simulations makes them suitable for designing problem solving techniques and mimics of nature. Realizations of nature, on the contrary, would be the primary target of the so-called strong artificial life (Chapter 8). It is also important to acknowledge that, as most natural computing approaches to be studied here usually have not the same goals as models, they have the advantages of being explicit about the assumptions and relevant processes incorporated, allowing for a closer control of the variables involved, and providing frameworks to explain a wide range of phenomena.

Due mainly to these differences in goals and levels of details incorporated, most of the highly simplified models discussed in this volume are usually treated as metaphors, simulations, or simple abstractions of natural phenomena or processes. In addition, natural computing techniques are usually based upon a different modeling approach. The theoretical models used in biological sciences are based, in most cases, on ordinary differential equations (ODE) or Monte Carlo simulations. For example, when theoretical biologists want to create a model of an army ant, they use some rule of thumb such as "the more pheromone (a chemical released by ants) an ant detects, the faster it runs". This rule would be translated into an equation of the type $dx/dt = kP$, where dx/dt is the speed (distance change, dx, divided by time change, dt) of the ant, k is a constant of proportionality, and P is the pheromone level. This simple formula captures the essence of ant movement as described.

Despite the differences in approach, level of details, and accuracy, it is undeniable, and this will become clearer throughout the text, that the inspiration from nature and the relationship with it is the core of natural computing. Metaphors are important approaches not only for the creation of useful and interesting tools, but they may also aid the design of more accurate models and a better understanding of nature. Thus, it is not surprising that many researchers in natural computing call their products models instead of metaphors.

2.1.2. From Nature to Computing and Back Again

In most cases, the first step toward developing a natural computing system is to look at nature or theoretical models of natural phenomena in order to have some insights into how nature is, works, and how it behaves. In other cases, it might happen that you have a given problem at hand, and you know some sort of natural system solves a similar problem. A good example is the immune system metaphor for computer security mentioned above. Another classical example is the neural network metaphor: if there are brains that allow us to reason, think, process visual information, memorize, etc., why can I not look into this system and try to find its basic functioning mechanisms in order to develop an (intelligent) 'artificial brain'?

The problem with the extraction of metaphors and inspiration from nature is that it is usually very difficult to understand how nature works. In the particular case of the brain, though some basic signal transmission processes might be already known (and many other facts as well), it is still out of human reach to fully uncover its mysteries, mainly some cognitive abilities such as hate and love. The use of technological means (e.g., computers) to simulate, emulate or reproduce natural phenomena may also not be the most suitable approach. Would computers, such as the ones we have nowadays, be suitable to build an 'artificial brain' or an 'artificial organism'? Can we simulate 'wetware' with the current 'hardware'? Furthermore, even if we do know how some natural processes work, would it still be suitable to simply reproduce them the way they are? For example, we know that most birds are capable of flying by flapping wings, however airplanes fly using propellers or turbines. Why do airplanes not fly by flapping wings?

Last, but not least, sometimes looking at nature or theoretical studies may not be sufficient to give us the necessary insight into what could be done in computing and engineering with these phenomena. We have already seen, in Chapter 1, that the clustering of dead bodies in ants may result in computer algorithms for solving clustering problems, and that simple behavioral rules applied to many virtual birds result in flock-like group behaviors. What if I tell you that the behavior of ants foraging for food resulted in powerful algorithms for solving combinatorial optimization problems? Also, what if I tell you that the behavior of ant prey retrieval has led to approaches for collective robotics? Can you have an idea of how these are accomplished without looking at the answers in Chapter 5?

Due to all these aspects, designing novel natural computing systems may not be a straightforward process. But this book is not about how to design new natural computing devices, though some insights about it will certainly be gained. Instead, it focuses on how the main natural computing systems available nowadays were motivated, emerged, and can be understood and designed. Designing natural computing systems is basically an engineering task; that is, physical, mechanical, structural, and behavioral properties of nature are made useful to us in computational terms. They can become new problem-solving techniques, new forms of (studying) nature, or new forms of computing. Each part of natural computing, and its many branches, is rooted in some specific feature(s):

- Evolutionary algorithms were inspired by evolutionary biology.
- Artificial neural networks were inspired by the functioning of the nervous system.
- Swarm systems are based on social organisms (from insects to humans).
- Artificial immune systems extract ideas from the vertebrate immune system.
- Fractal geometry creates life-like patterns using systems of interactive functions, L-systems, and many other techniques.
- Artificial life is based on the study of life on Earth to simulate life on computers and sometimes develop synthetic forms of life.
- DNA computing is based on the mechanisms used to process DNA strands in order to provide a new computing paradigm.
- Quantum computing is rooted on quantum physics to develop another new computing paradigm.

Although it is generally difficult to provide a single engineering framework to natural computing, some of its many branches allow the specification of major structures and common design procedures that can be used as frameworks to the design of specific natural computing techniques. For instance, evolutionary algorithms can be designed by specifying a representation for candidate solutions to a problem, some general-purpose operators that manipulate the candidate solutions, and an evaluation function that quantifies the goodness or quality of each candidate solution (Chapter 3). In artificial life, however, it is much harder to provide such a framework. It will be seen that most artificial life (ALife) approaches reviewed here are based on the specification of usually simple sets of rules describing the behavior of individual agents. The remaining of the ALife project will involve the modeling of the agents, environment, etc., which are not part of the scope of this book.

2.2 GENERAL CONCEPTS

2.2.1. Individuals, Entities, and Agents

There is a body of literature about *agents* and agent-based systems. One of the main themes of this book is collectivity; populations of individuals, insect societies, flocks of bird, schools of fish, herds of land animals, repertoires of immune cells and molecules, networks of neurons, and DNA strands. What all these systems have in common is the presence of a number of individual *entities* or *components*. When we model or study these systems, the *individuals* may go by the generic name of *agents*. However, the words individuals, entities, components, and agents are sometimes used interchangeably and with no distinction throughout the text.

The term agent is currently used to mean anything between a mere subroutine of a computer program and an intelligent organism, such as a human being. Intuitively, for something to be considered an agent, it must present some degree of autonomy or identity; that is, it must, in some sense, be distinguishable from its environment by some kind of spatial, temporal, or functional boundary. Traditionally, agent-based models are drawn on examples of biological phenomena and processes, such as social insects and immune systems (Rocha, 1999). These systems are formed by distributed collections of interacting elements (agents) that work under no central control. From simple agents, who interact locally following simple rules of behavior and responding to environmental stimuli, it is possible to observe a synergistic behavior that leads to higher-level behaviors that are much more intricate than those of individuals.

Agent-based research has a variety of definitions of what is an agent, each hoping to explain one particular use of the word. These definitions range from the simplest to the lengthiest ones. Here are some examples:

> "An agent is anything that can be viewed as perceiving its environment through sensors and acting upon that environment through effectors." (Russell and Norvig, 1995)

> "Perhaps the most general way in which the term agent is used is to denote a hardware or (more usually) software-based computer system that enjoys the following properties: autonomy, social ability, reactivity, and proactiveness." (Wooldridge and Jennings, 1995) [Summarized definition, for the full version please consult the cited reference]

> "An autonomous agent is a system situated within and part of an environment that senses that environment and acts on it, over time, in pursuit of its own agenda and also so as to effect what it senses in the future." (Franklin and Graesser, 1997)

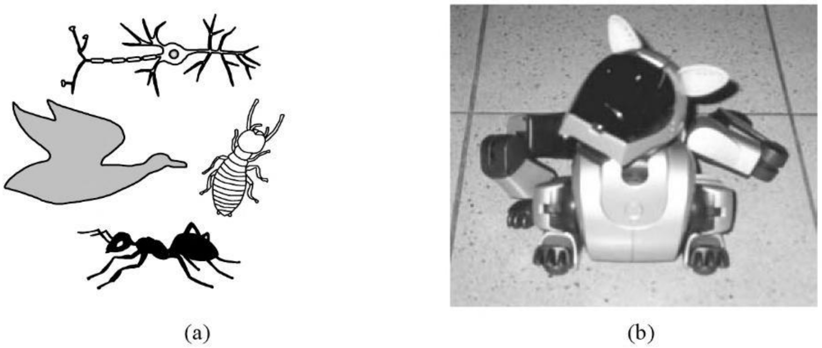


Figure 2.1: Examples of agents. (a) Pictorial representation of biological agents (bird, neuron, termite, and ant). (b) Physical agent (the AIBO ERS-210 robot by Sony®).

