A Comprehensive Overview of the Applications of Artificial Life

Kyung-Joong Kim
Sung-Bae Cho*
Department of Computer Science
Yonsei University
134 Shinchon-dong, Sudaemoon-ku
Seoul 120-749
Korea
kjkim@cs.yonsei.ac.kr
sbcho@cs.yonsei.ac.kr

Abstract We review the applications of artificial life (ALife), the creation of synthetic life on computers to study, simulate, and understand living systems. The definition and features of ALife are shown by application studies. ALife application fields treated include robot control, robot manufacturing, practical robots, computer graphics, natural phenomenon modeling, entertainment, games, music, economics, Internet, information processing, industrial design, simulation software, electronics, security, data mining, and telecommunications. In order to show the status of ALife application research, this review primarily features a survey of about 180 ALife application articles rather than a selected representation of a few articles. Evolutionary computation is the most popular method for designing such applications, but recently swarm intelligence, artificial immune network, and agent-based modeling have also produced results. Applications were initially restricted to the robotics and computer graphics, but presently, many different applications in engineering areas are of interest.

Keywords
Artificial life, perspective, simulation, software, hardware, evolution

1 Introduction

There are two types of modeling approaches for studying natural phenomena: the top-down approach (involving a complicated, centralized controller that makes decisions based on access to all aspects of the global state), and the bottom-up approach, which is based on parallel, distributed networks of relatively simple, low-level *agents* that simultaneously interact with each other. Most traditional artificial intelligence (AI) research focuses on the former approach.

However, to obtain the most adaptive and complex behaviors from intelligent applications that assist humans, such behaviors might be designed using the bottom-up approach. It is difficult, or even impossible, to model lifelike behaviors using the traditional AI approach. Usually, complex behaviors such as schooling of fishes, detection of unknown attack patterns, and evolution of economic markets can be modeled as the local interaction of agents whose decisions are based on the information about, and that directly affect, only their own local environment. There are many complex applications, not only in the robotics, computer graphics, and engineering fields, but also in the educational and artistic fields. It is too tedious and sometimes impossible to design such applications using the top-down approach.

Artificial life (also known as *ALife*) is an interdisciplinary study of life and lifelike processes that uses a synthetic methodology [1]. It complements traditional biological sciences concerned with the analysis of living organisms and finds the mechanisms of evolutionary processes for automatic

*Corresponding author.

design and creation of artifacts. A general property of ALife is that the whole system's behavior is represented only indirectly, and arises out of interactions of individuals with each other. In this context, known as the *philosophy of decentralized architecture*, we can say that ALife shares important similarities with some new trends in AI, including connectionism [2], multi-agent AI [3], and evolutionary computation [4].

According to Bedau [1], there are three branches of ALife. *Soft* ALife creates simulations or other purely digital constructions that exhibit lifelike behavior, *hard* ALife produces hardware implementations of lifelike systems, and *wet* ALife synthesizes living systems out of biochemical substances. For this overview, ALife articles of the soft and hard branches were surveyed, because most articles that apply to ALife fall under one of those two rubrics.

Scientific terms used in this review include the following. *Evolutionary computation* (EC) is a biologically inspired general computational concept that uses genetic algorithms (GAs), evolutionary strategies (ESs), genetic programming (GP), and evolutionary programming (EP). *Interactive evolutionary computation* (IEC) is the technology in which EC optimizes the target systems according to subjective human evaluation expressed as fitness values for system outputs. *Agent-based modeling* (AM) uses multiple agents whose decisions are entirely based on local information in order to simulate real-world situations. *Cellular automata* (CAs) are discrete dynamical systems that specify behavior in terms of a local relation. *Ant colony optimization* (ACO) is a meta-heuristic that uses artificial ants to find solutions to difficult combinatorial optimization problems.

The methodologies of ALife are described in Section 2, and an ALife application survey follows in Section 3. Finally, we discuss the future of ALife application research.

2 Basic Methodologies of ALife

Recently, the concept of *emergence* has arisen in research that involves nonlinear dynamics, ALife, complex systems, and behavior-based robotics. Emergence in a system, broadly, is said to refer to properties or behaviors that cannot easily be predicted from internal properties. Examples of this are flocking behaviors in simulated birds from a set of three simple steering behaviors [5], patterns in the game of life [6, pp. 817–850], and the highway pattern displayed by the artificial ant [7].

It is this concept of emergence that highlights the nature of ALife research. Emergence is exhibited by a collection of interacting entities whose global behaviors cannot be reduced to a simple aggregate of the individual contributions of the entities. Conventional methods in AI have to struggle to reveal and explain emergence, because they are generally reductionist. That is, they reduce systems to constituent subsystems and then study them in isolation (the top-down approach). In contrast, ALife adopts the bottom-up approach which starts with a collection of entities exhibiting simple and well-understood behavior and then synthesizes more complex systems. Technologies in ALife research include CAs, the Lindenmayer system (L-system), GAs, and neural networks. Their relations are diagrammed in Figure 1.

EC is a model of machine learning derived from the evolution process in nature. There are several different types of evolutionary computations: GAs, EP, and ESs. They are all population-based search algorithms. Different representation or encoding schemes and search operators differentiate ECs. For example, GAs normally use crossover and mutation as search operators, while EP only uses mutation [8]. GAs, the most popular method, are executed by creating a population of individuals that are represented by strings. The individuals in the population go through an evolutionary process in which individuals compete for resources in the environment. Stronger individuals are more likely to survive and propagate their genetic material to offspring. Interactive EC is such a technique whose fitness function is replaced by a human user [9]. EC is used for learning, adaptation, and searching, and there are many hybrid methodologies for synergism.

In agent-based modeling, global behaviors emerge from the local interaction of agents, and the states of the agents change in response to their neighborhoods' states. It is assumed that each agent can decide to learn a new skill and to cooperate with other agents.

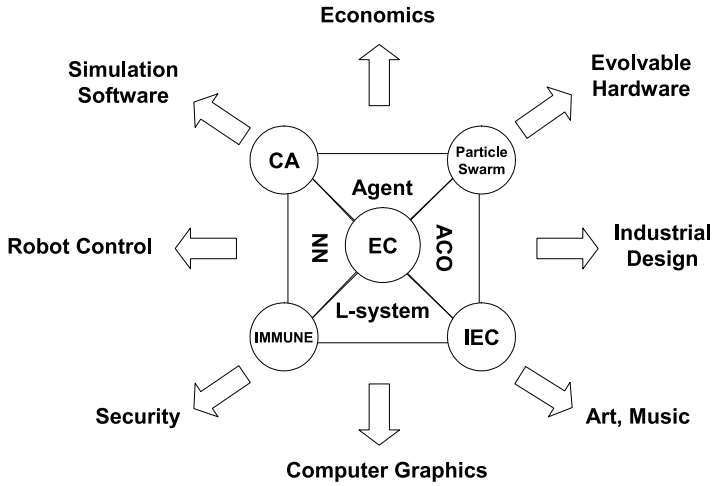

Figure 1. Relations of the ALife methodologies (CA = cellular automata, IEC = interactive evolutionary computation, EC = evolutionary computation, NN = neural network, ACO = ant colony optimization).

Multi-agent systems include methodologies derived from the behavior of insects. More recently, scientists have been turning to insects for ideas that can be used for heuristics. Ant colony optimization (ACO) is a meta-heuristic that uses artificial ants to find solutions to difficult combinatorial optimization problems [13]. Particle swarm optimization is based on the behavior of bees. Each agent has its own advantages, and a colony of agents produces the optimal solution. The direction of movement of each agent is determined as the vector sum of directional vectors for the personal best, global best, and current motion. ACO is a popular method, and has many applications, including electronics, industrial design, chemical process design, and data mining. Particle swarm optimization has relatively few applications. A new swarm intelligence algorithm has been developed by Abbass, based on the haploid-diploid genetic breeding of honeybees and known as honeybees optimization (HBO), for solving combinatorial optimization problems [14–17].

The L-system is frequently used to model trees, flowers, feathers, and roots of plants in computer graphics. It is a set of simple grammar rules, which generates complex sentences from primitive components. Although the rules are simple, the generated objects are very similar to the real structures.

An NN can be considered as a mapping device between input and output sets. Considerable literature has described NN-fuzzy, fuzzy-GA, GA-NN, and NN-fuzzy-GA mappings. When trying to solve a real-world problem with NNs, we are faced with a large variety of learning algorithms and a vast selection of possible network architectures. Recent theoretical and experimental studies indicate that we can improve the performance of NNs by considering methods for combining them.

CAs' basic function is computation—there is no autonomy. A CA is a population of interacting cells, each of which is itself a computer (automaton) that can be made to represent many kinds of complex behaviors by building appropriate rules into it [10–12]. CAs can model ecological systems or the behavior of insects, and can also be used for image processing and neural network construction [12]. CAs can consist of a 1D string of cells, a 2D grid, or a 3D solid. Usually the cells are arranged in a simple rectangular grid. CAs have three essential features: state, neighborhood, and program. The *state* is a variable that takes a discrete value for each cell. It can be either a number or a property. A cell's *neighborhood* is the set of cells that it interacts with. In a grid these are normally the closest cells. The *program* is the set of rules that define how the state changes in response to the current state and those of the neighborhood cells [10].

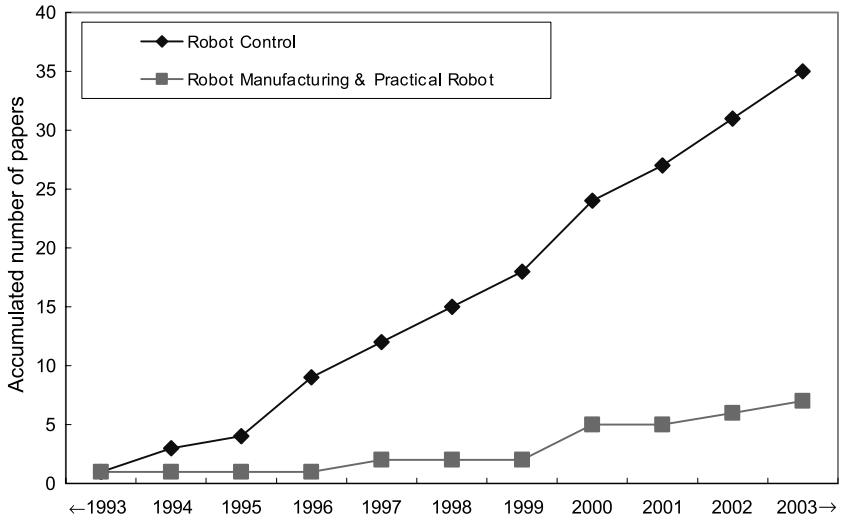

Figure 2. Statistics of application articles in artificial life by field and year: robotics.

3 ALife Applications

The first conference on artificial life, in 1989, where the term "artificial life" was coined, gave recognition to ALife as a field in its own right [18]. Statistics on application articles of ALife are shown in Figures 2, 3, 4, and 5. In general, two major research streams developed during the 1990s from the interdisciplinary cooperation of researchers interested in bio-inspired computing. A practical goal of ALife can be defined as finding a mechanism for an evolutionary process to be used in the automatic design and creation of artifacts [19], and there are many contributions by computer science researchers to implement artifacts. Drawing upon some of the computing techniques inspired by social insects such as ants [20], several mobile-agent-based paradigms were designed to solve control and routing problems in telecommunications and networking [21]. The natural immune system is also a source of inspiration for developing intelligent methodologies

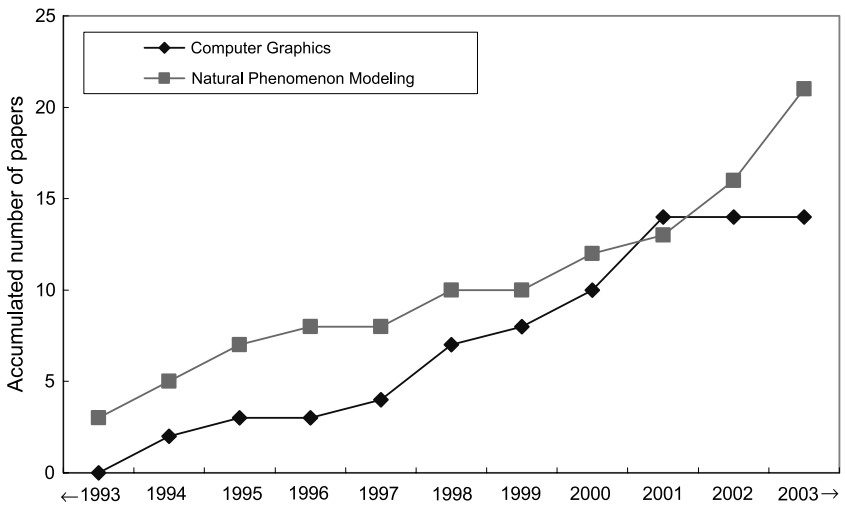

Figure 3. Statistics of application articles in artificial life by field and year: graphics.

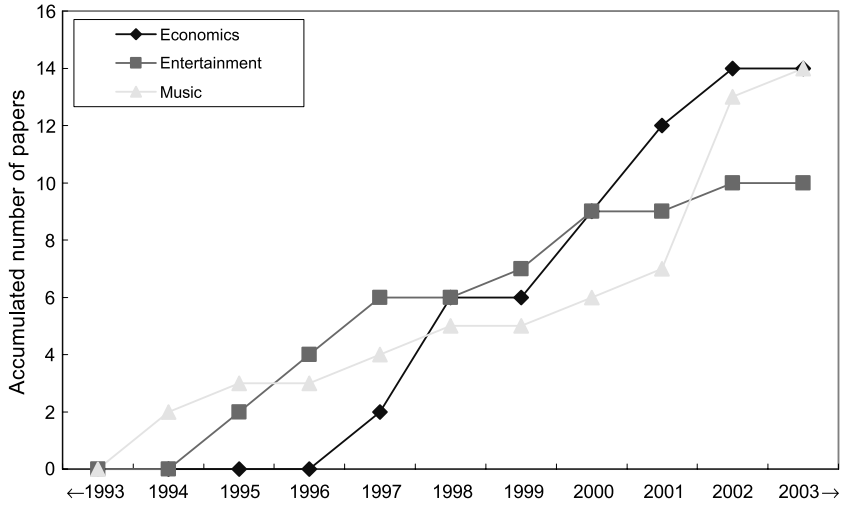

Figure 4. Statistics of application articles in artificial life by field and year: social.

toward problem solving [22]. Neural networks have been widely used in optimization, regression, and prediction problems.