Therefore, an agent can be understood as an entity endowed with a (partial) representation of the environment, capable of acting upon itself and the environment, and also capable of communicating with other agents. Its behavior is a consequence of its observations, knowledge, and its interactions with other agents and the environment. Agents can be of many types, including biological (e.g., ants, termites, neurons, immune cells, birds, etc.), physical (e.g., robots), and virtual agents (e.g., a computer algorithm, Tamagotchi, etc.), as illustrated in Figure 2.1.

2.2.2. Parallelism and Distributivity

There are several well-known examples involving the capability of processing more than one thing at the same time. In the natural world, *parallel processing* is evident in insect societies, brain processing, immune functioning, the evolution of species, and so forth. All these examples will be studied in this book.

In order for evolution to occur, there must be a number of individuals in a population competing for limited resources. These individuals suffer genetic variation and those more fit (adapted) to the environment have higher probabilities of survival and reproduction. All the individuals in the population play important roles in exploring the environment and sometimes exchanging (genetic) information, thus producing progenies more adapted to the life in a spatial location.

In insect societies, in particular in ant colonies, a colony of ants has individuals assigned to various tasks, such as harvesting food, cleaning the nest, and caring for the queen. Termites, bees, and wasps also perform similar tasks in a distributed way; there are individuals allocated for different tasks. All insects in a colony work in parallel in a given task, but they may switch tasks when needed. For instance, some worker ants may be recruited for battle when the nest is being invaded.

In immune systems, a large variety and number of cells are involved in an immune response. When a virus infects a cell, some specialized immune cells, named T-cells, recognize fragments of this virus presented by a molecular complex of another specialized antigen presenting cell. This recognition triggers the action of many other immune cells to the site of infection. In addition, several other cells are performing the same and other processes, all at once in a distributed and parallel form.

In the human nervous system, a huge number of neurons are involved in processing information at each time instant. Talking while driving, watching TV while studying, hearing a name while having a conversation with someone in the middle of a party (a phenomenon called the 'cocktail party effect'), all these are just samples of a kind of parallel processing. Who does not know about the joke of not being able to walk while chewing gum? The vast number of neurons we have endow us with this capability of processing multiple information from multiple sensors at the same time.

What is surprising about each of the individual processes from the examples above is that they are all a product of a large number of elements and processes occurring in parallel. At the lowest level of analysis, evolution requires a large number of individuals to allow for a genetic variety and diversity that ultimately result in a higher adaptability; insect colonies are composed of thousands, sometimes millions, of insects that work in concert to maintain life in the colony; immune systems are composed of approximately 10^{12} lymphocytes (a special type of immune cell); and the human brain contains around 10^{11} nervous cells. Each of these individual agents contributes its little bit to the overall global effect of evolution, maintenance of life in the colony (insects), and the body (immune systems), and thought processes and cognition (nervous system).

From a biological and computational perspective, all the end results discussed are going to be emergent properties of the parallel and distributed operations of individual entities. All these systems can be termed *parallel-distributed systems* (PDS). Rumelhart and collaborators (Rumelhart et al., 1986; McClelland et al., 1986) have coined the term *parallel distributed processing* (PDP) to describe parallel-distributed systems composed of processing elements, in particular neurons. They used this terminology to refer to highly abstract models of neural function, currently known as *artificial neural networks* (ANN). These will be discussed in more detail in Chapter 4 under the heading of Neurocomputing. PDP networks are thus a particular case of parallel-distributed systems.

2.2.3. Interactivity

A remarkable feature of natural systems is that individual agents are capable of interacting with one another or the environment. Individual organisms interact with one another in variety of forms: reproductively, symbiotically, competitively, in a predator-prey situation, parasitically, via channels of communication, and so on. At a macro level, an important outcome of these interactions is a struggle for limited resources and life. Individuals more adapted to the (local) environment tend to survive and mate thus producing more progenies and propagating their genetic material. Genetic variation together with the selection of the fittest individuals leads to the creation of increasingly fitter species. Besides, interactivity allows for the emergence of self-organized patterns.

Interactivity is an important mean nature has to generate and maintain life. Complex systems, organisms, and behaviors emerge from interacting components. For instance, take the case of genes, known to be the basic functional elements of life. Researchers have created genetically modified organisms in which a single gene has been deleted or blocked, a process known as *knockout*. In some situations, these researchers have been surprised to find that some other gene(s) can take over its whole function or at least part of it. Similar cases are constantly being reported in the news where people with damaged brains, from accidents for example, are capable of recovering some of their lost functions after often long periods of treatment and recovery. It is observed, in most of these cases, that other portions of the brain assume the functions previously performed by the damaged areas. Interactions, thus, are not only necessary for the complexity, diversity, and maintenance of life, but it also leads to emergent phenomena and behaviors that cannot be predicted by simply looking at discrete components.

In all the main systems studied in this book, several types of interactions can be observed. For instance, immune cells and molecules communicate with one another and foreign agents through chemical messengers and physical contact; insects may also communicate with one another via chemical cues, dancing (e.g., bees dance to indicate where there is food to the other bees in the nest) or physical contact (e.g., antennation); and neurons are known to be connected with one another via small portions of its axons known as synapses. All these communication and contact means allow for the interaction of individual agents in the many systems. The interactions between individuals can be basically of two types: direct and indirect. One important example of direct interaction, namely *connectivity*, and one important example of indirect interaction, namely *stigmergy*, will be discussed in the next two sections. Other important examples of direct interaction are reproduction and molecular signaling, and these will be specifically discussed in the next few chapters.

Connectivity

Connectionist systems employ a type of representation whereby information is encoded throughout the nodes and connections of a network of basic elements, also called units. Their content and representational function is often revealed only through the analysis of the activity patterns of the internal units of the system. Although the term *connectionism* appeared in the mid 1980s to denote network models of cognition based on the spreading activation of numerous simple units (cf. Rumelhart et al., 1986; McClelland et al., 1986), it can refer to any approach based on interconnected elements. These systems are sometimes referred to as *networks*.

The peculiarity of connectionist systems is due to several factors. The connections establish specific pathways of interaction between units; two units can only interact if they have a connection linking them. The connection is also in most cases an active element of interaction, i.e., it not only specifies who interacts with whom, but it also quantifies the degree of this interaction by weighting the signal being transmitted. The direct interaction via connections also results in a structured pattern for the system that may, for instance, reflect the structural organization of the environment in which the network is embedded. Networks are also very successful examples of parallel-distributed processors, for instance, neural networks and immune networks. These two types of networks will be fully explored in this book in Chapter 4 and Chapter 6, respectively.

Stigmergy

Grassé (1959) introduced the concept of *stigmergy* as a means to refer to how the members of a termite colony of the genus *Macrotermes* coordinate nest building. He realized how individual termites could act independently on a structure without direct communication or interactions. This process was termed *indirect social interactions* to describe the same mechanism of indirect communication among bees in a bee colony (Michener, 1974).

The concept of stigmergy provides a general mechanism that relates individual and colony-level behaviors: individual behaviors modify the environment, which in turn modifies the behavior of other individuals. The environment thus mediates the communication of individuals, i.e., there is an indirect communication, instead of direct, by means such as antennation, trophalaxis (food or liquid exchange), mandibular contact, visual contact, and so on (Bonabeau et al., 1999). Self-organization is thus made possible due to the intensity of the stigmergic interactions among termites that can adopt a continuum of interactions.

Grassé (1959) gave the original example to illustrate stigmergy involving nest building in termite colonies (Figure 2.2). He observed that termite workers are stimulated to act during nest building according to the configuration of the construction and of other workers. Termite workers use soil pellets, which they impregnate with a chemical substance known as *pheromone*, to build pillars. Initially, termites deposit pellets in a random fashion until one of the deposits reaches a critical size. Then, if the group of builders is large enough and the pillars start to emerge, a coordination phase begins. The accumulation of pellets reinforces the attractivity of deposits due to the diffusing pheromone emitted by the pellets. Therefore, the presence of an initial deposit of soil pellets stimulates workers to accumulate more pellets through a *positive feedback* or self-reinforcing mechanism (Dorigo et al., 2000).

It is possible to extend the idea of stigmergy to other domains (Holland and Melhuish, 1999). It can be seen as an even more impressive and general account of how the interaction of simple entities, such as ants or termites, can produce a wide range of highly organized and coordinated behaviors and behavioral outcomes, simply acting and exploiting the influence of the environment. By exploiting the stigmergic approach to coordination, researchers have been able to design a number of successful algorithms and systems that can be applied to several domains, such as discrete optimization, clustering, and robotics.

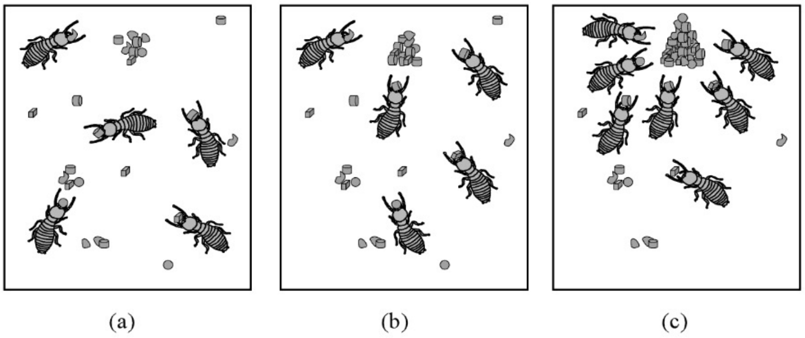


Figure 2.2: Termite mound building. (a) Pellets are initially deposited randomly in space. If the group of builders is large enough and pillars start to emerge (b), then a coordinated building phase starts (c).

Chapter 5 reviews some of these applications focusing on those systems inspired by the behavior of ants. Chapter 8 also provides some examples of stigmergic interactions such as the wasp nest building behavior.

2.2.4. Adaptation

Adaptation can be defined as the ability of a system to adjust its response to stimuli depending upon the environment. Something, such as an organism, a device, or a mechanism, that is changed (or changes) so as to become more suitable to a new or a special application or situation, becomes more adapted to the new application or situation. The use of the word adaptation is, in many cases, related with *evolution* (cf. Wilson and Keil, 1999; p. 3–4). However, many other important concepts in natural computing, such as learning and self-organization, can also be viewed as types of, or resulting from, adaptation mechanisms.

Learning

Learning may be viewed as corresponding to the act, process, or experience of gaining knowledge, comprehension, skill, or mastery, through experience, study, or interactions. Learning systems are those able to change their behavior based on examples in order to solve information-processing demands. An important virtue of adaptation in learning is the possibility of solving information processing tasks and the ability to cope with changing (dynamic) environments.

A consideration of what it takes to learn reveals an important dependence on gradedness (the passing through successive stages of changes) and other aspects of natural mechanisms (O'Reilly and Munakata, 2000). Learning, or more generally adapting, can be viewed as a synonym for *changing* with the end result of knowledge (memory) acquisition. When a system learns it changes its pattern of behavior (or another specific feature), such as the way information is processed.

It is much easier to learn if the system responds to these changes in a graded, proportional manner, instead of radically altering the way it behaves.

These graded changes allow the system to try out a number of different patterns of behavior, and get some kind of graded proportional indication of how these changes are affecting the system's interaction with the environment. By exploring several little changes, the system can evaluate and strengthen those that improve performance, while abandoning or weakening those that do not. There are, however, other types of learning procedures in nature that are more discrete than the graded one just described. For instance, it is believed that there are some specialized areas in the brain particularly good at 'memorizing' discrete facts or events.

In contrast to some beliefs, learning does not depend purely on consciousness and also does not require a brain. Insect societies learn how to forage for food, and our immune systems learn how to fight against disease-causing agents - a principle explored in the vaccination procedures. Even evolution can be viewed as resulting in learning, though evolutionary systems are more appropriately characterized as adaptive systems in the context of natural computing.

Neurocomputing models, and others based on computational neuroscience, provide useful accounts of many forms of learning, such as graded learning and memorization. Chapter 4 reviews some of the standard and most widely spread neurocomputing techniques, and provides a discussion about the main learning paradigms in this field, namely *supervised*, *unsupervised*, and *reinforcement learning*.

Evolution

In its simplest form, the *theory of evolution* is just the idea that life has changed over time, with younger forms descending from older ones. This idea existed well before the time of Charles Darwin, but he and his successors developed it to explain both the diversity of life and the adaptation of living things to their environment (Wilson and Keil, 1999; p. 290–292).

In contrast to learning, evolution requires some specific processes to occur. First, evolution involves an individual or a population of individuals that reproduce and suffer genetic variation followed by natural selection. Without any one of these characteristics, there is no evolution. Therefore, there cannot be evolution if there is a single individual, unless this individual is capable of asexually reproducing. Also, some variation has to occur during reproduction so that the progeny brings some 'novelty' that allows it to become more adapted to the environment. Finally, natural selection is responsible for the maintenance of the genetic material responsible for the fittest individuals to the environment; these will have survival and reproductive advantages over the others, less fit individuals. The outcome of evolution, like the outcome of learning, is a better adaptability to life and the environment.

Both, the evolved genetic configuration of organisms together with their learning capabilities make important contributions to their adaptability to the environment. But perhaps only in the context of learning the genetic encoding can be fully understood, much as the role of DNA itself in shaping the phenotype must be understood in the context of emergent developmental processes.

2.2.5. Feedback

Essentially, *feedback* occurs when the response to a stimulus has an effect of some kind on the original stimulus. It can be understood as the return of a portion of the output of a process or system to the input, especially when used to maintain performance or to control a system or process. The nature of the response determines how the feedback is labeled: *negative feedback* is when the response diminishes the original stimulus (they go in the opposite direction); and *positive feedback* is when the response enhances the original stimulus (they go in the same direction). An important feature of most natural systems described in this text is that they rely extensively on *feedback*, both for growth and self-regulation.

Take the case of the human brain as an example of extensive feedback loops and their importance. The brain can be viewed as a massive network of neurons interconnected via tiny gaps known as synapses. Any brain activity, such as thinking of a word or recognizing a face, triggers a vast array of neural circuitry. Each new brain activity triggers a new array, and an unimaginably large number of possible neuronal circuits go unrealized during the lifetime of an individual. Beneath all that apparent diversity, certain circuits repeat themselves over and over again. All these feedback and reverberating loops are believed to be necessary for learning, and are consequences of the high interconnectivity of the brain.

Positive Feedback

Positive feedback is a sort of self-reinforcing (growth) process in which the more an event occurs, the more it tends to occur. Take the case of the immune system as an example. When a bacterium invades our organism, it starts reproducing and causing damage to our cells. One way the immune systems find to cope with these reproducing agents is by reproducing the immune cells capable of recognizing these agents. And the more cells are generated, the more cells can be generated. Furthermore, the immune cells and molecules release chemicals that stimulate other immune cells and molecules to fight against the disease-causing agent. Therefore, the response of some immune cells provides some sort of *positive feedback* to other immune cells reproduce and join the pool of cells involved in this immune response.

The termite mound building behavior discussed previously is another example of a positive feedback mechanism. The more soil pellets are deposited in a given portion of the space, the more pellets tend to be deposited in that portion because there is more pheromone attracting the termites (Figure 2.3). But these self-reinforcing (positive feedback) processes have to be regulated by *negative feedback* processes, otherwise the systems would go unstable or the resources would be depleted.

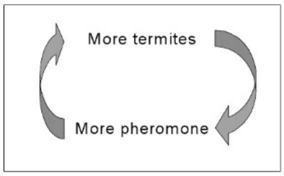


Figure 2.3: Example of positive feedback.

There are several other examples of positive feedback in nature:

- *Human breeding*: the more humans reproduce, the more humans exist to reproduce.
- *Feeding the baby*: a baby begins to suckle her mother's nipple and a few drops of milk are released, stimulating the production and release of more milk.
- *Avalanche*: an avalanche starts at rest and, when disturbed, accelerates quickly towards its end point at the base of a slope.
- *Autocatalysis*: autocatalysis occurs in some digestive enzymes such as pepsin. Pepsin is a protein-digesting enzyme that works in the stomach. However, the stomach does not secrete pepsin; it secretes an inactive form, called pepsinogen. When one pepsinogen molecule becomes activated, it helps to activate other pepsinogens nearby, which in turn can activate others. In this way, the number of active pepsin molecules can increase rapidly by using positive feedback.
- *Giving birth*: while giving birth, the more uterine contractions a mother has, the more it is stimulated to have, until the child is born.
- *Scratching an itch*: scratching an itch makes it more infected and damaged, and thus more itchy.
- *Ripening fruits*: a ripening apple releases the volatile plant hormone ethylene, which accelerates the ripening of unripe fruit in its vicinity; so nearby fruits also ripen, releasing more ethylene. All the fruits become quickly ripe.

Negative Feedback

Negative feedback by contrast, plays the role of regulating positive feedback so as to maintain a(n) (dynamic) equilibrium of the medium. It refers to change in the opposite direction to the original stimulus. The thermostat is one of the most classic examples of negative feedback. It takes the reading of a room's temperature, measures that reading according to a desired setting, and then adjusts its state accordingly. If the room's temperature is too low, more hot air is allowed to flow into the room; else if the temperature is too high, then more cold air flows into the room (Figure 2.4).