Figure 6 describes the increase of the number of application articles. This recent rapid increase is related to similar publications in mechanics, product design, and chemical processes. Figure 7 shows the proportion of each application area. Robot control is the most active research area in ALife. The robotics, graphics, social, and engineering areas are 22%, 21%, 21%, and 36%, respectively.

Statistics on application articles about ALife by methodology are shown in Figure 8. Some of the articles are not counted for the statistics because they are surveys or introductory articles without technical depth. Also, if the amount of research is relatively small for the specific methodology, the research is ignored. The figure shows the percentages of methodology usage. EC-related methodologies (EC, EC with NN, EC with L-system, and interactive EC) occupy about 50% of the research. Figure 9 shows the portion of the methodologies by categories. Of the robotics research,

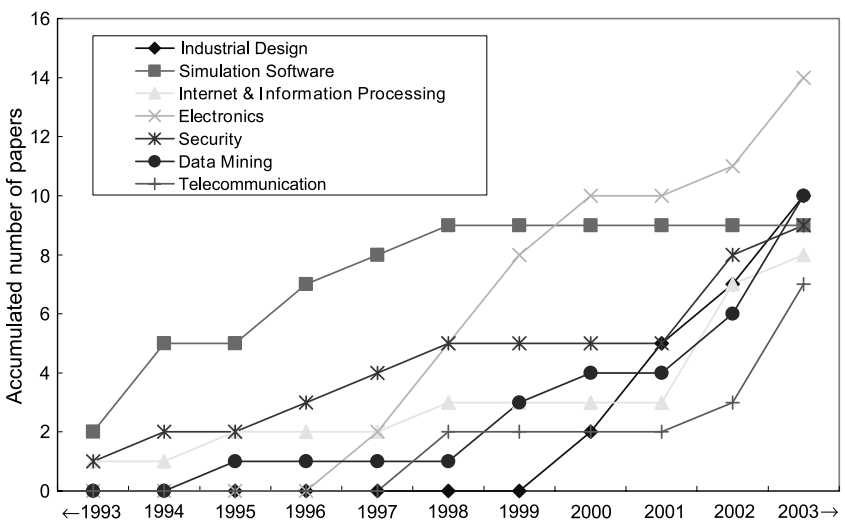

Figure 5. Statistics of application articles in artificial life by field and year: engineering.

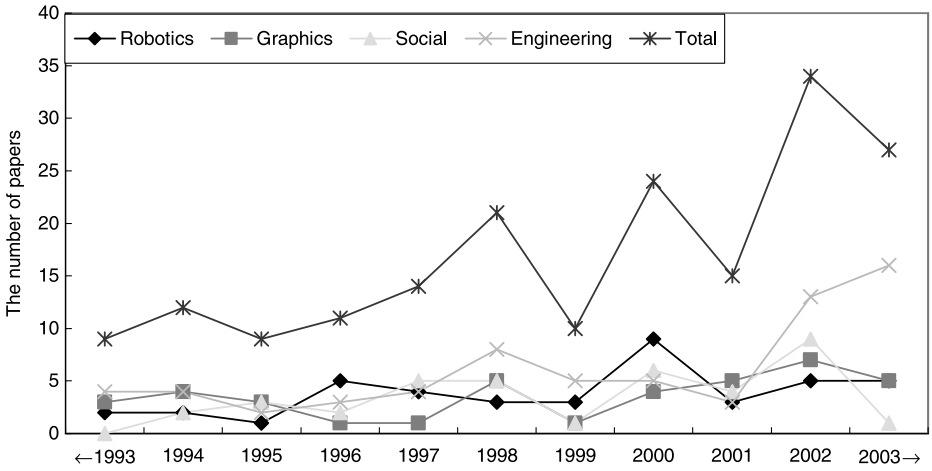

Figure 6. The number of application articles by year and application category.

EC-related methodologies occupy about 80%. Most notably, EC with NNs comprises 50% of all the research. This means that the hybrid method is the most popular approach for designing controllers for robots. In graphics, the proportion of EC-related methods is about 50%. It is remarkable that the methodologies related to L-systems comprise about 43% of the research. The L-system is the most popular representation for researchers who model natural phenomena such as growth of plants, feathers, and forests. In the social field (including social science and human-related applications such as entertainment and music), the agent-based model is dominant, because most articles in economics use agent-based modeling to simulate real-world economic phenomena. In engineering, the proportion of EC-related methods is relatively small. Swarm intelligence comprises 34% of the research because of the popularity of ACO.

Statistics on application articles about ALife by methodology and year are shown in Figures 10 and 11. Figure 12 shows that EC-related methods are being used continually, with minor increases, and that methods other than EC have recently gained interest among researchers.

3.1 Robot Control

Recently, the robotics area has relied heavily on EC for designing controllers. EC with NNs is especially dominant. This subsection includes evolving controllers, the combination of EC and NNs,

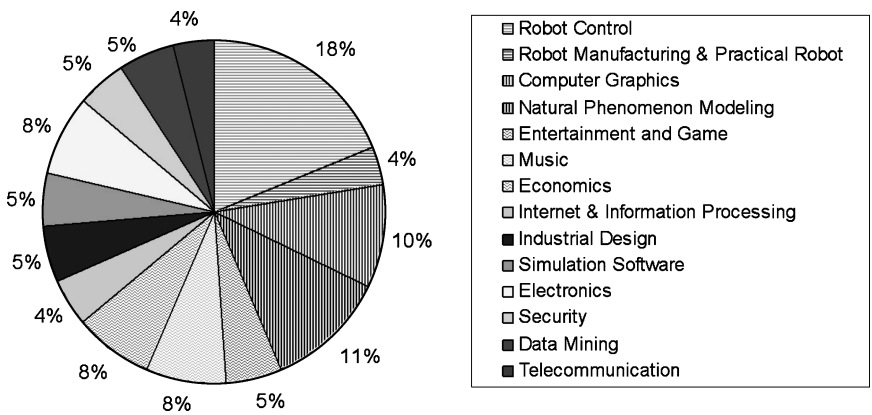

Figure 7. The proportion of each application area.

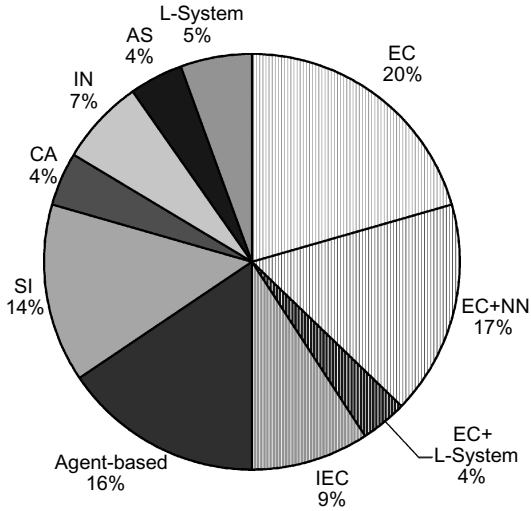

Figure 8. The percentage of each methodology usage (EC = evolutionary computation, NN = neural networks, IEC = interactive evolutionary computation, SI = swarm intelligence, CA = cellular automata, IN = artificial immune networks, and AS = action selection).

coevolution, collective robotics, and CAs for robotics. Basically, designing controllers for robots is the main interest, but there are many other interesting issues in robotics. These include map building, planning, and human-robot interaction (HRI). CAs can be used for planning, and coevolution with humans can be the solution for HRI. Another issue of evolutionary robotics is how to minimize the difference between simulation results and results in the real world. Evolutionary robotics can be categorized by the method of evaluation, as depicted in Figure 13.

Most ALife researchers try to obtain synthetic forms of organization inspired by biological principles of life [23]. Evolutionary robotics is the attempt to develop robots through a self-organized process based on artificial evolution [24–27]. An initial population of strings, each encoding the control system (and sometimes the morphology) of a robot, is randomly created and put in the environment. Each robot is then allowed to act according to a genetically specified controller, and its performance on various tasks is automatically evaluated. The fittest robots are allowed to reproduce by generating copies of their genotypes with the addition of changes introduced by some genetic operators. This process is repeated until a satisfactory specimen is born. From an engineering point of view, the main advantage of relying on self-organization is that the designer does not need to divide the desired behavior into simple basic behaviors to be implemented in separate layers of the robot control system [28].

In fact, it has been demonstrated that training or evolving robots in the real world is possible, but the number of trials needed to test the system discourages the use of physical robots during the training period. Miglino et al. proposed an accurate model of particular robot-environment dynamics by continuing the evolutionary process in the real-world environment for a few generations [29]. Meanwhile, embodied evolution (EE) avoids the pitfalls of the simulate-and-transfer method and allows the speedup of evaluation by utilizing parallelism [30]. Mataric and Cliff focused on the problems of evolving controllers for physically embodied and embedded systems that deal with the noise and uncertainty present in the world [31]. Brooks proposed GP to evolve programs to control physically embodied mobile robots [32].

Many researchers think that evolving NNs is the most promising approach, for a number of reasons [33–35]. NNs can easily exploit various forms of learning during their lifetime, and this learning process may speed up the evolutionary process. The whole evolutionary process may not be implemented in the real world, due to the high time complexity of the simulation, which causes a serious gap between simulated and real environments. Kondo et al. tried to overcome the problem

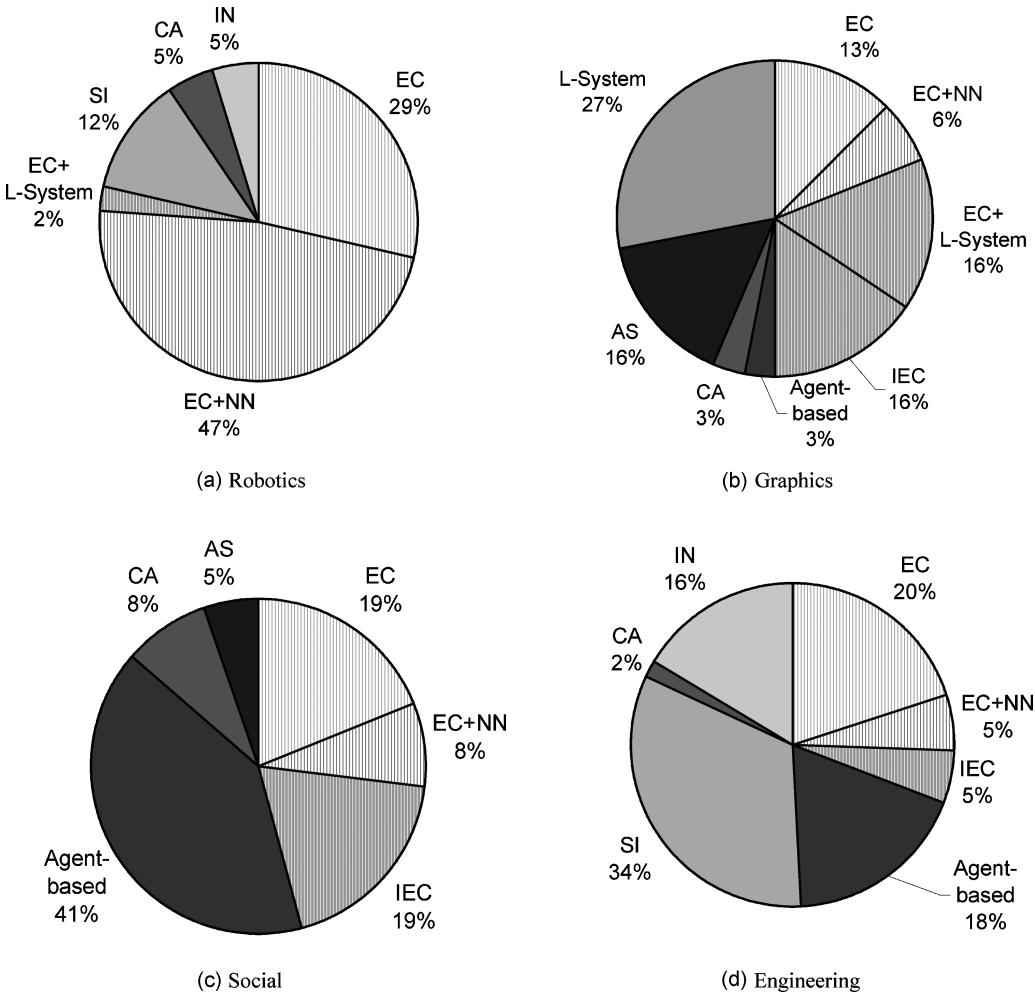

Figure 9. The proportion of each methodology for each category.

by incorporating the concept of a dynamic rearrangement function of biological neural networks into evolutionary robotics [36]. Floreano et al. employed a different approach where synaptic strengths are not genetically specified and adaptation during life consists of Hebbian synaptic changes. For each synapse, the genetic string encodes four Hebbian rules, a learning rate, the sign, and the postsynaptic effect of the traveling signal [37]. Nelson et al. proposed evolutionary training of artificial NN controllers for competitive team game-playing behaviors by teams of real mobile robots [38, 39]. Lee and Cho developed a fuzzy logic controller for a mobile robot with a genetic algorithm in simulation environments [40], which analyzes the behavior of the controller from the perspective of observational emergence [41] and evolvability [42].