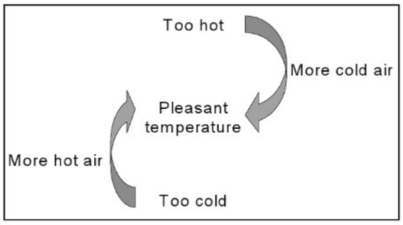


Figure 2.4: Example of negative feedback.

Negative feedback is at the heart of every stable, self-regulating system. If a company raises prices too high, people stop buying, and soon the company cuts the price to increase sales. In the immune system example given above, after the infection is successfully eliminated, specific immune cells are stimulated to release other chemical substances that suppress the replication of immune cells, thus ceasing the immune response. Without this negative feedback mechanism, death by uncontrolled cell reproduction would be inevitable. And without the positive feedback, death from infection would be inevitable.

There are also plenty of examples of negative feedback in nature:

- *Ecosystems*: in an ecosystem composed of, say rabbits and grass, when there is plenty of grass to feed the rabbits, they tend to reproduce with greater rates. But as there are more rabbits in the environment, the more grass will be eaten, and the less grass will be left as food; the amount of grass provides a feedback to the rabbits birth rate.
- *Homeostasis*: blood glucose concentrations rise after eating a meal rich in sugar. The hormone insulin is released and it speeds up the transport of glucose out of the blood and into selected tissues, decreasing blood glucose concentrations.
- *Metabolism*: exercise creates metabolic heat that raises the body temperature. Cooling mechanisms such as vasodilatation (flushed skin) and sweating begin, decreasing the body temperature.
- *Climate theory*: the curvature of the earth helps making it so that continental glaciers expanding equator ward experience strong sunlight and tend to melt. Another example is the tendency for continental glaciers to make cold, high-pressure regions, which do not favor further snowfall.

2.2.6. Self-Organization

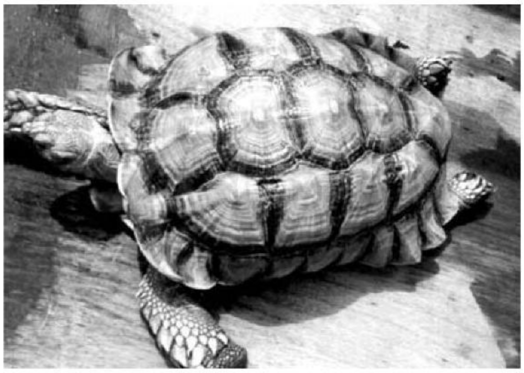
An important question in biology, physics, and chemistry is "Where does order come from?" The world abounds with systems, organisms, and phenomena that maintain a high internal energy and organization in seeming defiance of the laws of physics (Decker, 2000). Water particles suspended in air form clouds; a social insect grows from a single celled zygote into a complex multicellular organism and then participates in a structured social organization; birds gather together in a coordinated flock; and so forth. What is so fascinating is that the organization seems to emerge spontaneously from disordered conditions, and it does not appear to be driven solely by known physical laws or global rules. Somehow, the order arises from the multitude of interactions among the simple parts. Self-organization may also go by the name ‘emergent structuring’, ‘self-assembly’, ‘autocatalysis’, and ‘autopoiesis’, though most of these concepts have some slight differences to the self-organization concept provided here (see Project 3).

Self-organization refers to a broad range of pattern-formation processes in both physical and biological systems, such as sand grains assembling into rippled dunes, chemical reactants forming swirling spirals, cells making up highly structured tissues, and fishes joining together in schools. A basic feature of these diverse systems is the means by which they acquire their order and structure. In self-organizing systems, pattern formation occurs through interactions internal to the system, without intervention by external directing influences. As used here, a *pattern* corresponds to a particular, organized arrangement of objects in space or time. Examples of biological behavioral patterns include a school of fish, a raiding column of army ants, the synchronous flashing of fireflies, and the complex architecture of a termite mound (Camazine et al., 2001). But self-organization does not only affect behavioral patterns, it is also believed to play a role in the definition of patterns, such as shapes, and the coating of several animals (see Figure 2.5). In these cases, it is believed that not only the genetic code of these animals determine their physical expressed characteristics, some self-organized processes may also be involved.

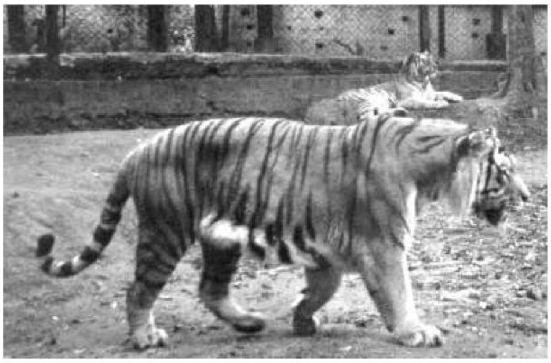
The concept of self-organization can also be conveyed through counterexamples. A system can form a precise pattern receiving instructions from outside, such as a blueprint, recipe, orders, or signals. For instance, the soldiers marching form a neat organized process that is not self-organized. Their sequence of steps, direction of movement, velocity, etc., are all dictated by specific instructions. In such cases, the process is organized but not self-organized. It is less obvious, however, to understand how a definite pattern can be produced in the absence of such instructions.

The self-organized pattern formation in social systems is one of the main themes of this book, and will become clearer and more exemplified along the chapters. It will be seen that many social insects, such as ants, termites, and bees, are capable of building extremely complex nests without following any blueprint, recipe, leader, or template. It is interesting to note that, although all of them have a queen (the queen ant, the queen termite, and the queen bee), queens are basically involved in reproduction, mainly when the colony size is very large.

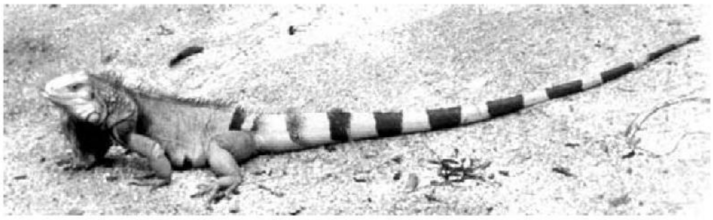
There is even an interesting historical fact about the queen bee. Until the late 19th century, in a time when men were considered superior to women, queen bees were called kings, because people could not accept that anything so well organized could be run by females. Although females could actually run the colony, it is now known that, in most cases, the main role of the queens is to lay eggs. Even more surprising, such complex patterns of behavior, architecture designs, and foraging and hunting strategies, do not require any global control or rule whatsoever; they are amazing self-organized undertakings.



(a)



(b)



(c)

Figure 2.5: Animal patterns believed to involve self-organized pattern formation. (a) Polygonal shapes on the shell of a turtle. (b) Stripes coating the tiger skin. (c) Stripes in the iguana's tail.

Characteristics of Self-Organization

Self-organization refers to spontaneous ordering tendencies sometimes observed in certain classes of complex systems, both natural and artificial. Most self-organized systems present a number of features:

- *Collectivity and interactivity*: self-organizing systems (SOS) are usually composed of a large number of elements that interact with one another and the environment.
- *Dynamics*: the multiplicity of interactions that characterize self-organizing systems emphasize that they are dynamic and require continual interactions of lower-level components to produce and maintain structure.
- *Emergent patterns*: SOS usually exhibit what appears to be spontaneous order; the overall state of a self-organized system is an emergent property.
- *Nonlinearities*: an underlying concept in self-organization is nonlinearity. The interactions of components result in qualitatively new properties that cannot be understood as the simple addition of the individual contributions.
- *Complexity*: most self-organizing systems are complex. The very concepts of complexity and emergence are embodied in SOS. However, it is more accurate to say that complex systems can be self-organizing systems.
- *Rule-based*: most SOS are rule-based, mainly biological self-organizing systems. Examples of rules governing natural self-organized systems were already reviewed, such as the ones that result in dead body clustering in ant colonies. Further examples will be given in the following chapters.
- *Feedback loops*: positive and negative feedback contribute to the formation of self-organized processes by amplifying and regulating fluctuations in the system.

Alternatives to Self-Organization

Self-organization is not the only means responsible for the many patterns we see in nature. Furthermore, even those patterns that arise through self-organization may involve other mechanisms, such as the genetic encoding and physical constraints (laws). Camazine et al. (2001) provide four alternatives to self-organization:

- *Following a leader*: a well-informed leader can direct the activity of the group, providing each group member with detailed instructions about what to do. For example, a captain in a battle field gives order to each soldier relating to where to attack, etc.

- *Building a blueprint*: a blueprint is a compact representation of the spatial or temporal relationships of the parts of a pattern. For instance, each musician of an orchestra receives a musical score that fully specifies the pattern of notes within the composition and the tonal and temporal relationships among them.
- *Following a recipe*: each member of the group may have a recipe, i.e., a sequential set of instructions that precisely specify the spatial and temporal actions of the individual's contribution to the whole pattern. For example, you tell someone how to get to your place by specifying the precise sequence of streets he/she must follow. A blueprint is different from a recipe because it does not specify how something is done, only what is to be done.
- *Templates*: a template is a full-size guide or mold that specifies the final pattern and strongly steers the pattern formation process. For example, a company that makes car parts uses a template in which raw material is poured in order to make the desired parts; each part has its own template.

2.2.7. Complexity, Emergence, and Reductionism

Complexity and emergence are some of the most difficult terms to conceptualize in this chapter. Viewpoints and definitions of complexity (complex systems) and emergence vary among researchers and disciplines. This section discusses only some of the many perspectives; further and more complete studies can be found, for instance, in (Emmeche, 1997; Baas and Emmeche, 1997), the special issue on Complex Systems of the Science magazine (Science, 1999), on the Santa Fe volumes on Artificial Life and Complex Systems (e.g., Cowan et al., 1994; Morowitz and Singer, 1995), and on the Artificial Life and Complex Systems journals (see Appendix C).

Complexity

To start the discussion, let us present a very simplistic idea that fits into the context of natural computing: a *complex system* is a system featuring a large number of interacting components whose aggregate activity is nonlinear (not derivable by summing the behavior of individual components) and typically exhibit self-organization (Morowitz and Singer, 1995; Holland, 1995; Gallagher and Appenzeller, 1999; Rocha, 1999). Consider the case of an organism; say the human body. Can you fully uncover how it works by looking at its major systems and organs? The answer is no. Take an even more reductionist approach, and try to understand the organism by analyzing all its cells and molecules. Can you understand it now? Not still. Some limitations of this more traditional *reductionist* way of thinking will be discussed later. What is important here is the fact that for complex systems we are unable to understand/explain their behavior by examining its component parts alone.

The studies on *complexity* suggest that not only the internal organization (e.g., the genetic code of a biological organism) of a system is sufficient for its full understanding, but also how the system itself and its component parts interact with one another and the environment. The internal microstructure, self-organizing capabilities, and natural selection are part of the most important aspects necessary for a better understanding of complex systems.

Perhaps the most remarkable contribution of complexity to science was the perception that many natural phenomena and processes can be explained and sometimes reproduced by following some basic and simple rules. For instance, the most striking aspects of physics are the simplicity of its laws. Maxwell's equations, Schrödinger's equations, and Hamiltonian mechanics can each be expressed in a few lines. Many ideas that form the foundations of nature are also very simple indeed: nature is lawful, and some basic laws hold everywhere. Nature can produce complex structures even in simple situations and can obey simple laws even in complex situations (Goldenfeld and Kadanoff, 1999).

The five basic forms of investigating complex systems have already been discussed (Section 2.1.1), namely, experimentation, simulation, theoretical modeling, emulation, and realization. Experiments are necessary for raising a range of information about how the natural organisms and processes behave. Simulations are often used to check the understanding, validate experimental results, or simulate a particular system or process. Theoretical models are useful for the understanding, complementation, prediction, critical analysis, and quantitative and qualitative description of natural phenomena. Finally, realizations and emulations are fundamental for the possibility of creating and studying (new) life-like patterns and forms of life.

In order to explore the complexity inherent in nature, one must focus on the right level of description. In natural computing, higher-levels of description are usually adopted. Most systems developed are highly abstract models or metaphors of their biological counterparts. The inclusion of too many processes, details, and parameters, can obscure the desired qualitative understanding and can also make the creation of computational systems based on nature unfeasible. For instance, the 'artificial' neural networks are based upon very simple mathematical models of neural units structured in a network-like architecture and subjected to an iterative procedure of adaptation (learning). Despite all this simplicity, the approaches to be presented still capture some important features borrowed from nature that allow them to perform tasks and solve problems that, in most cases, could not be solved satisfactorily with the previously existing approaches.

In *Hidden Order*, J. Holland (1995) starts with a discussion of how natural (biological and social) systems are formed and self-sustained. Among the several instances discussed, there is the case of the immune system with its numerous cells, molecules and organs. Other examples range from the New York City to the central nervous system. These systems are termed *complex adaptive systems* (CAS), in which the (complex) behavior of the whole is more than a simple sum of individual behaviors. One of the main questions involved in complex adaptive systems is that of how a decentralized system - with no central planning or control - is self-organized.

Despite the differences among all complex adaptive systems and organizations, in most cases, the persistence of the system relies on some main aspects: 1) *interactions*, 2) *diversity*, and 3) *adaptation*. Adaptability allows a system or organism to become better fit to the environment or to learn to accomplish a given task. Adaptability also has to do with the system's capability of processing information (or computing); another important feature of a complex adaptive system. The major task of surviving involves the gathering of information from the environment, its processing and responding accordingly. It is clear, thus, that computing or processing information does not require a brain; ants, immune systems, evolutionary processes, flocks of birds, schools of fish, and many other complex adaptive systems present the natural capability of processing information.

According to Holland, the choice of the name *complex adaptive systems* is more than a terminology "It signals our intuition that general principles rule CAS behavior, principles that point to ways of solving attendant problems." (Holland, 1995; p. 4). This turns back to the idea that there are general rules or principles governing natural systems. The question, thus, can be summarized as how to extract these general principles. This is also one of the main aims of this book and will be illustrated and more clearly identified in all chapters.

Emergence

Important questions about complex adaptive systems rely upon the understanding of *emergent* properties. At a very low level, how do living systems result from the laws of physics and chemistry? That is, how do the genes specify the unfolding processes of biochemical reactions and interactions that result in the development of an organism? Are the genes the necessary and sufficient ingredients to development? At higher levels, how insect societies are organized? How do brains process information? How does the immune system cope with disease-causing agents? Why does a flock of bird present such a coordinated behavior? None of these questions can be answered without having in mind the concept of *emergence*; that is to say, the properties of the whole are not possessed by, nor are they directly derivable from, any of the parts - a water particle is not a cloud, and a neuron is not conscious.

All the systems and processes discussed above present behaviors by drawing upon populations of relatively 'unintelligent' individual agents, rather than a single, 'intelligent' agent. They are *bottom-up* systems, not *top-down*. They are complex adaptive systems that display emergent behaviors. Emergence is a concept tightly linked with complex systems. In these systems, agents residing on one scale or level start producing behaviors that lie scale(s) above them: social insects create colonies; social animals create flocks, herds, and schools; immune cells and molecules compose the immune system; neurons form brains, and so forth. The movement from low-level rules to higher-level sophistication is what we call emergence (Johnson, 2002).

One important feature most emergent systems and processes discussed in this book share is that they are rule-governed. Remember, nature is lawful! It means that it is possible to describe them in terms of a usually simple set of laws or rules that govern the behavior of individual agents. This book presents various instances of small numbers of rules or laws generating (artificial) systems of surprising complexity and potential for problem solving, computing, and the simulation of life-like patterns and behaviors. It will be interesting to note that even with a fixed set of rules, the overall emergent behaviors are *dynamic* (they change over time) and, in most cases, unpredictably.

Holland (1998) underlines a number of features of emergent (or complex) systems and suggests that emergence is a product of coupled, context-dependent interactions. These interactions, together with the resulting system are nonlinear, where the overall behavior of the system cannot be obtained by summing up the behavior of its constituent parts. Temperature and pressure are easy to grasp examples of emergent properties. Individual molecules in motion or closely placed result in temperature and pressure, but the molecules by themselves do not present a temperature or a pressure.

For short, the complex and flexible behavior of the whole depends on the activity and interactions of a large number of agents described by a relatively small number of rules. There is no easy way to predict the overall behavior of the system by simply looking at the local rules that govern individual behaviors, though we may have some intuition about it. The difficulty increases even further when the individual agents can adapt; that is, when the complex systems become complex adaptive systems. Then, an individual's strategy (or the environment) is not only conditioned by the current state of the system, it can also change over time. Even when adaptation may not affect an individual directly, it may still affect it indirectly. For instance, stigmergic effects result in the adaptation of the environment as a result of the action of individual agents. This, in turn, results in different patterns of behavior for the agents. What is important to realize, thus, is that the increase in complexity results in an increase of the possibilities for emergent phenomena.