Coevolution (that is, the evolution of two or more competing populations with coupled fitness) has several features that may potentially enhance the power of adaptation of artificial evolution [43–46]. By using a combination of commercial off-the-shelf CAD/CAM simulation software and physical simulators constrained to correspond to real physical devices, Pollack et al. have been developing technology for the coevolution of body and brain [47]. A hybrid GP-GA framework is presented to evolve complete robot systems, including controllers and bodies, to achieve fitness-specified tasks [48].

Collective behavior as demonstrated by social insects is a form of decentralized control that may prove useful in controlling multiple robots [49, 50]. In some species of ants, workers cooperate to

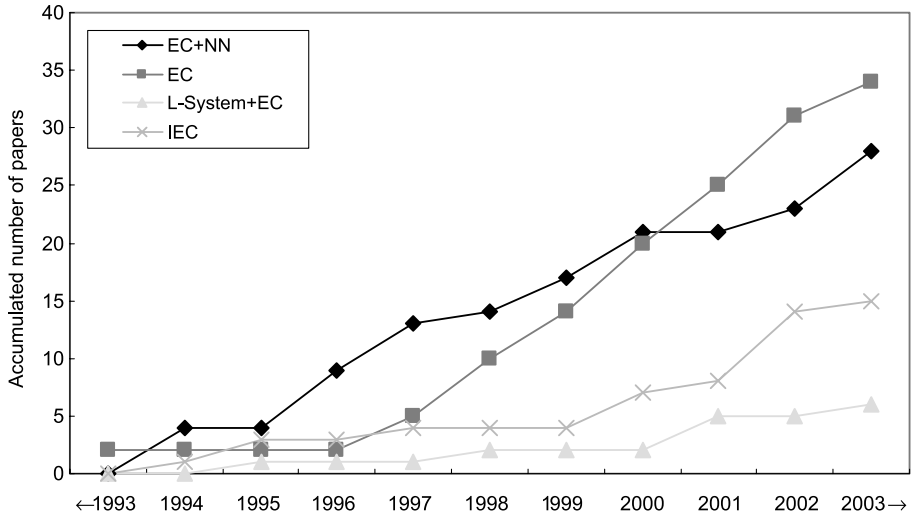

Figure 10. Statistics on application articles in artificial life by methodology and year: EC-related methods.

retrieve large prey. Usually, one ant finds prey, tries to move it, and, when unsuccessful, recruits friends through direct contact or chemical marking (pheromone). When a group of ants tries to move large prey, the ants change position and alignment until the prey can be moved toward the nest. A robotic implementation of this phenomenon is described in [51]. Groups of robots using ant-inspired algorithms of decentralized control techniques foraged more efficiently and maintained higher levels of group energy than single robots [52].

The CA model is a powerful instrument in many applications. Marchese proposed a reactive path-planning algorithm for a non-holonomic mobile robot on multilayered cellular automata [53]. CAM-Brain is a model to create neural networks based on CA, and finally aims at developing an artificial brain. The original CAM-Brain model has been modified to solve problems caused by evolving the model to control a mobile robot [54]. The biological immune system has features such as self-organization and learning ability. Based on these facts, researchers have been trying to construct engineering models of the immune system, and apply these to robotics [55, 56]. Meyer has

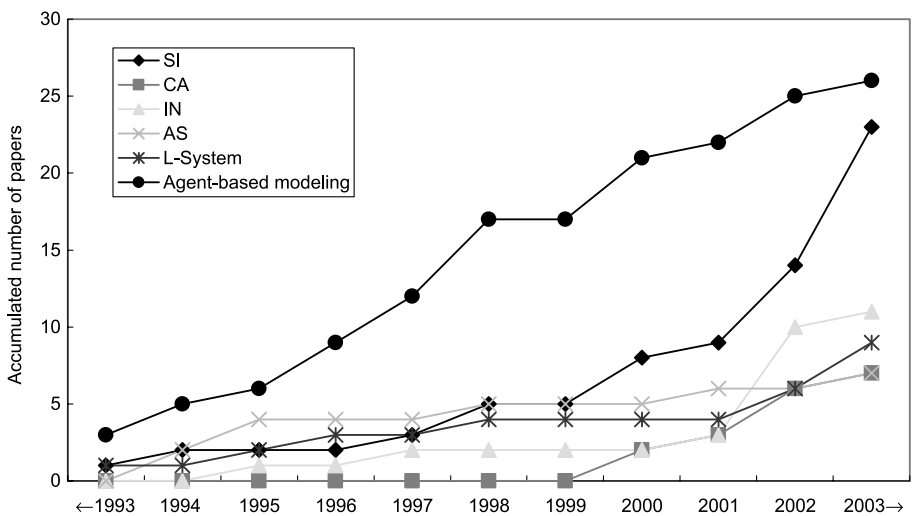

Figure 11. Statistics on application articles in artificial life by methodologies and year: methods other than EC.

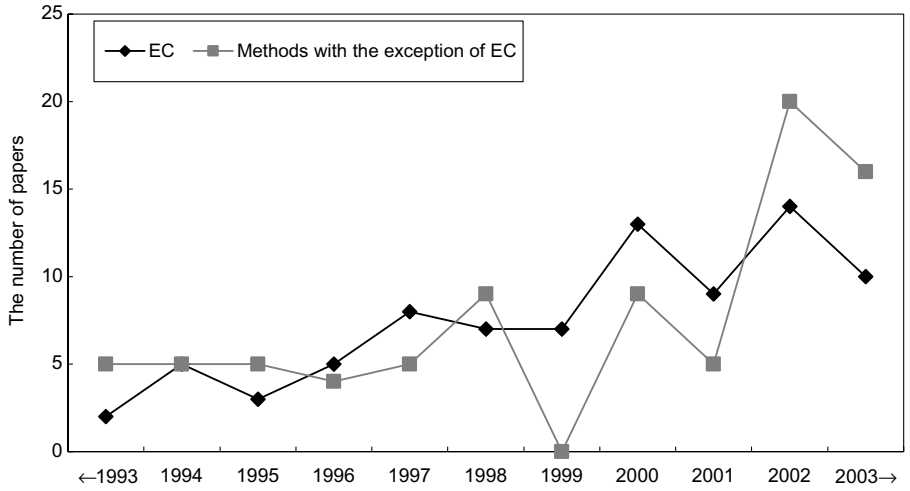

Figure 12. The number of application articles by methodologies and year.

proposed an integration of several specific mechanisms that underlie the adaptive behaviors of animals into a coherent function model using biological knowledge [57].

3.2 Robot Manufacturing: Practical Robot

Because most research in the robot control area is conducted in the laboratory, many unsolved problems have appeared in commercial application. The GOLEM project is the most interesting example of solving such problems in using a 3D solid printing tool for rapid manufacturing. The GOLEM project is an attempt to extend evolutionary techniques into the physical world by evolving diverse electromechanical machines (robots) that can be fabricated automatically [58–60]. To date, robots are still designed laboriously and constructed by teams of human engineers at great cost. Few robots are available, because these costs must be absorbed through mass production that is currently

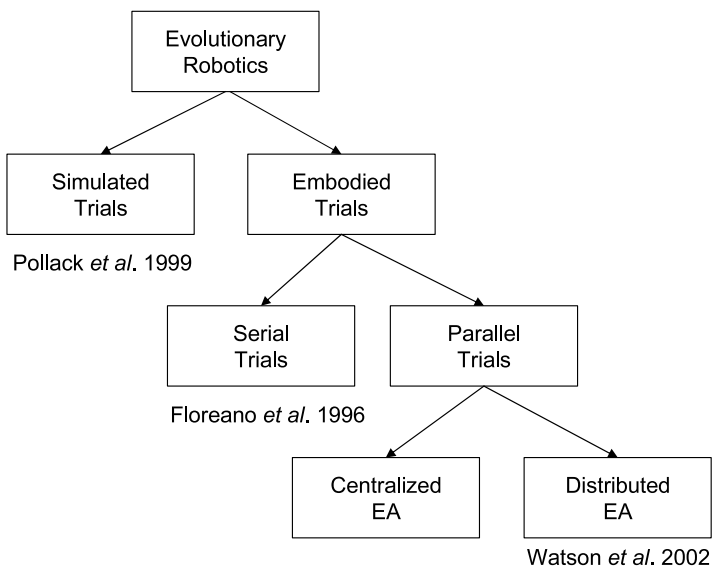

Figure 13. The categorization of evolutionary robotics. The basic idea is from [30].

justified only for toys, weapons, and industrial systems like automatic teller machines. Lipson and Pollack reported a set of experiments in which simple electromechanical systems evolve to yield physical locomoting machines [59]. They achieve autonomy of design and construction using evolution in a limited-universe physical simulation coupled to off-the-shelf rapid manufacturing technology. Using 3D solid printing, the evolved creatures then replicate automatically into reality, where they faithfully reproduce the performance of their virtual ancestors [60]. Also, they use a generative representation, which can reuse components in regular and hierarchical ways, providing a systematic way to create more complex modules from simpler ones [61].

Some researchers tackle more practical problems. It is still not clear if the evolutionary approach for building autonomous robots is adequate for real-life problems. A mobile robot has been successfully trained to keep clear an arena surrounded by walls by locating, recognizing, and grasping garbage objects and taking them outside the arena. The controller of the robot was evolved in simulation and then downloaded and tested on the real robot [62]. A decentralized method is proposed for controlling a homogeneous swarm of autonomous mobile robots that collectively transport a single palletized load [63]. Mondada et al. have proposed a new robotic concept, called SWARM-BOT, based on a swarm of small and simple autonomous mobile robots called S-BOTs [64]. S-BOTs have a particular assembling capability that allows them to connect physically to other S-BOTs and form a bigger robot entity, the SWARM-BOT. By taking advantage of the collective and distributed approaches, this concept achieves resistance to failure even in challenging environment conditions such as rough terrain.

3.3 Computer Graphics

This subsection deals with designing virtual characters, 2D image generation, and animation. Because these products are difficult to evaluate objectively, interactive evolutionary computation is used to design them. However, subjective evaluation by each user leads to practical problems such as fatigue, difficulty in maintaining large populations, and small numbers of generations. To solve these problems, Lee et al. proposed the direct manipulation method [65]. This allows the user to manipulate individuals directly instead of using evolutionary operators as an interface to each individual. In the interactive evolutionary system, the cost of evaluating each individual is relatively high, and it is difficult to maintain large populations. To solve this problem, Kim and Cho propose a hybrid genetic algorithm based on clustering, which considerably reduces the number of evaluations without any loss of performance [66]. The algorithm divides the whole population into several clusters and evaluates only one representative from each cluster. The fitness values of other individuals are estimated indirectly from the representative fitness values. Knowledge-based encoding includes removing impractical solutions, using partial information about the final solution, focusing on a specific search space, and encoding in a modular manner [67]. Figure 14 shows a

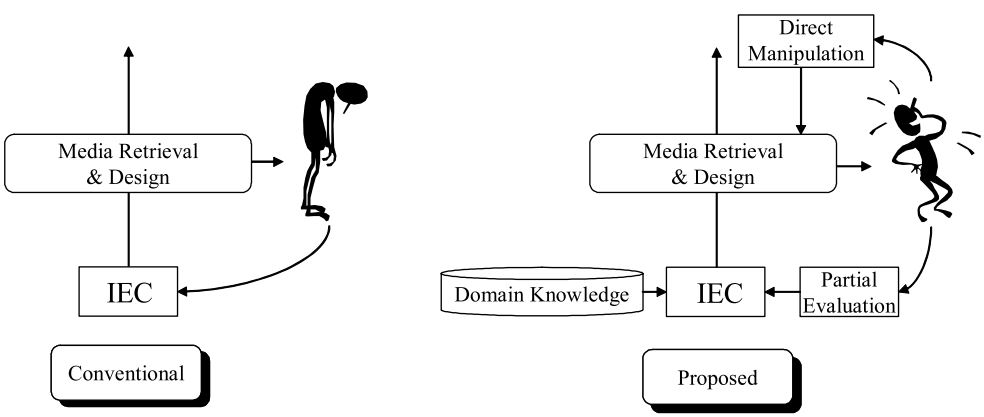

Figure 14. A framework for improving interactive evolutionary computation.

framework for improving interactive evolutionary computation. Cho and Lee have developed an image retrieval system based on human preference and emotion by using an interactive GA [83, 84]. It searches the images not only with explicit queries, but also with implicit queries such as "cheerful expression" and "gloomy expression." Gritz and Hahn used interactive genetic programming to automatically derive control programs for the agents [85, 86].

ALife concepts play a central role in the construction of advanced graphics models for image synthesis, animation, multimedia, and virtual reality [68]. Virtual creatures play an increasingly important role in computer graphics as special effects and background characters. The artificial evolution of such creatures potentially offers some relief from the difficult and time-consuming task of specifying morphologies and behaviors. Additionally, a hybrid of the L-system and evolution is widely used in computer graphics. Hornby and Pollack have proposed a system that uses L-systems to encode an EA for creating virtual creatures [69].

Karl Sims has developed a novel system for creating virtual creatures that move and behave in simulated 3D physical worlds. When simulated evolutions are performed with populations of competing creatures, interesting and diverse strategies and counter-strategies emerge [70, 71]. The work of Sims has been extended by many authors [74–76]. Teo and Abbass investigated the use of a self-adaptive Pareto evolutionary multi-objective approach for evolving the controllers of virtual embodied organisms [72] and the underlying fitness landscape of the controller's search space [73].

Gentropy is a genetic programming system that evolves into 2D procedural textures [80]. It synthesizes textures by combining mathematical and image manipulation functions into formulas. Most evolutionary art programs are interactive, and require the user to repeatedly choose the best images from a displayed generation. Gentropy uses an unsupervised approach, where one or more target texture images are supplied to the system.

Hornby and Pollack have demonstrated the ability of evolving L-systems to design tables [81]. Kato et al. proposed a novel method that enables automatic modeling of virtual cities [82]. The method makes use of the L-system to generate road networks, and the genetic algorithm to generate building layouts.

The tradeoff between control and automation is a well-known problem in computer-assisted character animation. It has been the goal of many researchers to allow the automation of agents while still allowing the developer to change the actions of the agents. Despite this, most professionals still use key framing, because the automatic motion control schemes have not allowed for sufficiently specific control over the agents. Miranda et al. have proposed evolving finite state machines for the behavioral control of characters [87]. Lim and Thalmann have argued that tinkering with existing models may be a more practical approach than starting from scratch in some applications of evolutionary techniques, and they validate this point with respect to setting parameters of a human-walk model for computer animation [88]. Animation through the numerical simulation of physics-based graphics models offers unsurpassed realism, but it can be computationally demanding.

While the dynamic nature of typical movie stunts makes them dangerous to perform, it also makes them attractive candidates for the application of physics-based animation. Faloutsos et al. created an autonomous virtual stuntman [77]. A comprehensive ALife approach to the realistic modeling of animals for virtual reality is emerging [78]. An extensive survey of synthetic actors can be found in [79]. Grzeszczuk et al. proposed the NeuroAnimator, a novel approach to creating physically realistic animation that exploits NNs [89]. A rich environment from which interesting strategies can be extracted by evolved creatures is needed, and Parisy and Schlick have proposed a simulation framework that can be used as a starting point for the development of dynamic behavioral systems [90].

3.4 Natural Phenomenon Modeling

This subsection deals with CA-based modeling, ecological modeling such as that of fish behavior, flock behavior modeling and its applications, and applications of L-systems. The usage of L-systems is relatively large in this area. It has been extended to agricultural modeling such as that of cotton and

plant-eating insects, and to explaining the evolution of fossils. Because Reynolds' flocking algorithm [5] generates realistic simulations of thousands of animals, it is frequently used in movies. A hybrid of the EA and the L-system, and a hybrid of the EA and Reynolds' flocking algorithm, are examples of synergism.

Simple CA models offer methods of incorporating computer modeling in the study of natural phenomena. Malamud and Turcotte proposed three different CA models: the forest fire, slider block, and sandpile models [91]. Each of these three models is associated with an important natural hazard that exhibits similar behavior. The forest-fire model is associated with actual forest fires and wildfires, the slider-block model with earthquakes, and the sandpile model with landslides. Computational plant models, or *virtual plants*, are increasingly seen as useful tools for comprehending complex relationships between gene function, plant physiology, plant development, and the resulting plant forms [92].

Artificial fish are autonomous agents whose appearance and complicated group interactions are as faithful as possible to nature's own. Terzopoulos et al. proposed a bottom-up, computational approach in which they modeled not just form and superficial appearance, but also the basic physics of the animal and its environment, the means of locomotion, the perception of its world, and its behavior [93, 94]. Stephens et al. created artificial fish for real-time interactive virtual worlds aimed at desktop environments with hardware 3D support [95]. The artificial fish can move, sense, and think. A high-performance computer simulation of Europe's largest aquarium lets its human visitors interact with 25 species and about 1,000 individual creatures and plants [96]. Miller has modeled legless figures such as snakes and worms as mass-spring systems [97].

The aggregate motion of a flock of birds, a herd of land animals, or a school of fish is a beautiful and familiar part of the natural world. But this type of complex motion was rarely seen in computer animation in 1987. Reynolds developed an approach based on simulation as an alternative to scripting the path of each bird individually [98]. The aggregate motion of the simulated flock is created by a distributed behavioral model; the birds choose their own courses. Behavioral animation techniques that are similar to Reynolds' have been used to create special effects in movies like *Batman Returns* (animated flocks of bats), *The Lion King* (herds of wildebeests), and *Mulan* (crowd scenes). Oboshi et al. [99] and Birt and Shaw [100] designed the mechanism of fish-school behavior using EC.

Production systems and L-grammars, as introduced by Prusinkiewicz and Lindenmayer, are very popular tools for creating images [101, 102]. The theory of L-systems has been mainly used for the visualization of the development and growth of living organisms like plants, trees, and cells. Jacob uses evolutionary algorithms for inferring L-systems encoding structures with specified characteristics [103, 104]. In the Wildwood project, genetic algorithms are applied to a simplified L-system representation in order to generate ALife-style plants for virtual worlds [105]. Chen et al. used L-systems for realistic modeling and rendering of feathers and birds [106]. Prusinkiewicz et al. proposed a combined discrete-continuous model of plant development that integrates L-system-style productions and differential equations [107]. L-Cotton, an architectural model, captures morphological development and the emergent pyramidal shape of a cotton plant by expressing the process using the L-system formalism [108].

Interaction with the environment is a key factor affecting the development of plants and plant ecosystems. Mech and Prusinkiewicz extended the formalism of L-systems with the constructs needed to model bidirectional information exchanges between plants and their environment [109]. Hanan et al. modeled insect movement and development of 3D structures of a plant, allowing physiological processes associated with responses to being damaged by insects [110].

The form of trace fossils results from complex interactions among the organism's morphology. Computer-based study of morphologies utilizes L-systems and related methods for the generation of branching and 3D theoretical morphologies [111].

3.5 Entertainment and Games

Designing AI of computer game players is tedious and knowledge-oriented work. ALife can reduce the worker's load by allowing an evolutionary strategy search and realistic modeling of human

behavior. Top-level computer programs that are developed by humans need expert knowledge to tune themselves, but a modern master-level checkers program can design its strategy by coevolving with other computer programs. Rule-based AI cannot provide users with unexpected behavior patterns, and thus have trouble keeping their interest. Basically, an evolutionary strategy search does not use much knowledge. It can also provide unexpected behavior patterns and continuously upgrade itself. Also, emergent properties of evolutionary algorithms allow game users to play creatively. From the perspective of optimization, evolutionary algorithms optimize multiple parameters of games. Recently, commercially available game software has used evolutionary algorithms for tuning the software's performance, and Evolutionary Checkers is now on sale.

The goal of building an autonomous agent is as old as the field of AI itself. The ALife community initiated a radically different approach to this goal, which focuses on fast, reactive behavior rather than on knowledge and reasoning, as well as on adaptation and learning. One potential application area of agent research that has received surprisingly little interest so far is entertainment [112]. Artificial Life Interactive Video Environment (ALIVE) is a virtual environment that allows wireless full-body interaction between a human participant and a virtual world inhabited by animated autonomous agents. One of the goals of the ALIVE project is to demonstrate that virtual environments can offer a more emotional and evocative experience by allowing the participant to interact with animated characters [113]. An evolutionary algorithm is used to evolve the gait of the Sony entertainment robot AIBO [114–116]. The aim of the MICROB project (Making Intelligent Collective Robotics) is to investigate collective phenomena of organization in societies of robots. The application of a robot soccer team was chosen to investigate the design of robots that can organize themselves to achieve a collective task [117].

ALife technologies offer a way to build a better and more believable environment for computer games [118, 119]. Flocking has found its way into some of the more recent game releases. Half-life and Unreal both use flocking to control the movement of groups of fish, birds, and monsters to create a more realistic and natural environment. Creatures makes heavy use of a combination of chemistry-based GAs and NNs to control virtual pets, which learn how to speak, feed themselves, and interact with the player in a variety of ways [120]. Oidian Systems' Cloak, Dagger, and DNA is another game that uses genetic algorithms to produce smarter opponents. Recently, coevolution has been used to evolve a NN (called Anaconda) that, when coupled with a min-max search, can evaluate checkerboard positions and play at the level of a human expert [121].

3.6 Music

Music is also difficult to evaluate objectively, so interactive evolutionary computation is frequently used. This subsection shows how to represent music as gene code and how to automate the subjective evaluation procedure. It examines an interactive evolutionary system without any automatic evaluation effort, a system with automated evaluation, and a CA-based music system. The difference with the interactive evolutionary system, as applied to music, is that there are some attempts to develop an automated evaluator. The key methodologies for ALife music are interactive evolutionary computation and cellular automata.

As with most problem-solving activities, musical tasks like composing, arranging, and improvising involve a great deal of searching. The evolutionary algorithm provides a powerful technique for searching large, often ill-behaved problem spaces. Interactive evolutionary computation, that is, evolutionary computation whose fitness function is provided by users, has been applied to music, art, and design. It is difficult to define the fitness function explicitly in these areas. GenJam is an interactive genetic algorithm-based model of a novice jazz musician learning to improvise [122]. Tokui and Iba proposed a new approach to music composition, more precisely the composition of rhythms, by means of interactive evolutionary computation [123]. The main feature of this method is combining GAs and GP. Unemi designed a support system for musical composition, named SBEAT3, based on simulated breeding [124, 125]. Each individual in the population is a short musical section of 16 beats, including 23 parts, 13 solos, 2 chords, and 8 percussions. Jacob used an

interactive genetic algorithm to produce a set of data filters that identify acceptable material from the output of a stochastic music generator [126]. Cho applied an interactive genetic algorithm to music information retrieval [127].

The Vox Populi system is designed to perform a series of sound experiments and eventually to be used as a tool for algorithmic composition [128]. Objectively defined criteria are used to evaluate the system's musical fitness, and a user may interfere in the fitness function through interface controls. Burton and Vladimirova proposed a means by which NN fitness evaluation can be applied to a GA that is applied to musical rhythm composition [129]. The GP-MUSIC system [130] is an interactive system that allows a user to evolve short musical sequences using interactive GP, and its extensions aim at making the system fully automated. Using this interactive technique, it was possible to generate pleasant tunes for runs of 20 individuals over 10 generations. As the user is the bottleneck in these kinds of interactive systems, the system takes rating data from a user's runs and uses it to train an NN-based automatic rater, or "auto rater," which can replace the user in bigger runs. De la Puente et al. describe how grammatical evolution may be applied to the domain of automatic composition [131]. Alander compiled an indexed bibliography of GAs in arts and music [132].

Miranda employed cyclic, self-organizing CAs to control a sound synthesizer and used a pattern-propagation CA to generate musical structures [133, 134]. Wada et al. proposed a novel method of producing a compressive and completely reproducible description of digital sound data by rule dynamics of CAs [135].

3.7 Economics

Decentralized market economies are complex adaptive systems, consisting of large numbers of adaptive agents involved in parallel local interactions. Agent-based computational economics (ACE) is the study of economies modeled as evolving systems of autonomous interacting agents [136–139]. Tassier and Menczer modeled a labor market that included referral networks using an agent-based simulation [140]. Agents could maximize their employment satisfaction by allocating resources to build friendship networks and to adjust search intensity. The multi-agent simulation system SWARM is employed to simulate and analyze economic concepts [141, 142]. Jennings et al. argued that agent technology can improve a company's efficiency by ensuring that businesses' activities are better scheduled, executed, monitored, and coordinated [143, 144]. Agent-based design and implementation philosophy has been used as a prototype for a business process management system for British Telecom (BT) [145].

Tsvetovatyy et al. proposed architecture for an agent-based virtual market that includes all the elements required to simulate a real market [146]. Kephart et al. investigated brokering agents' dynamic behavior, such as their price-setting mechanisms, in the context of a simple information filtering economy [147, 148]. An agent-based system simulates changes in predominant coordination mechanisms [149]. Inoue et al. optimized SCM (supply chain management) with HAL (hyper-artificial-life) middleware [150], which supports sequential GAs, asynchronous colony GAs, autonomous immigration GAs, and hyper-ALife algorithms for cluster computing.

3.8 Internet and Information Processing

ALife might come to play an important role in the World Wide Web, both as a source of new algorithmic paradigms and as a source of inspiration for its future development [151]. Menczer et al. [152] proposed a model, inspired by evolving agents, and applied it to the problem of retrieving information from a large, distributed collection of documents. By competing for relevant documents, agents robustly adapted to their environment and were allocated to efficiently exploit shared resources. Sheth and Maes evolved a population of personalized information filtering agents to make a personalized selection of Usenet news messages for a particular user [153]. If an adaptive pool of antibodies can produce "intelligent" behavior, the power of computation can tackle the problem of preference matching and recommendation [154, 155]. Chao and Forrest proposed an approach to building an information immune system that eliminates undesirable information before it reaches the

user [156]. An artificial immune system detects different user access patterns on Web sites [157]. Ibanez et al. propose applying emergent collective behavior ideas to automatically program Internet multi-channel radio stations [158]. The proposed model simulates *n* virtual DJs (one per channel) playing songs at the same time. Every virtual DJ takes into account the songs played by the other ones, programming a sequence of songs whose order is also coherent.

3.9 Industrial Design

This subsection is divided into two paragraphs, describing the colony-based approach and the others. Lucic and Teodorovic proposed applications of artificial bee and fuzzy ant systems in the field of traffic and transportation engineering [159]. The single-row machine layout problem concerns one of the most commonly used layout patterns, especially in flexible manufacturing systems. Solimanpur et al. used an ant algorithm to solve the problem [160]. Traffic congestion has become a major concern for many cities throughout the world. A colony-based traffic simulation system can provide helpful tools for engineers to plan traffic systems [161]. In order to solve the problem of the arrangement of letters on a computer keyboard, an algorithm based on ACO has been developed and applied [162]. Successful implementation of just-in-time (JIT) production systems is very important to modern manufacturing firms. McMullen used an ACO approach to solve multiple-objective JIT sequencing problems [163]. Batch processes have always been of considerable importance in chemical manufacturing due to their flexibility in processing multiple products and their ability to accommodate diverse operating conditions. Jayaraman et al. used the ACO approach for the optimal design of batch chemical processes [164]. Abbass et al. proposed a new technique for constructing programs through ACO using the tree adjunct grammar (TAG) formalism [165].

Kim and Cho developed a more realistic design aid system for women's clothes by incorporating domain-specific knowledge into an interactive genetic algorithm [166]. Nishino et al. adopted interactive evolutionary computation to create 3D geometric models [167]. Yang et al. used a new emergent colonization algorithm for optimum design of short bearings [168]. In the design of engine mounts, after specifying the amount of shock-absorber fluid, design parameters can be varied in order to obtain a desired notch frequency and notch depth. Ahn et al. used a hybrid of a conventional ALife algorithm with a random taboo search (R-taboo) algorithm to solve that problem [169].