To make the conceptual idea of emergence even clearer, consider the following example. Imagine a computer simulation of highway traffic. Simulated 'semi-intelligent' cars are designed to drive on the highway based on specific interactions with other cars following a finite set of rules. Several rules can be defined, such as:

- Cars should drive on the high speed lane if their speed is over 100Km/h, otherwise they should drive on the lower speed lanes.
- Cars should only take over using the higher speed lane.
- Avoid collisions.
- Stop in the red light.
- Vary speed smoothly.

As the cars can vary speed, change lanes, and are subjected to several constraints (speed, traffic lights, road limits, etc.), their programming enables them to have unexpected behaviors. Such a system would define a form of complex behavior, for there are several agents dynamically interacting with one another and the environment in multiple ways, following local rules and unaware of any higher-level instructions. Nevertheless, the behavior of such a system can only be considered emergent if discernible macro-behaviors are observed. For instance, these lower-level rules may result in the appearance of traffic jams and a higher flux of cars on the high-speed lane than on the lower speed ones. All these are emergent phenomena that were not programmed in the local rules applied to the cars. They simply emerged as outcomes of the lower-level interactions, even though some of them could be predicted.

Several features and behaviors of natural systems are not so obviously emergent; only a detailed analysis of complex interactions at the organism level can show that they are genuinely new features that only appear at the higher levels. However, emergent behaviors of natural systems can be observed through the creation of models. Even highly abstract and simplified models, such as the ones that will be reviewed here, allow us to simulate, emulate, observe, and study several emergent behaviors of natural complex adaptive systems. In particular, DNA strands, chromosomes, ant colonies, neurons, immune systems, flocks of birds, and schools of fish, will all be seen to present a large variety of emergent properties.

Reductionism

The classical *vitalist* doctrines of the 18th century are based on the idea that all life phenomena are animated by immaterial life spirits. These life spirits determine the various life phenomena, but are themselves unexplainable and indescribable from a physical perspective. By contrast, the *reductionist* position, also in the 18th century, insisted that a large part, if not all, of the life phenomena can be reduced to physics and chemistry (Emmeche et al., 1997).

For long, scientists have been excited about the belief that natural systems could be understood by reductionism; that is, by seeking out their most fundamental constituents. Physicists search for the basic particles and forces, chemists seek to understand chemical bonds, and biologists scrutinize DNA sequences and molecular structures in an effort to understand organisms and life. These reductionist approaches suggest that questions in physical chemistry can be answered based on atomic physics, questions in cell biology can be answered based on how biomolecules work, and organisms can be understood in terms of how their cellular and molecular systems work (Gallagher and Appenzeller, 1999; Williams, 1997).

However, apart from a few radicals, the reductionists do not claim that the higher psychological functions can be reduced to physics and chemistry. As an outcome of the scientific development in many areas, such as cytology, neuroanatomy, immunology, and neurophisiology, it became very difficult to maintain the more classical positions (Emmeche et al., 1997).

Advances in science and technology have led to transformations in the vitalists' and reductionists' positions as well. After a number of scientific discoveries in the early 19th century, the vitalists gradually limited their viewpoints to a narrower field. They now insisted that only the higher psychological functions were irreducible, but admitted that a large range of biological phenomena could be described scientifically. Reductionists now claimed that every phenomenon in the world, including the highest psychological ones, could be reduced to physics and chemistry (Emmeche et al., 1997).

Although the reductionist approaches work to some extent, scientists are now beginning to realize that reductionism is just one of the many tools needed to uncover the mysteries of nature. There might be additional principles to life and nature embodied in properties found at higher levels of organization, and that cannot be seen in lower levels. These properties are known as emergent properties. For instance, it is still not possible to understand higher psychological phenomena, such as love and hate, by simply looking at how neurons work.

2.2.8. Bottom-up vs. Top-down

Broadly speaking, there are two main approaches to addressing the substantive question of how in fact nature works. One exploits the model of the familiar serial, digital computer, where representations are symbols and computations are formal rules (algorithms) that operate on symbols. For instance, 'if-then' rules are most often used in formal logic and circuit design. The second approach is rooted in many natural sciences, such as biology, neuroscience, and evolutionary theory, drawing on data concerning how the most elementary units work, interact, and process information. Although both approaches ultimately seek to reproduce input-output patterns and behaviors, the first is more *top-down*, relying heavily on computer science principles, whereas the second tends to be more *bottom-up*, aiming to reflect relevant natural constraints.

Bottom-Up

Most complex systems exhibiting complex autonomous patterns and behaviors are parallel and distributed systems of interacting components taking decisions that directly affect their state and the state of some other components (e.g., their neighbors). Each component's decisions are based on information about its own local state. Bottom-up systems and models are those in which the global behavior emerges out of the aggregate behavior of an ensemble of relatively simple and autonomous elements acting solely on the basis of local information.

The reductionist approach is to some extent a bottom-up approach. Reductionism assumes that the study of individual components is sufficient for a full understanding of the organism as a whole. Bottom-up approaches also seek for the study of component parts, but do not necessarily claim that the whole is just the sum of its parts; it allows for emergent phenomena as well. Biology is in most cases a reductionist scientific enterprise (it has been changing over the last few years though). Natural computing, although highly inspired or based on biology, is more rooted on bottom-up approaches than on purely reductionist techniques. Theories about complexity and emergence are pervasive in natural computing, and may be of primary importance for the study and formalization of natural computing techniques.

Another way of viewing bottom-up systems or approaches is related to how they develop or evolve; a design perspective: the most deeply embedded structural unit is created first and then combined with more of its parts to create a larger unit, and so on. For instance, if a system develops or evolves by constantly adding new parts to the system until a given criterion or state of maturity is achieved, this system is said to follow a bottom-up design.

A classical example of bottom-up design in natural computing involves artificial neural networks whose architecture is not defined *a priori*; they are a result of the network interactions with the environment. Assume, for instance, that you are trying to design an artificial neural network capable of recognizing a set of patterns (e.g., distinguish apples from oranges). If the initial network has a single neuron and more neurons are constantly being added until the network is capable of appropriately recognizing the desired patterns, this network can be said to follow a bottom-up architecture design. In nature, a classical example of a bottom-up system is the theory of evolution, according to which life on Earth is a result of a continuous and graded procedure of adaptation to the environment.

Top-Down

In the early days of artificial intelligence, by the time of the Dartmouth summer school in 1956, most researchers were trying to develop computer programs capable of manipulating symbolic expressions. These programs were developed in a *top-down* manner: by looking at how humans solve problems and trying to 'program' these problem-solving procedures into a computer. Top-down approaches assume that it is possible to fully understand a given system or process by looking at its macro-level patterns.

Most artificial intelligence techniques based upon the top-down paradigm are known as knowledge-based or expert systems. They rely upon the existence of a knowledge base or production system containing a great deal of knowledge about a given task provided by an expert. In these systems, an action is taken if a certain specific condition is met. A set of these production systems results in an intelligent system capable of inferring an action based on a set of input conditions. Knowledge-based systems were rooted on a philosophy inspired by cognitive theories of the information processes involved in problem solving.

Top-down systems and approaches can also be studied under the perspective of how the system develops with time - the design perspective: an over-complicated system is initially designed and then parts of it are eliminated so as to result in a satisfactorily more parsimonious structure still capable of performing its task. We have discussed, in the previous section, that it is possible to design an artificial neural network for pattern recognition by simply adding neurons until the desired patterns are recognized. The opposite direction could also be adopted: an over-sized network could be initially designed and neurons would then be pruned until a network of reasonable size remained, still capable of meeting its goal.

To make the distinction even clearer, consider the case of building a sand castle. A bottom-up approach is the one in which you keep pouring sand and molding it, and a top-down approach is the one in which you initially pour a lot of sand, making a big mountain of sand, and then you start molding the castle from the mountain of sand.