3.10 Simulation Software

Echo is a model of complex adaptive systems formulated by John Holland. It eliminates virtually all of the physical details of real systems and concentrates on a small set of primitive agent-agent and agent-environment interactions [170–172]. LEE (Latent Energy Environment) is both an ALife model and a software tool to be used for simulations within the framework of that model [173, 174].

Yaeger developed a computer model of living organisms and ecology in PolyWorld [175], which attempts to bring together all the principal components of real living systems into a single artificial living system. It synthesizes genetics, physiologies and metabolisms, Hebbian learning in arbitrary neural networks, visual perception, and primitive behavior. PolyWorld can be downloaded from http://homepage.mac.com/larryy/larryy/PolyWorld.html.

There have been a number of ecology models based on discrete lattices, but the Gecko system uses no lattice [176]. Agents have free range in two dimensions, and they compete directly for space.

The MUTANT platform includes a model of a self-adaptive agent, with genetic evolving capabilities as learning mechanisms and a powerful graphical user interface providing many tools for both modeling and simulation [177]. Swarm is a multi-agent software platform for the simulation of complex adaptive systems. Swarm can be obtained from the Web page http://www.swarm.org [178].

3.11 Electronics

Contrary to conventional hardware, where the structure is irreversibly fixed in the design process, evolvable hardware (EHW) is designed to adapt to changes in task requirements or changes in the environment through its ability to reconfigure its own hardware structure dynamically and

autonomously [179, 180]. Higuchi et al. have introduced EHW chips and six applications: an analog EHW chip for cellular phones, a clock-timing architecture for gigahertz systems, an NN EHW chip capable of autonomous reconfiguration, a data compression EHW chip for printers, and a gate-level EHW chip [180–182]. Yao and Higuchi reviewed the status of EHW, discussed its promises and possible advantages, and presented challenges [183]. Zebulum et al. applied an evolutionary algorithm based on fixed and variable genotypes in the problem of passive analog filter design [184]. Two basic methodologies of evolutionary hardware are evolutionary computation and swarm intelligence.

Very large scale integration (VLSI) circuits, known as developmental explosions during the 1980s, are often too complex to be designed or analyzed globally. Hence, partitioning lies at the root of numerous computer-aided design problems. Kuntz et al. used an ant clustering algorithm to solve that problem [185]. Issacs et al. produced evolvable random number generators (RNGs) that can be written to hardware using a genetic algorithm to evolve ant colony systems [186].

As in nature, the space of bio-inspired hardware systems can be partitioned along three axes, phylogeny, ontogeny, and epigenesis (POE), giving rise to a new model. Stauffer et al. briefly presented three systems based on field-programmable gate arrays (FPGAs), each situated along a different axis of the POE model [190]. When searching for prey, many predator species exhibit remarkable behavior: after prey capture, the predators promptly engage in *area-restricted search*, probing for successive captures nearby. The strategy is effective for so many species because it strikes a good balance between the exploitation and the exploration of the search space. Linhares proposed synthesizing a similar search strategy for the gate matrix layout, an important problem in VLSI architecture [191]. The predatory search strategy restricts the search to a small area after each improved solution is found.

Mange et al. proposed the design of highly robust integrated circuits endowed with properties usually associated with the living world: self-repair and self-replication. The architecture is based on hierarchical levels of organization (molecule, cell, organism, and population) [192].

Scheuermann et al. [187] proposed a hardware implementation of population-based ACO (P-ACO) on FPGAs. They demonstrate that a straightforward hardware mapping of the standard ACO algorithm is not very well suited to implementation of the resources provided by current commercial FPGA architectures.

Particle swarm optimization is a robust stochastic evolutionary computation technique based on the movement and intelligence of swarms. Individual particles are accelerated toward the location of the best solution (the location in parameter space of the best fitness for the entire swarm) and the location of their own personal best (the location in parameter space of the best fitness for the individual). Recently, this technique has been successfully applied to antenna design [188]. A self-structuring antenna (SSA) can arrange itself into a large number of configurations. Coleman et al. investigated the use of ACO, simulated annealing, and genetic algorithms for finding suitable states of the SSA [189].

3.12 Security

Recent security incidents and analysis have demonstrated that manual response to such attacks is no longer feasible. Intrusion detection systems (IDSs) offer techniques for modeling and recognizing normal and abusive system behavior [193]. Current IDS solutions are overwhelmed by the burden of capturing and classifying new intrusion patterns. To overcome this problem, a self-adaptive distributed-agent-based defense immune system based on biological strategies was developed [194, 195]. Dasgupta and Gonzales proposed a technique inspired by the negative selection mechanism of the immune system that can detect foreign patterns in the complement space. In particular, the novel pattern detectors use a genetic search [196]. Aickelin et al. proposed ideas on creating the next generation IDS based on the latest immunological theories [197]. Immunologists are increasingly finding fault with traditional self-nonself thinking, and a new *danger theory* is emerging. This theory suggests that the immune system reacts to threats based on the correlation of various

(danger) signals, and it provides a method of *grounding* the immune response. The aim of their research is to investigate the correlation of these signals and to translate the theory into the realms of computer security.

There has been considerable interest in computer viruses since they first appeared in 1981. Scientists ask if computer viruses are not a form of ALife self-replicating organisms [198]. Similar to IDSs, other research efforts are aimed at creating computer-virus-immune systems [199–201].

ALife has been applied extensively in the area of defense. Artificial life techniques such as agent-based models and evolutionary algorithms provide a potentially powerful new approach to understanding some of the fundamental processes of combat [205]. Ilanchinski introduced two simple ALife-like "toy models" of land combat called ISAAC (Irreducible Semi-Autonomous Adaptive Combat) and EINSTein (Enhanced ISAAC Neural Simulation Toolkit) [202]. MANA (Map Aware Non-uniform Automata) is a CA-model of conflict that has been developed by the New Zealand Defence Technology Agency [203]. Yang et al. have attempted to understand the characteristics and properties of the search space of complex adaptive combat systems [204].

3.13 Data Mining

Ngan et al. introduced a system for discovering medical knowledge by learning Bayesian networks and rules. Evolutionary computation is used as the search algorithm. The system is applied to real-life medical databases for limb fracture and scoliosis [206]. Pena-Reyes and Shipper surveyed how evolutionary algorithms can be applied to solve medical problems, including diagnosis, prognosis, imaging, signal processing, planning, and scheduling, with an extensive bibliography [207].

The prediction of the 3D structure of proteins is a great challenge both for the difficulty of the task and for the importance of the problem. Many researchers have tried to predict the structure on the sole basis of the amino acid sequence. Calabretta et al. used a genetic algorithm to obtain appropriate matrices of folding potentials, that is, "forces" that drive the folding process to produce correct tertiary structures [208].

Inferring a gene regulatory network is one of the challenging topics in the field of bioinformatics. In order to infer network structure effectively, a new approach allows human intervention and strategic data acquisition in the inference process. Iba and Mimura used interactive EC for inferring gene regulatory networks using gene expression data from DNA microarrays [209]. Kim and Cho have proposed an evolutionary neural network that classifies gene expression profiles into normal and colon cancer cells [210].

Parpinelli et al. proposed an ant colony-based algorithm for data mining called Ant-Miner. The goal of Ant-Miner is to extract classification rules from data [211]. They compared the performance of Ant-Miner with CN2, a well-known data mining algorithm for classification, in six public domain data sets. Shelokar et al. used Ant-Miner to classify chemical process data sets [212]. Liu et al. presented an improvement to the Ant-Miner [213]. Tsai et al. reported a novel clustering method (described as ACO with a different favor algorithm), which performs better than the fast SOM + *K*-means approach and the genetic *K*-means algorithm [215]. Abraham and Ramos applied ant colony clustering to Web usage mining [214]. Lu et al. applied particle swarm optimization to train the multi-layer perceptrons and to predict pollutant levels (air quality) in Hong Kong [216].

3.14 Telecommunication

ACO is capable of solving various routing and congestion problems in computer networking by continuously modifying routing tables in response to congestion [217–220]. Denby and Hegarat-Mascle developed an application of an algorithm that mimics the behavior of ants to routing in a satellite-based telecommunications network [221].

Location management is a very important and complex problem in today's mobile environments. Subrata and Zomaya compared several well-known ALife techniques to gauge their suitability for solving location management problems [222] and used CAs combined with genetic algorithms to create an evolving parallel-reporting-cells planning algorithm [223].

4 Conclusion

Results of ALife applications cannot be easily evaluated, because their performance is difficult to measure objectively. Users' impressions in applications are the only criterion of success. However, Evolutionary Checkers can be evaluated using traditional ranking methods and shows master-level performance; it can defeat most people on the online game site. Successful applications of ALife have common properties, which can be summarized as follows:

1. *Achieving similar behavior to biological creatures.* A reason why some applications receive great interest from the public is that they show similar behavior to biological creatures. For example, Karl Sims evolved interesting locomotion movements similar to those of animals. These movements are difficult to invent or build manually [70]. As noted by Sims, the creation of virtual actors, whether animal, human, or alien, may be limited mainly by our ability to design them, rather than our ability to satisfy their computational requirements. In [60], some individuals (simple electromechanical systems) used sliding articulated components to produce crablike sideways motions. In [5], the animations showing simulated flocks built from Reynolds' model seem to correspond to the observer's intuitive notion of what constitutes flocklike motion.
2. *The details of the final results have not been described before experimentation.* In traditional engineering approaches, the final results of applications can be described in advance. By depending on emergent properties, some applications achieve great success. A genetic language that uses directed graphs to describe both morphology and behavior defines an unlimited hyperspace of possible creatures, and a variety of interesting virtual creatures have been shown to emerge [70].
3. *Designing without expert knowledge.* AI has relied mainly on capturing human expertise in the form of knowledge or programmed rules of behavior that emulate the behavior of the human expert. Recently, coevolution has been used to evolve a neural network (called Anaconda) that, when coupled with a minimax search, can evaluate checkerboard positions and play to the level of a human expert, as indicated by its rating of 2045 on an international Web site for playing checkers. The neural network uses only the location, type, and number of pieces on the board as input [121].
4. *Interdisciplinary cooperation.* ALife is an interdisciplinary study of life and lifelike processes that use a synthetic methodology [1]. Swarm Development Group (SDG) members have different backgrounds, including systems modeling, oceanography, geography, mathematics, computer science, economics, chemistry, political science, anthropology, environmental engineering, and ecology. The SDG was founded in 1999 as a private, not-for-profit organization to support the development of the Swarm simulation system. They use multi-agent-based simulation to advance the Swarm simulation software [178]. The group's success is based on the diverse backgrounds of its members.
5. *Huge computational requirements.* The evolutionary approach requires huge computational power. For example, it takes 4 hours to complete a simulation on a 32-processor CM-5 (Thinking Machines supercomputer) [70]. One hundred generations on a 400-MHz Pentium II require about two days in evolutionary checkers [121].
6. *Evolution based on simple primitive shapes such as box and pipe.* In successful applications of ALife, the primitive shapes of a physical body are the simple box and the pipe. Karl Sims used hexahedron boxes [70], and Pollack and Lipson constructed complex robot morphologies using a linear actuator, bar, and ball joint [60]. Though they have built relatively simple morphologies, they have shown the possibility of an ALife approach to real-world applications.
7. *Computer simulation.* Most of the successful results use computer simulations [5, 60, 70, 121, 178], but there are a few projects based on real hardware platforms. The latter suffer from very long evolution time, high cost, and uncertainty. In contrast, computer simulation can minimize the required time and cost by ignoring the details of the real world. With the growth of virtual worlds such as the Internet, the influence of computer simulation on users' lifestyles will increase.

Because ALife focuses more on synthesizing life than on analyzing it, there are applications in a wide variety of fields. Usually, an EA is exploited, but some other new methodologies, including ACO, particle swarm optimization, CAs, and artificial immune networks, can be alternatives. In computer graphics, the L-system is the dominant method for generating realistic images, and in industrial design, ACO is the dominant method because it is easily applicable to real-world design problems. Recently, the use of ACO and artificial immune networks has been growing rapidly because there is so much interest from the academic community. Special issues have been devoted to ACO, artificial-immune-network-based intrusion detection systems, and agent-based modeling for economics by *IEEE Transactions on Evolutionary Computation* [224–226]. Also, there have been several international conferences on ACO and the artificial immune system.

ALife techniques have many possibilities, because they can provide methods for generating complex situations with simple rules. They show interesting results in movies, computer graphics, robotics, and games. Of course, many applications are developed in engineering areas, but the most successful ones are the those that are evaluated only by humans. This means that surprise, emotional aspects, naturalism, creativity, and emergence are the key points for the success of the applications in the domain of ALife. Showing humanlike natural characteristics is a main criterion for the evaluation of applications. In this they are very different from traditional applications.

Most applications are still at the laboratory level, but a few of them are already being used in the real world. Although the emphasis of ALife is on the scientific discovery of the meaning of life, the outcomes of the research can be fruitful in the real world. ALife provides much potential for generating real-world applications by reducing tedious human effort. Some keywords of future applications are "user-friendly," "adaptive," "humanlike," "personalized," "emotional," and "entertaining." Traditional methodologies based on mathematics can make good solutions for real-world problems. However, people want more advanced or complicated applications that show interesting and optimized behavior. Nature and life show complex behaviors that are not easily explainable, but humans provide and analyze solutions for nature modeling. Although they are not optimized in some situations, they can be adaptive and provide joy to people by complementing the traditional methodologies. From this perspective, a hybrid of ALife methodologies and traditional optimization methods may be used in future application design techniques.

Acknowledgments

This research was supported by Brain Science and Engineering Research Program sponsored by the Korean Ministry of Commerce, Industry and Energy.

References

1. Bedau, M. A. (2003). Artificial life: Organization, adaptation and complexity from the bottom up. *Trends in Cognitive Sciences*, *7*(11), 505–512.
2. Haykin, S. (1998). *Neural networks: A comprehensive foundation* (2nd ed.). Upper Saddle River, NJ: Prentice-Hall.
3. Ferber, J. (1999). *Multi-agent systems: An introduction to distributed artificial intelligence*. Redwood City, CA: Addison-Wesley.
4. Goldberg, D. E. (1989). *Genetic algorithms in search, optimization, and machine learning*. Redwood City, CA: Addison-Wesley.
5. Reynolds, C. W. (1987). Flocks, herds, and schools: A distributed behavioral model. *Computer Graphics*, *21*, 25–34.
6. Berlekamp, E. R., Conway, J. H., & Guy, R. K. (1982). *Winning ways for your mathematical plays*. New York: Elsevier.
7. Gajardo, A., Moreira, A., & Goles, E. (2002). Complexity of Langton's ant. *Discrete Applied Mathematics*, *117*(1–3), 41–50.

8. Yao, X. (1996). An overview of evolutionary computation. *Chinese Journal of Advanced Software Research*, *3*(1), 12–29.

9. Takagi, H. (2001). Interactive evolutionary computation: Fusion of the capabilities of EC optimization and human evaluation. *Proceedings of the IEEE*, *89*(9), 1275–1296.

10. Green, D. G. (1993). Cellular automata. http://physics.hallym.ac.kr/education/chaos/green/tutorial1.html.

11. Gutowitz, H. (Ed.) (1991). Cellular automata: Theory and experiment. *Physica D*, *45*, 1–3.

12. Toffoli, T., & Margolus, N. (1987). *Cellular automata machines: A new environment for modeling.* Cambridge, MA: MIT Press.

13. Tarasewich, P., & McMullen, P. R. (2002). Swarm intelligence. *Communications of the ACM*, *45*(8), 62–67.

14. Teo, J., & Abbass, H. A. (2003). A true annealing approach to the marriage in honey-bees optimization algorithm. *The International Journal of Computational Intelligence and Applications*, *3*(2), 199–208.

15. Abbass, H. A. (2002). An agent based approach to 3-SAT using marriage in honey-bees optimization. *International Journal of Knowledge-based Intelligent Systems*, *6*(2), 1–8.

16. Abbass, H. A. (2001). A single queen single worker honey bees approach to 3-SAT. In H. Beyer et al. (Eds.), *Proceedings of Genetic Evolutionary Computation Conference.* San Mateo, CA: Morgan Kaufmann.

17. Abbass, H. A. (2001). Marriage in honey bees optimization: A haplometrosis polygynous swarming approach. In *IEEE Congress on Evolutionary Computation*, (pp. 207–214).

18. Langton, C. (1989). *Artificial life.* Redwood City, CA: Addison-Wesley.

19. Vaario, J. (1993). *Artificial life primer* (Technical report TR-H-033). Kyoto: ATR Human Information Processing Research Labs.

20. Bonabeau, E., Dorigo, M., & Theraulaz, G. (2000). Inspiration for optimization from social insect behavior. *Nature*, *406*, 39–42.

21. Sim, K.-M., & Sun, W. H. (2003). Ant colony optimization for routing and load-balancing: Survey and new directions. *IEEE Transactions on Systems, Man and Cybernetics—Part A*, *33*(5), 560–572.

22. Dasgupta, D., & Attoh-Okine, N. (1997). Immunity-based systems: A survey. In *IEEE International Conference on Systems, Man and Cybernetics*, Vol. 1 (pp. 369–374).

23. Michel, O. (1996). An artificial life approach for the synthesis of autonomous agents. In J.-M. Alliot, E. Lutton, E. Ronald, M. Schoenauer, & D. Snyers (Eds.), *Artificial evolution* (pp. 220–231). New York: Springer.

24. Nolfi, S., & Floreano, D. (2002). Synthesis of autonomous robots through evolution. *Trends in Cognitive Sciences*, *6*(1), 31–37.

25. Nolfi, S., & Floreano, D. (2000). *Evolutionary robotics: The biology, intelligence, and technology of self-organizing machines.* Cambridge, MA: MIT Press.

26. Harvey, I., Husbands, P., Cliff, D., Thompson, A., & Jakobi, N. (1996). Evolutionary robotics at Sussex. In M. Jamshidi, F. Pin, & P. Dauchez (Eds.), *International Symposium on Robotics and Manufacturing* (pp. 293–298). New York: ASME Press.

27. Harvey, I. (1997). Artificial evolution and real robots. *Artificial Life and Robotics*, *1*, 35–38.

28. Nolfi, S. (1998). Evolutionary robotics: Exploiting the full power of self-organization. *Connection Science*, *10*, 167–183.

29. Miglino, O., Lund, H. H., & Nolfi, S. (1996). Evolving mobile robots in simulated and real environments. *Artificial Life*, *2*(3), 417–434.

30. Watson, R. A., Ficici, S. G., & Pollack, J. B. (2002). Embodied evolution: Distributing an evolutionary algorithm in a population of robots. *Robotics and Autonomous Systems*, *39*, 1–18.

31. Mataric, M. J., & Cliff, D. (1996). Challenges in evolving controllers for physical robots. *Robotics and Autonomous Systems*, *19*(1), 67–83.

32. Brooks, R. (1992). Artificial life and real robots. In *Proceedings of the First European Conference on Artificial Life* (pp. 3–10). Cambridge, MA: MIT Press.

33. Nolfi, S., Floreano, D., Miglino, O., & Mondada, F. (1994). How to evolve autonomous robots: Different approaches in evolutionary robotics. In R. Brooks & P. Maes (Eds.), *Proceeding of Artificial Life IV* (pp. 190–197). Cambridge, MA: MIT Press.

34. Floreano, D., & Mondad, F. (1998). Evolutionary neurocontrollers for autonomous mobile robots. *Neural Networks*, *11*(7–8), 1461–1478.

35. Floreano, D., & Mondad, F. (1996). Evolution of homing navigation in a real mobile robot. *IEEE Transactions on Systems Man and Cybernetics—Part B*, *26*, 396–407.

36. Kondo, T., Ishiguro, A., Tokura, S., Uchikawa, Y., & Eggenberger, P. (1999). Realization of robust controllers in evolutionary robotics: A dynamically-rearranging neural network approach. In *Proceedings of the 1999 Congress on Evolutionary Computation* (pp. 366–373). Piscataway, NJ: IEEE.

37. Floreano, D., & Urzelai, J. (1999). Evolution of neural controllers with adaptive synapses and compact genetic encoding. In D. Floreano, et al. (Eds.), *Advances in Artificial Life—ECAL99* (pp. 183–194). New York: Springer.

38. Nelson, A. L., Grant, E., & Henderson, T. C. (2004). Evolution of neural controllers for competitive game playing with teams of mobile robots. *Robotics and Autonomous Systems*, *46*, 135–150.

39. Nelson, A. L., Grant, E., Galeotti, J. M., & Rhody, S. (2004). Maze exploration behaviors using an integrated evolutionary robotics environment. *Robotics and Autonomous Systems*, *46*, 159–173.

40. Lee, S.-I., & Cho, S.-B. (2001). Emergent behaviors of a fuzzy sensory-motor controller evolved by genetic algorithm. *IEEE Transactions on Systems, Man and Cybernetics (B)*, *31*(6), 919–929.

41. Lee, S.-I., & Cho, S.-B. (2001). Observational emergence of a fuzzy logic controller evolved by genetic algorithm. In *Proceedings of the 2001 Congress on Evolutionary Computation,* Vol. 2 (pp. 1047–1054). Piscataway, NJ: IEEE.

42. Lee, S.-I., & Cho, S.-B. (2003). A quantitative analysis of evolvability for an evolutionary fuzzy logic controller. *Integrated Computer-Aided Engineering*, *10*(4), 369–385.

43. Nolfi, S., & Floreano, D. (1998). Co-evolving predator and prey robots: Do "arms races" arise in artificial evolution? *Artificial Life*, *4*(4), 311–335.

44. Dolin, B., Bennett, F. H., III, & Rieffel, E. G. (2002). Co-evolving an effective fitness sample: Experiments in symbolic regression and distributed robot control. In *Proceedings of the 2002 ACM Symposium on Applied Computing* (pp. 553–559).

45. Floreano, D., & Nolfi, S. (1997). Adaptive behavior in competing co-evolving species. In P. Husbands & I. Harvey (Eds.), *Proceedings of the 4th European Conference on Artificial Life* (pp. 378–387). Cambridge, MA: MIT Press.

46. Brooks, R. (2000). Artificial life: From robot dreams to reality. *Nature*, *406*, 945–947.

47. Pollack, J., Lipson, H., Funes, P., Ficici, S., & Hornby, G. (1999). Coevolutionary robotics. In *Proceedings of the First NASA/DOD Workshop on Evolvable Hardware* (pp. 208–216).

48. Lee, W.-P. (2000). Evolving autonomous robot: From controller to morphology. *IEICE Transactions on Information and System*, *E83-D*(2), 200–210.

49. Kube, C. R., & Zhang, H. (1994). Collective robotics: From social insects to robots. *Adaptive Behavior*, *2*, 189–218.

50. Lee, M. (2003). Evolution of behaviors in autonomous robot using artificial neural network and genetic algorithm. *Information Sciences*, *155*(1–2), 43–60.

51. Kube, C. R., & Bonabeau, E. (2000). Cooperative transport by ants and robots. *Robotics and Autonomous Systems*, *30*, 85–101.

52. Krieger, M. J., Billerter, J., & Keller, L. (2000). Ant-like task allocation and recruitment in cooperative robots. *Nature*, *406*, 992–995.

53. Marchese, F. M. (2002). A directional diffusion algorithm on cellular automata for robot path-planning. *Future Generation Computer Systems*, *18*(7), 983–994.

54. Cho, S.-B., & Song, G.-B. (2000). Evolving CAM-brain to control a mobile robot. *Applied Mathematics and Computation*, *111*, 147–162.

55. Ishiguro, A., Kuboshiki, S., Ichikawa, S., & Uchikawa, Y. (1995). Gait coordination of hexapod

walking robots using mutual-coupled immune networks. In *IEEE International Conference on Evolutionary Computation*, Vol. 2 (pp. 672–677).

56. Meshref, H., & Valandingham, H. (2001). Immune network simulation of reactive control of a robot arm manipulator. In *Proceedings of the 2001 IEEE Mountain Workshop on Soft Computing in Industrial Applications* (pp. 81–85).
57. Meyer, J.-A. (1997). From natural to artificial life: Biomimetic mechanisms in animat designs. *Robotics and Autonomous Systems*, *22*, 3–21.
58. Forbes, N. (2000). Life as it could be: ALife attempts to simulate evolution. *IEEE Intelligent Systems, Nov.–Dec.*, 2–7.
59. Lipson, H., & Pollack, J. (2000). Automatic design and manufacture of robotic lifeforms. *Nature*, *406*(6799), 974–978.
60. Pollack, J. B., & Lipson, H. (2000). The GOLEM project: Evolving hardware bodies and brains. In *Proceedings of The Second NASA/DoD Workshop on Evolvable Hardware*, (pp. 37–42).
61. Hornby, G. S., Lipson, H., & Pollack, J. B. (2003). Generative representation for the automated design of modular physical robots. *IEEE Transactions on Robotics and Automation*, *19*(4), 703–719.
62. Nolfi, S. (1997). Evolving non-trivial behaviors on real robots: A garbage collecting robot. *Robotics and Autonomous System*, *22*, 187–198.
63. Stilwell, D. J., & Bay, J. S. (1993). Toward the development of a material transport system using swarms of ant-like robots. In *IEEE International Conference on Robotics and Automation*, (pp. 766–771).
64. Mondada, F., Guignard, A., Colot, A., Floreano, D., Deneubourg, J.-L., Gambardella, L. M., Nolfi, S., & Dorigo, M. (2002). *SWARM-BOT: A new concept of robust all-terrain mobile robotic system* (Technical report LSA2-I2S-STI). Lausanne: Swiss Federal Institute of Technology.
65. Lee, J.-H., Kim, H.-S., & Cho, S.-B. (2001). Accelerating evolution by direct manipulation for interactive fashion design. In *Fourth International Conference on Computational Intelligence and Multimedia Applications* (pp. 362–366). Piscataway, NJ: IEEE.
66. Kim, H.-S., & Cho, S.-B. (2001). An efficient genetic algorithm with less fitness evaluation by clustering. In *Proceedings of the 2001 Congress on Evolutionary Computation,* Vol. 2, (pp. 887–894).
67. Kim, H.-S., & Cho, S.-B. (2000). Genetic algorithm with knowledge-based encoding. In *Pacific-Rim International Conference on Artificial Intelligence*, (p. 829).
68. Terzopoulos, D. (1999). Artificial life for computer graphics. *Communications of the ACM*, *42*(8), 32–42.
69. Hornby, G. S., & Pollack, J. B. (2001). Evolving L-systems to generate virtual creatures. *Computers & Graphics*, *25*, 1041–1048.
70. Sims, K. (1994). Evolving 3D morphology and behavior by competition. In *Proceedings of Artificial Life IV* (pp. 28–39). Cambridge, MA: MIT Press.
71. Sims, K. (1994). Evolving virtual creatures. In *Computer Graphics (Siggraph '94) Annual Conference Proceedings* (pp. 43–50).
72. Teo, J., & Abbass, H. A. (2004). Automatic generation of controllers for embodied legged organisms: A Pareto evolutionary multi-objective approach. *Evolutionary Computation*, *12*(3), 355–394.
73. Teo, J., & Abbass, H. A. (2004). Information-theoretic landscape analysis of neuro-controlled embodied organisms. *Neural Computing and Applications*, *31*(1), 80–89.
74. Pachepsky, E., Taylor, T., & Jones, S. (2002). Mutualism promotes diversity and stability in a simple artificial ecosystem. *Artificial Life*, *8*(1), 5–24.
75. Taylor, T. (2000). Artificial life techniques for generating controllers for physically modeled characters. In *Proceedings of the First International Conference on Intelligent Games and Simulation (GAME-ON 2000).*
76. Taylor, T., & Massey, C. (2001). Recent developments in the evolution of morphologies and controllers for physically simulated creatures. *Artificial Life*, *7*(1), 77–87.
77. Faloutsos, P., van de Panne, M., & Terzopoulos, D. (2001). The virtual stuntman: Dynamic characters with a repertoire of autonomous motor skills. *Computers & Graphics*, *25*(6), 933–953.
78. Heleno, P., & dos Santos, M. P. (1998). Artificial animals in virtual ecosystems. *Computer Networks and ISDN Systems*, *30*, 1923–1932.