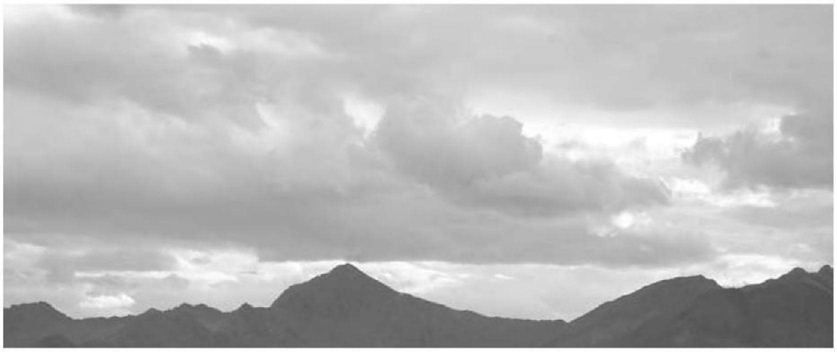
2.2.9. Determinism, Chaos, and Fractals

One of the classical positions in the theory of science is that scientific theories are capable of providing deterministic relations between the elements being investigated. A *deterministic system* can have its time evolution predicted precisely; all events are inevitable consequences of antecedent sufficient causes. The main characteristic of this type of deterministic system is *predictability*. When it is possible to predict the development or time evolution of a system from some predefined conditions there is a deterministic relation between the elements that constitute the system. This classical perspective demands the capacity of predicting the time evolution of a system, thus precluding the appearance of new and emergent phenomena.

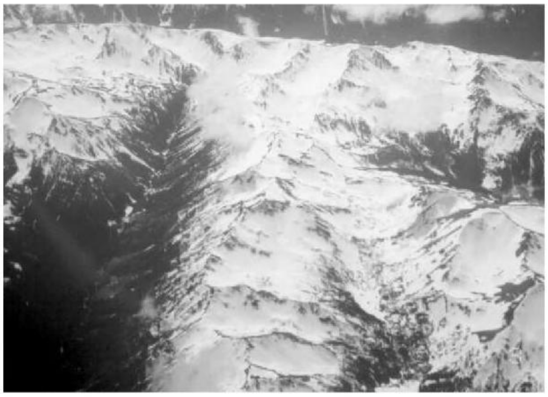
One of the most interesting and exciting results of recent scientific development, mainly in physics, is the remodeling of the relation between determinism and prediction. It is now evident that there are many systems that can be described adequately as being strictly deterministic but that still remain unpredictable. The impossibility of predicting the properties arising within many systems considered totally deterministic is the consequence of the well-known Poincaré's treatment of the three-body problem and the Hadamard's investigation of the sensitivity to initial states - insights from the latter half of the 19th century that have recently given rise to the *chaos theory*. Several processes in physics and biology are deterministic but unpredictable. Thus, one of the very important theoretical consequences of chaos theory is the divorce between determinism and predictability.

Before chaos theory, scientists (mainly physicists) were suffering from a great ignorance about disorder in the atmosphere, in the turbulent sea, in the fluctuations of wildlife populations, in the oscillations of the heart and the brain. The irregular side of nature, the discontinuous and erratic side, has been a puzzle to science. The insights from chaos theory led directly into the natural world - the shapes of clouds, the paths of lightening, the microscope intertwining of blood vessels, the galactic clustering of stars, the coast lines. Chaos has created special techniques of using computers and special kinds of graphic images, pictures that capture fantastic and delicate structures underlying complexity. The new science has spawned its own language, such as the word *fractals*.

The word fractal comes to stand for a way of describing, calculating, and thinking about shapes that are irregular and fragmented, jagged and broken-up - shapes like the crystalline curves of snowflakes, coast shores, mountains, clouds, and even the discontinuous dusts of galaxies (see Figure 2.6). A fractal curve implies an organizing structure that lies hidden among the hideous complication of such shapes.



(a)



(b)

Figure 2.6: Examples of the fractal geometry of nature. (a) Clouds. (b) Top of mountains in Alaska.

Fractals - the term used to describe the shape of chaos - seem to be everywhere: a rising column of cigarette smoke breaks into swirls; a flag snaps back and forth in the wind; a dripping faucet goes from a steady pattern to a chaotic one; and so forth (Gleick, 1997; Stewart, 1997).

Chaos theory is often cited as an explanation for the difficulty in predicting weather and other complex phenomena. Roughly, it shows that small changes in local conditions can cause major perturbations in global, long-term behavior in a wide range of 'well-behaved' systems, such as the weather. Therefore, chaos embodies three important principles: sensitivity to initial conditions, cause and effect are not proportional, and nonlinearities. This book talks very little about chaos; however Chapter 7 describes the Fractal Geometry of Nature as the main branch of the study of biology by means of computers aimed at creating life-like shapes or geometrical patterns.

2.3 SUMMARY

This chapter started with a discussion of what are models, metaphors, experiments, simulations, emulations, and realizations. These concepts are important for they allow us to distinguish natural computing techniques from theoretical models, computer simulations from realizations, and so on. Some comments about the difficulty in creating a general framework to design natural computing systems were also made. However, it was argued that each approach, with the exception of some topics in the second part of this book, do have specific frameworks for their design. Also, it was emphasized that this is a book about how existing techniques can be understood, reproduced, and applied to particular domains, not a book about how to engineer new techniques. Of course, from reading this text the reader will certainly get the feeling of natural computing and will then find it much easier to go for his/her own personal explorations and design of novel natural computing approaches.

If the contents of this chapter were to be summarized in a single word, this word would be *complexity* or *complex system*. Complexity encompasses almost all the terminology discussed here. It may involve a large number of interacting individuals presenting or resulting in emergent phenomena, self-organizing processes, chaos, positive and negative feedback, adaptability, and parallelism. Although this book is not about the *theory of complex systems*, it provides design techniques, pseudocode, and applications for a number of complex systems related with nature.

2.4 EXERCISES

2.4.1. Questions

1. Provide alternative definitions for all the concepts described in Section 2.2.
2. List ten journals that can be used as sources of information about natural systems and processes that could be useful for natural computing and explain your choice.
3. Name two connectionist systems in nature in addition to the nervous system and the immune network. Explain.
4. Section 2.2.5 presented the concepts of positive and negative feedback. In Section 2.2.3, the presence of an initial deposit of soil pellets was demonstrated to stimulate worker termites to accumulate more pellets through a positive feedback mechanism. It is intuitive to think that if no negative feedback mechanism existed in this process, the process could go uncontrolled. Name three negative feedback mechanisms involved in the termite mound building process.
5. Name a natural system or process that involves both, positive and negative feedback, and describe how these are observed.

6. Exemplify some natural systems (excluding those involving humans) that exhibit alternatives to self-organization. That is, provide an example of a natural system whose parts behave in a follow a 'leader' manner, one that follows a 'blueprint', one that follows a 'recipe', and one that follows a 'template'. The resultant pattern or process may vary from one case to another. Provide a list of references.
7. Discuss the advantages and disadvantages of self-organization over its alternatives.
8. The discussion about self-organization shows it is potentially relevant to fundamental questions about evolution. One issue is whether life itself can come into existence through a self-organizing process. A second issue is the relationship between natural selection and self-organization once life is up and running. Discuss these two possible implications of self-organization to life.
9. Two other important concepts in natural computing are *competition* and *cooperation* among agents. Provide definitions for both concepts under the perspective of agent-based theory.

2.4.2. Thought Exercise

1. An example of vehicular traffic flow was given as an instance of an emergent system (Section 2.2.7). Assume that there is a road with a single lane, a radar trap is installed in this road and a number of cars are allowed to run on this road. Embody the following rules in each car:
   - If there is a car close ahead of you, slow down.
   - If there isn't any car close ahead of you, speed up (unless you are already at maximum speed).
   - If you detect a radar trap, then slow down.

   What types of emergent behavior would you expect from this system? Will there be any traffic jam?

   (After answering the question, see Chapter 8 for some possible implications.)

2.4.3. Projects and Challenges

1. Write a computer program to simulate the traffic flow described in the exercise above. Compare the results obtained with your conclusions and the discussion presented in Chapter 8.
2. It was discussed in Section 2.2.6 that some patterns coating the skin of many animals are resultant from self-organizing processes. Perform a broad search for literature supporting this claim. Name, summarize, and discuss the most relevant works, and give your conclusive viewpoint.

3. Write an essay discussing the similarities and differences between the concepts *autopoiesis*, *autocatalisys*, and *self-organization*. The essay should be no longer than 10 pages. Provide a list of references.
4. Name other alternatives to self-organization and give examples.
5. From our math courses we know that a point is a zero-dimensional geometrical form, a line is a one-dimensional form, a square is a two-dimensional form, and so on. Can you determine the dimension of a snowflake? Justify your answer.

2.5 REFERENCES

[1] Baas, N. A. and Emmeche, C. (1997), "On Emergence and Explanation", *Intellectica*, **1997/2**(25), pp. 67–83.

[2] Bonabeau, E., Dorigo, M. and Théraulaz, G. (1999), *Swarm Intelligence from Natural to Artificial Systems*, Oxford University Press.