79. Pina, A., Cerezo, E., & Seron, F. J. (2000). Computer animation: From avatars to unrestricted autonomous actors (A survey on replication and modeling mechanisms). *Computers & Graphics*, *24*, 297–311.

80. Weins, A. L., & Ross, B. J. (2002). Gentropy: Evolving 2D textures. *Computers & Graphics*, *26*(1), 75–88.

81. Hornby, G. S., & Pollack, J. B. (2001). The advantages of generative grammatical encodings for physical design. In *Proceedings of the 2002 Congress on Evolutionary Computation,* Vol. 1, (pp. 600–607).

82. Kato, N., Okuno, T., Okano, A., Kanoh, H., & Nishihara, S. (1998). An ALife approach to modeling virtual cities. In *IEEE International Conference on Systems, Man, and Cybernetics,* Vol. 2, (pp. 1168–1173).

83. Cho, S.-B., & Lee, J.-Y. (2002). A human-oriented image retrieval system using interactive genetic algorithm. *IEEE Transactions on Systems, Man and Cybernetics Part A*, *32*(3), 452–458.

84. Cho, S.-B (2002). Towards creative evolutionary systems with interactive genetic algorithm. *Applied Intelligence*, *16*(2), 129–138.

85. Gritz, L., & Hahn, J. K. (1995). Genetic programming for articulated figure motion. *Journal of Visualization and Computer Animation*, *6*(3), 129–142.

86. Gritz, L., & Hahn, J. K. (1997). Genetic programming evolution of controllers for 3-D character animation. In K. Deb et al. (Eds.), *Genetic Programming 1997: Proceedings of the Second Annual Conference* (pp. 139–146). San Mateo, CA: Morgan Kaufmann.

87. Miranda, F. R., Jr., J. E. K., Hernandez, E. D. M., & Netto, M. L. (2001). An artificial life approach for the animation of cognitive characters. *Computers & Graphics*, *25*, 955–964.

88. Lim, I. S., & Thalmann, D. (2000). Solve customers' problems: Interactive evolution for tinkering with computer animation. In *Proceedings of the 2000 ACM Symposium on Applied Computing*, (pp. 404–407).

89. Grzeszczuk, R., Terzopoulos, D., & Hinton, G. (1998). NeuroAnimator: Fast neural network emulation and control of physics-based models. In *Proceedings of the ACM SIGGRAPH 98 Conference*, (pp. 9–20).

90. Parisy, O., & Schlick, C. (2002). A physically realistic framework for the generation of high-level animation controllers. In *Proceedings of the 2nd International Symposium on Smart Graphics* (pp. 47–54). New York: ACM Press.

91. Malamud, B. D., & Turcotte, D. L. (2000). Cellular-automata models applied to natural hazards. *Computing in Science & Engineering*, *2*(3), 42–51.

92. Prusinkiewicz, P. (2004). Modeling plant growth and development. *Current Opinion in Plant Biology*, *7*(1), 79–83.

93. Tu, X., & Terzopoulos, D. (1994). Artificial fishes: Physics, locomotion, perception, behavior. In *Proceedings of ACM SIGGRAPH'94* (pp. 43–50).

94. Terzopolos, D., Tu, X., & Grzeszczuk, R. (1994). Artificial fishes: Autonomous locomotion, perception, behavior, and learning in a simulated physical world. *Artificial Life*, *1*(4), 327–351.

95. Stephens, K., Pham, B., & Wardhani, A. (2003). Modelling fish behaviour. In *Proceedings of the 1st International Conference on Computer Graphics and Interactive Techniques in Australasia and South East Asia* (pp. 71–78). ANZGRAPH.

96. Frohlich, T. (2000). The virtual oceanarium. *Communications of the ACM*, *43*(7), 94–101.

97. Miller, G. S. P. (1988). The motion dynamics of snakes and worms. *Computer Graphics*, *22*(4), 169–173.

98. Reynolds, C. W. (1987). Flocks, herds, and schools: A distributed behavioral model. *Computer Graphics*, *21*(4), 25–34.

99. Oboshi, T., Kato, S., Mutoh, A., & Itoh, H. (2002). Collective or scattering: Evolving schooling behaviors to escape from predator. In *Artificial Life VIII*, (pp. 386–389). ISAL.

100. Birt, A., & Shaw, S. (2003). Evolving boids: Incorporating machine learning into artificial life. *The Journal of Computing in Small Colleges*, *18*(5), 290–291.

101. Noser, H., & Thalmann, D. (1994). Simulating the life of virtual plants, fishes and butterflies. In N. Magnenat-Thalmann & D. Thalmann (Eds.), *Artificial Life and Virtual Reality* (pp. 45–60). New York: Wiley.

102. Deussen, O., Hanrahan, P., Pharr, M., Lintermann, B., Mech, R., & Prusinkiewicz, P. (1998). Realistic modeling and rendering of plant ecosystems. In *Proceedings of SIGGRAPH '98* (pp. 275–286). New York: ACM Press.

103. Jacob, C. (1995). Genetic L-system programming: Breeding and evolving artificial flowers with Mathematica. In *Proceedings of the First International Mathematica Symposium*, (pp. 215–222).

104. Jacob, C. (2001). *Illustrating evolutionary computation with Mathematica*. San Mateo, CA: Morgan Kaufmann.

105. Mock, K. J. (1998). Wildwood: The evolution of L-system plants for virtual environments. In *The 1998 IEEE International Conference on Evolutionary Computation Proceedings* (pp. 476–480).

106. Chen, Y., Xu, Y., Guo, B., & Shun, H.-Y. (2002). Modeling and rendering of realistic feathers. In *Proceedings of the 29th Annual Conference on Computer Graphics and Interactive Techniques* (pp. 630–636). ACM.

107. Prusinkiewicz, P., Hammel, M. S., & Mjolsness, E. (1993). Animation of plant development. In *Proceedings of the 20th Annual Conference on Computer Graphics and Interactive Techniques* (pp. 351–360). New York: ACM Press.

108. Hanan, J. S., & Hearn, A. B. (2003). Linking physiological and architectural models of cotton. *Agricultural Systems*, *75*(1), 47–77.

109. Mech, R., & Prusinkiewicz, P. (1996). Visual models of plants interacting with their environment. In *SIGGRAPH '96 Proceedings* (pp. 397–410). New York: ACM Press.

110. Hanan, J., Prusinkiewicz, P., Zalucki, M., & Skirvin, D. (2002). Simulation of insect movement with respect to plant architecture and morphogenesis. *Computers and Electronics in Agriculture*, *35*(2–3), 255–269.

111. Plotnick, R. E. (2003). Ecological and L-system based simulations of trace fossils. *Palaeogeography, Palaeoclimatology, Palaeoecology*, *192*(1–4), 45–58.

112. Maes, P. (1995). Artificial life meets entertainment: Lifelike autonomous agents. *Communications of the ACM*, *38*(11), 108–114.

113. Maes, P., Darrell, T., Blumberg, B., & Pentland, A. (1995). The ALIVE system: Full-body interaction with autonomous agents. In *Proceedings of Computer Animation '95* (pp. 11–18). Piscataway, NJ: IEEE.

114. Takamura, S., Hornby, G., Yamamoto, T., Yokono, J., & Fujita, M. (2000). Evolution of dynamic gaits for a robot. In *Proceedings of the International Conference on Consumer Electronics* (pp. 192–193). Piscataway, NJ: IEEE.

115. Hornby, G. S., Fujita, M., Takamura, S., Yamamoto, T., & Hanagata, O. (1999). Autonomous evolution of gaits with the Sony quadruped robot. In *Proceedings of 1999 Genetic and Evolutionary Computation Conference (GECCO)* (pp. 1297–1304). San Mateo, CA: Morgan Kaufmann.

116. Hornby, G. S., Takamura, S., Yokono, J., Hanagata, O., Yamamoto, T., & Fujita, M. (2000). Evolving robust gaits with AIBO. In *IEEE International Conference on Robotics and Automation,* Vol. 3, (pp. 3040–3045).

117. Collinot, A., Drogoul, A., & Benhamou, P. (1996). Agent oriented design of robotic soccer team. In *Proceedings of ICMAS'96* (pp. 41–57). American Association for Artificial Intelligence.

118. Pagliarini, L., Lund, H. H., Miglino, O., & Parisi, D. (1997). Artificial life: A new way to build educational and therapeutic games. In *Proceedings of Artificial Life V* (pp. 152–156). Cambridge, MA: MIT Press.

119. Woodcock, S. (1998). Game AI: The state of the industry. *Game Developer, Oct.*, pp. 28–35.

120. Grand, S., Cliff, D., & Malhotra, A. (1997). Creatures: Artificial life autonomous software agents for home entertainment. In *Proceedings of the First International Conference on Autonomous Agents (Agents'97)* (pp. 22–29).

121. Fogel, D. B., & Chellapilla, K. (2002). Verifying Anaconda's expert rating by competing against Chinook: Experiments in co-evolving a neural checkers player. *Neurocomputing*, *42*(1–4), 69–86.

122. Biles, J. A. (1994). Genjam: A genetic algorithm for generating jazz solos. In *International Computer Music Conference* (pp. 131–137). Computer Music Association.

123. Tokui, N., & Iba, H. (2000). Music composition with interactive evolutionary computation. In *Proceedings of the 3rd International Conference on Generative Art (GA2000)*.

124. Unemi, T. (2002). SBEAT3: A tool for multi-part music composition by simulated breeding. In *Artificial Life VIII: Proceedings* (pp. 410–413).

125. Unemi, T. (2002). A design of genetic encoding for breeding short musical pieces. In *ALife VIII Workshop Proceedings* (pp. 25–30).

126. Jacob, B. L. (1995). Composing with genetic algorithms. In *Proceedings of the International Computer Music Conference (ICMC'95)* (pp. 452–455). Nashville, TN: Computer Music Association.

127. Cho, S.-B. (2004). Emotional image and musical information retrieval with interactive genetic algorithm. *Proceedings of the IEEE*, *92*(4), 702–711.

128. Moroni, A., Manzolli, J., Zuben, F. V., & Gudwin, R. (2002). *Vox Populi: Evolutionary computation for music evolution* (NICS.A.05.02.v1). Campinas, Brazil: Interdisciplinary Nucleus for Sound Communication (NICS).

129. Burton, A. R., & Vladimirova, T. (1997). A genetic algorithm utilising neural network fitness evaluation for musical composition. In G. D. Smith, N. C. Steele, & R. F. Albrecht (Eds.), *Proceedings of the 1997 International Conference on Artificial Neural Networks and Genetic Algorithms* (pp. 220–224). Vienna: Springer-Verlag.

130. Johanson, B., & Poli, R. (1998). *GP-Music: An interactive genetic programming system for music generation with automated fitness raters* (Technical report CSRP-98-13). Birmingham, UK: School of Computer Science, The University of Birmingham.

131. De la Puente, O. A., Alfonso, R. S., & Moreno, M. A. (2002). Automatic composition of music of grammatical evolution. In *Proceedings of the 2002 Conference on APL* (pp. 148–155). New York: ACM Press.

132. Alander, J. T. (1994). *An indexed bibliography of genetic algorithms in arts and music* (Report 94-1-ART). Vaasa, Finland: Dept. of Information Technology and Production Economics, University of Vaasa.

133. Miranda, E. R. (2003). On the evolution of music in a society of self-taught digital creatures. *Digital Creativity*, *14*(1), 29–42.

134. Miranda, E. R. (2002). Sounds of artificial life. In *Proceedings of the Fourth Conference on Creativity & Cognition* (pp. 173–177). New York: ACM Press.

135. Wada, M., Kuroiwa, J., & Nara, S. (2002). Completely reproducible description of digital sound data with cellular automata. *Physics Letters A*, *306*(2–3), 110–115.

136. Tesfatsion, L. (1997). How economists can get ALife. In W. Brian Arthur, S. Durlauf, & D. Lane (Eds.), *The Economy as an Evolving Complex System, II* (pp. 533–564). Redwood City, CA: Addison-Wesley.

137. Tesfatsion, L. (2002). Agent-based computational economics: Growing economics from the bottom up. *Artificial Life*, *8*(1), 55–82.

138. Tesfatsion, L. (1998). Preferential partner selection in evolutionary labor markets: A study in agent-based computational economics. In V. W. Porto, N. Saravanan, D. Waagen, & A. E. Eiben (Eds.), *Evolutionary Programming VII, Proceedings of the Seventh Annual Conference on Evolutionary Programming* (pp. 15–24). Berlin: Springer-Verlag.

139. Tesfatsion, L. (2001). Structure, behavior, and market power in an evolutionary labor market with adaptive search. *Journal of Economic Dynamics and Control*, *25*, 419–457.

140. Tassier, T., & Menczer, F. (2001). Emerging small-world referral networks in evolutionary labor markets. *IEEE Transactions on Evolutionary Computation*, *5*(5), 482–492.

141. Lin, F.-R., Huang, S.-H., & Lin, S.-C. (2002). Effects of information sharing on supply chain performance in electronic commerce. *IEEE Transactions on Engineering Management*, *49*(3), 259–268.

142. Luna, F., & Stefansson, B. (2000). *Economic simulations in swarm: Agent-based modelling and object*. Dordrecht, The Netherlands: Kluwer Academic.

143. Jennings, N. R., Norman, T. J., & Faratin, P. (1998). ADEPT: An agent-based approach to business process management. *ACM SIGMOD Record*, *27*(4), 32–39.