[3] Camazine, S., Deneubourg, J. -L., Franks, N. R., Sneyd, J., Theraulaz, G. and Bonabeau, E. (2001), *Self-Organization in Biological Systems*, Princeton University Press.

[4] Cowan, G., Pines, D. and Meltzer, D. (Eds.) (1994), *Complexity: Metaphors, Models, and Reality*, Proc. Volume XIX, Santa Fe Institute, Studies in the Sciences of Complexity. Addison-Wesley Pub. Company.

[5] de Castro, L. N. and Von Zuben, F. J. (2004), *Recent Developments in Biologically Inspired Computing*, Idea Group Inc.

[6] Decker, E. H. (2000), "Self-Organizing Systems: A Tutorial in Complexity", *Vivek: A Quarterly in Artificial Intelligence*, **13**(1), pp. 14–25.

[7] Dorigo, M., Bonabeau, E. and Theraulaz, G. (2000), "Ant Algorithms and Stigmergy", *Future Generation Computer Systems*, **16**, pp. 851–871.

[8] Emmeche, C. (1997), "Aspects of Complexity in Life and Science", *Philosophica*, **59**(1), pp. 41–68.

[9] Emmeche, C., Koppe, S. and Stjernfelt, F. (1997), "Explaining Emergence: Towards an Ontology of Levels", *Journal for General Philosophy of Science*, **28**, pp. 83–119.

[10] Franklin, S. and Graesser, A. (1997), "Is It an Agent, or Just a Program?: A Taxonomy for Autonomous Agents", in J. P. Muller, M. J. Wooldridge, and N. R. Jennings (eds.), *Intelligent Agents III: Agents Theories, Architectures, and Languages*, Springer-Verlag, pp. 21–35.

[11] Gallagher, R. and Appenzeller, T. (1999), "Beyond Reductionism", *Science*, **284**(5411), pp. 79.

[12] Gleick, J. (1997), *Chaos: Making a New Science*, Vintage.

[13] Goldenfeld, N. and Kadanoff, L. P. (1999), "Simple Lessons from Complexity", *Science*, **284**(5411), pp. 87–89.

[14] Grassé, P. -P. (1959), "La Reconstruction du Nid et Les Coordinations Interindividuelles Chez *Bellicositermes Natalensis* et *Cubitermes sp.* La Théorie de la Stigmergie: Essai d'interprétation du Comportement des Termites Constructeurs", *Insect. Soc.*, **6**, pp. 41–80.

[15] Holland, J. H. (1995), Hidden Order: How Adaptation Builds Complexity, Addison-Wesley.

[16] Holland, J. H. (1998), *Emergence: From Chaos to Order*, Oxford University Press.

[17] Holland, O. and Melhuish, C. (1999), "Stigmergy, Self-Organization, and Sorting in Collective Robotics", *Artificial Life*, **5**(2), pp. 173–202.

[18] Johnson, S. (2002), Emergence: The Connected Lives of Ants, Brains, Cities and Software, Penguin Books.

[19] Kennedy, J., Eberhart, R. and Shi. Y. (2001), *Swarm Intelligence*, Morgan Kaufman Publishers.

[20] Lewin, R. (1999), *Complexity: Life at the Edge of Chaos*, 2nd. Ed., Phoenix.

[21] McClelland, J. L., Rumelhart, D. E. and the PDP Research Group (1986), Parallel Distributed Processing: Explorations in the Microstructure of Cognition, Volume 2: Psychological and Biological Models, The MIT Press.

[22] Mehler, A. (2003), "Methodological Aspects of Computational Semiotics", *S.E.E.D. Journal* (*Semiotics, Evolution, Energy, and Development*), **3**(3), pp. 71–80.

[23] Michener, C. D. (1974), *The Social Behavior of Bees: A Comparative Study*, Harvard University Press.

[24] Morowitz, H. and Singer, J. L. (Eds.) (1995), *The Mind, The Brain, and Complex Adaptive Systems*, Addison-Wesley Publishing Company.

[25] O'Reilly, R. C. and Munakata, Y. (2000), Computational Explorations in Cognitive Neuroscience: Understanding the Mind by Simulating the Brain, The MIT Press.

[26] Pattee, H. H. (1988), "Simulations, Realizations, and Theories of Life", in C. Langton (ed.), *Artificial Life*, Addison-Wesley, pp. 63–77.

[27] Paton, R. (1994), *Computing with Biological Metaphors*, International Thomson Computer Press.

[28] Peck, S. L. (2004), "Simulation as Experiment: A Philosophical Reassessment for Biological Modeling", *Trends in Ecology and Evolution*, **19**(10), pp. 530–534.

[29] Rocha, L. M. (1999), "Complex Systems Modeling: Using Metaphors from Nature in Simulation and Scientific Models", *BITS: Computer and Communications News*, Computing, Information, and Communications Division, Los Alamos National Laboratory, November.

[30] Rumelhart, D. E., McClelland, J. L. and the PDP Research Group (1986), Parallel Distributed Processing: Explorations in the Microstructure of Cognition, Volume 1: Foundations, The MIT Press.

[31] Russell, S. J. and Norvig, P. (1995), *Artificial Intelligence: A Modern Approach*, Prentice Hall.

[32] Science (1999), Special Issue on Complex Systems, *Science Magazine*, **284**(5411).

[33] Solé, R. and Goodwin, B. (2002), *Signs of Life: How Complexity Pervades Biology*, Basic Books.

[34] Stewart, I. (1995), *Nature's Numbers*, Phoenix.

[35] Stewart, I. (1997), Does God Play Dice?: The New Mathematics of Chaos, Penguin Books.

[36] Trappenberg, T. (2002), *Fundamentals of Computational Neuroscience*, Oxford University Press.

[37] Williams, N. (1997), "Biologists Cut Reductionist Approach Down to Size", *Science*, **277**(5325), pp. 476–477.

[38] Wilson, R. A. and Keil, F. C., (eds.) (1999), *The MIT Encyclopedia of the Cognitive Sciences*, The MIT Press.

[39] Wooldridge, M. J. and Jennings, N. R. (1995), "Intelligent Agents: Theory and Practice", *Knowledge Engineering Reviews*, **10**(2), pp. 115–152.

It is possible, on the other side, to assess and compare the performance of different algorithms in specific problem domains and, therefore, it is possible to look for a technique that provides the best performance, on average, in this domain.

In contrast to the global optimum, a *local optimum* is a potential solution $x \in F$ in respect to a neighborhood N of a point y, if and only if $eval(x) \leq eval(y)$, $\forall\, y \in N(x)$, where $N(x) = \{y \in F : dist(x,y) \leq \varepsilon\}$, *dist* is a function that determines the distance between x and y, and ε is a positive constant. Figure 3.1 illustrates a function with several local optima (minima) solutions, a single global optimum (minimum), and the neighborhood of radius ε around one of the local optima solutions.

The evaluation function defines a response surface, which will later be termed *fitness landscape* (Section 3.4.4), that is much like a topography of hills and valleys. The problem of finding the (best) solution is thus the one of searching for a peak, assuming a maximization problem, in such a fitness landscape. If the goal is that of minimization, then a valley has to be searched for. Sampling new points in this landscape is basically made in the immediate vicinity of current points, thus it is only possible to take local decisions about where to search next in the landscape. If the search is always performed uphill, it might eventually reach a peak, but this might not be the highest peak in the landscape (global optimum). The search might sometimes go downhill in order to find a point that will eventually lead to the global optimum. Therefore, by not knowing the fitness landscape and using only local information, it is not possible to guarantee the achievement of global optima solutions.

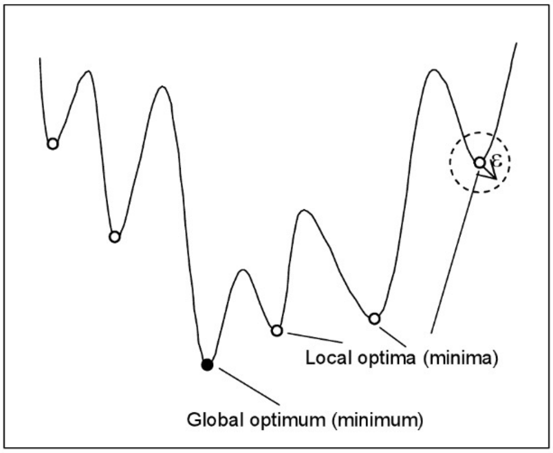


Figure 3.1: Illustration of global and local minima of an arbitrary function. Dark circle: global minimum; White circles: local minima.