144. Jennings, N. R., Faratin, P., Norman, T. J., O'Brien, P., & Odgers, B. (2000). Autonomous agents for business process management. *Applied Artificial Intelligence*, *14*(2), 145–189.

145. Jennings, N. R., Faratin, P., Norman, T. J., O'Brien, P., Odgers, B., & Alty, J. L. (2000). Implementing a business process management system using ADEPT: A real-world case study. *Applied Artificial Intelligence*, *14*(5), 421–465.

146. Tsvetovatyy, M., Gini, M., Mobasher, B., & Wieckowski, Z. (1997). MAGMA: An agent based virtual market for electronic commerce. *Applied Artificial Intelligence*, *11*(6), 501–523.

147. Kephart, J. O., Hanson, J. E., & Sairamesh, J. (1998). Price and niche wars in a free-market economy of software agents. *Artificial Life*, *4*(1), 1–23.

148. Kephart, J. O., Hanson, J. E., & Greenwald, A. R. (2000). Dynamic pricing by software agents. *Computer Networks*, *32*, 731–752.

149. Eymann, T., Padovan, B., & Schoder, D. (1998). Simulating value chain coordination with artificial life agents. In *Proceedings of the 3rd International Conference on Multi-Agent Systems (ICMAS'98)* (pp. 423–424).

150. Inoue, H., Funyu, Y., Kishino, K., Jinguji, T., Shiozawa, M., Yoshikawa, S., & Nakao, T. (2001). Development of artificial life based optimization system. In *Proceedings of Eighth International Conference on Parallel and Distributed Systems* (pp. 429–436). Piscataway, NJ: IEEE Computer Society.

151. Pagliarini, L., Dolan, A., Menczer, F., & Lund, H. H. (1998). ALife meets web: Lessons learned. In J.-C. Heudin (Ed.), *Virtual Worlds 1998* (pp. 156–167). New York: Springer-Verlag.

152. Menczer, F., Below, R.K., & Willuhn, W. (1995). Artificial life applied to adaptive information agents. In *Spring Symposium on Information Gathering from Distributed Heterogeneous Databases* (pp. 128–132). Washington, DC: AIAA Press.

153. Sheth, B., & Maes, P. (1993). Evolving agents for personalized information filtering. In *Proceedings of Ninth Conference on Artificial Intelligence for Applications* (pp. 345–352). Piscataway, NJ: IEEE.

154. Morrison, T., & Aickelin, U. (2002). An artificial immune system as a recommender for web sites. In J. Timmis & P. J. Bentley (Eds.), *Proceedings of the 1st International Conference on Artificial Immune Systems (ICARIS-2002)* (pp. 161–169).

155. Cayzer, S., & Aickelin, U. (2002). A recommender system based on the immune network. In *Proceedings of the 2002 Congress on Evolutionary Computation,* Vol. 1 (pp. 12–17). Piscataway, NJ: IEEE.

156. Chao, D. L., & Forrest, S. (2002). Information immune systems. In J. Timmis & P. J. Bentley (Eds.), *Proceedings of the First International Conference on Artificial Immune Systems (ICARIS)* (pp. 132–140).

157. Nasraoui, O., Gonzalez, F., & Dasgupta, D. (2002). The fuzzy artificial immune system: Motivations, basic concepts and application to clustering and web profiling. In *IEEE International Conference on Fuzzy Systems* (pp. 711–716).

158. Ibanez, J., Gomez-Skarmeta, A. F., & Blat, J. (2003). DJ-boids: Emergent collective behavior as multichannel radio station programming. In *Proceedings of the 2003 International Conference on Intelligent User Interfaces* (pp. 248–250). New York: ACM Press.

159. Lucic, P., & Teodorovic, D. (2002). Transportation modeling: An artificial life approach. In *Proceedings of the 14th IEEE International Conference on Tools with Artificial Intelligence* (pp. 216–223).

160. Solimanpur, M., Vrat, P., & Shankar, R. (2005). An ant algorithm for the single row layout problem in flexible manufacturing systems. *Computers and Operations Research*, *32*(3), 583–598.

161. Hoar, R., Penner, J., & Jacob, C. (2002). Evolutionary swarm traffic: If ant roads had traffic lights. In *Proceedings of the 2002 Congress on Evolutionary Computation,* Vol. 2 (pp. 1910–1915). Piscataway, NJ: IEEE.

162. Eggers, J., Feillet, D., Kehl, S., Wagner, M. O., & Yannou, B. (2003). Optimization of the keyboard arrangement problem using an ant colony algorithm. *European Journal of Operational Research*, *148*(3), 672–686.

163. McMullen, P. R. (2001). An ant colony optimization approach to addressing a JIT sequencing problem with multiple objectives. *Artificial Intelligence in Engineering, 15*(3), 309–317.

164. Jayaraman, V. K., Kulkarni, B. D., Karale, S., & Shelokar, P. (2000). Ant colony framework for optimal design and scheduling of batch plants. *Computers & Chemical Engineering, 24*(8), 1901–1912.

165. Abbass, H. A., Xuan, H. R., & McKay, I. (2002). AntTAG: A new method to compose computer programs using colonies of ants. In *Proceedings of the 2002 Congress on Evolutionary Computation,* Vol. 2 (pp. 1654-1659). Piscataway, NJ: IEEE.

166. Kim, H.-S., & Cho, S.-B. (2000). Application of interactive genetic algorithm to fashion design. *Engineering Applications of Artificial Intelligence*, *13*(6), 635–644.

167. Nishino, H., Takagi, H., Cho, S.-B., & Utsumiya, K. (2001). A 3D modeling system for creative design. In *Proceedings of the International Conference on Information Networking* (pp. 479–486). Piscataway, NJ: IEEE.

168. Yang, B.-S., Lee, Y.-H., Choi, B.-K., & Kim, H.-J. (2001). Optimum design of short journal bearings by artificial life algorithms. *Tribology International*, *34*, 427–435.

169. Ahn, Y. K., Song, J. D., & Yang, B.-S. (2003). Optimal design of engine mount using an artificial life algorithm. *Journal of Sound and Vibration*, *261*, 309–328.

170. Hraber, P., Jones, T., & Forrest, S. (1994). The ecology of Echo. *Artificial Life*, *3*(3), 165–190.

171. Jones, T., & Forrest, S. (1993). *An introduction to SFI echo* (Technical report 93-12-074). Santa Fe, NM: Santa Fe Institute.

172. Forrest, S., & Jones, T. (1994). Modeling complex adaptive systems with Echo. In R. J. Stonier & X. H. Yu (Eds.), *Complex systems: Mechanism of adaptation* (pp. 3–21). Amsterdam, The Netherlands: IOS Press.

173. Menczer, F., & Below, R. K. (1996). Latent energy environment. In *Adaptive individuals in evolving populations: Models and algorithms* (pp. 191–210). Santa Fe, NM: Santa Fe Institute.

174. Menczer, F., & Belew, R. K. (1993). *Latent energy environments: A tool for artificial life simulations* (Technical report CS93-301). San Diego: University of California.

175. Yaeger, L. (1994). Computational genetics, physiology, metabolism, neural systems, learning, vision, and behavior or PolyWorld: Life in a new context. In C. G. Langton (Ed.), *Proceedings of the Artificial Life III Conference* (pp. 263–298). Redwood City, CA: Addison-Wesley.

176. Booth, G. (1997). Gecko: A continuous 2-D world for ecological modeling. *Artificial Life Journal*, *3*(3), 147–163.

177. Calderoni, S., & Marcenac, P. (1998). Mutant: A multiagent toolkit for artificial life simulation. In M. Singh, B. Meyer, J. Gil, & R. Mitchell (Eds.), *Proceedings of the 26th International Conference on Technology of Object-Oriented Languages and Systems (TOOLS-USA-98)* (pp. 218–229). Santa Barbara, CA: IEEE Computer Society Press.

178. Minar, N., Burkhart, R., Langton, C., & Askenazi, M. (1996). The swarm simulation system: A toolkit for building multi-agent simulations. http://www.swarm.org/.

179. Sipper, M., & Ronald, E. M. A. (2000). A new species of hardware. *IEEE Spectrum*, *37*(3), 59–64.

180. Higuchi, T., Iwata, M., Keymeulen, D., Sakanashi, H., Murakawa, M., Kajitani, I., Takahashi, E., Toda, K., Salami, M., Kajihara, N., & Otsu, N. (1999). Real-world applications of analog and digital evolvable hardware. *IEEE Transactions on Evolutionary Computation*, *3*(3), 220–235.

181. Salami, M., Iwata, M., & Higuchi, T. (1997). Lossless image compression by evolvable hardware. In *Proceedings of 4th European Conference on Artificial Life (ECAL97)* (pp. 407–416). Cambridge, MA: MIT Press.

182. Keymeulen, D., Iwata, M., Kuniyoshi, Y., & Higuchi, T. (1998). Comparison between an off-line model-free and an on-line model-based evolution applied to a robotics navigation system using evolvable hardware. In *Proceedings of the Sixth International Conference on Artificial Life* (pp. 199–209). Cambridge, MA: MIT Press.

183. Yao, X., & Higuchi, T. (1999). Promises and challenges of evolvable hardware. *IEEE Transactions on Systems, Man and Cybernetics—Part C*, *29*(1), 87–97.

184. Zebulum, R., Pacheco, M. A., & Vellasco, M. (1998). Comparison of different evolutionary methodologies applied to electronic filter design. In *Proceedings of the IEEE International Conference on Evolutionary Computation (ICEC98)* (pp. 434–439).

185. Kuntz, P., Layzell, P., & Snyers, D. (1997). A colony of ant-like agents for partitioning in VLSI technology. In *Proceedings of European Conference on Artificial Life (ECAL 97)*. Cambridge, MA: MIT Press.

186. Issacs, J. C., Watkins, R. K., & Foo, S. Y. (2002). Evolving ant colony systems in hardware for random number generation. In *Proceedings of the 2002 Congress on Evolutionary Computation,* Vol. 2 (pp. 1450–1455). Piscataway, NJ: IEEE Neural Networks Council.

187. Scheuermann, B., So, K., Guntsch, M., Middendorf, M., Diessel, O., El Gindy, H., & Schmeck, H. (2004). FPGA implementation of population-based ant colony optimization. *Applied Soft Computing*, *4*, 303–322.

188. Robinson, J., & Rahmat-Samii, Y. (2004). Particle swarm optimization in electromagnetics. *IEEE Transactions on Antennas and Propagation*, *52*(2), 397–407.

189. Coleman, C. M., Rothwell, E. J., & Ross, J. E. (2004). Investigation of simulated annealing, ant-colony optimization, and genetic algorithms for self-structuring antennas. *IEEE Transactions on Antennas and Propagation*, *52*(4), 1007–1014.

190. Stauffer, A., Sipper, M., & Pérez-Uribe, A. (1998). Some applications of FPGAs in bio-inspired hardware. In *Proceedings of 6th IEEE Symposium on FPGAs for Custom Computing Machines (FCCM'98)* (pp. 278–279).

191. Linhares, A. (1999). Synthesizing a predatory search strategy for VLSI layouts. *IEEE Transactions on Evolutionary Computation*, *3*(2), 147–152.

192. Mange, D., Sipper, M., Stauffer, A., & Tempesti, G. (2000). Towards robust integrated circuits: The embryonics approach. *Proceedings of the IEEE*, *88*(4), 516–541.

193. Verwoerd, T., & Hunt, R. (2002). Intrusion detection techniques and approaches. *Computer Communications*, *25*(15), 1356–1365.

194. Harmer, P. K., Williams, P. D., Gunsch, G. H., & Lamont, G. B. (2002). An artificial immune system architecture for computer security applications. *IEEE Transactions on Evolutionary Computation*, *6*(3), 252–279.

195. Dasgupta, D., & Attoh-Okine, N. (1997). Immunity-based systems: A survey. In *IEEE International Conference on Systems, Man and Cybernetics,* Vol. 1, (pp. 369–374).

196. Dasgupta, D., & Gonzalez, F. (2002). An immunity-based technique to characterize intrusions in computer networks. *IEEE Transactions on Evolutionary Computation*, *6*(3), 281–291.

197. Aickelin, U., Bentley, P., Cayzer, S., Kim, J., & Mcleod, J. (2003). Danger theory: The link between AIS and IDS? In *Proceedings of the Second International Conference on Artificial Immune Systems* (pp. 147–155). New York: Springer-Verlag.

198. Spafford, E. H. (1991). *Computer viruses—A form of artificial life?* (Technical report CSD-TR-985). Department of Computer Sciences, Purdue University.

199. Kephart, J. O. (1994). A biologically inspired immune system for computers. In *Artificial Life IV, Proceedings of the Fourth International Workshop on the Synthesis and Simulation of Living Systems* (pp. 130–139). Cambridge, MA: MIT Press.

200. Hedberg, S. (1996). Combating computer viruses: IBM's new computer immune system. *IEEE Parallel & Distributed Technology*, *4*(2), 9–11.

201. Marmelstein, R. E., Veldhuizen, D. A. V., & Lamont, G. B. (1998). A distributed architecture for an adaptive computer virus immune system. In *IEEE International Conference on Systems, Man and Cybernetics,* Vol. 4 (pp. 11–14).

202. Ilachinski, A. (2000). Irreducible semi-autonomous adaptive combat (ISAAC): An artificial life approach to land combat. *Military Operations Research*, *5*(3), 29–46.

203. Lauren, M. K., & Stephen, R. T. (2002). MANA: Map-aware non-uniform automata—A New Zealand approach to scenario modeling. *Journal of Battlefield Technology*, *5*(1), 27–31.

204. Yang, A., Abbass, H. A., & Sarker, R. (2004). Landscape dynamics in multi-agent simulation combat systems. In G. I. Webb & X. Yu (Eds.), *The 17th Australian Joint Conference on Artificial Intelligence* (pp. 39–50). New York: Springer-Verlag.

205. Ilachinski, A. (2004). *Artificial war: Multiagent-based simulation of combat.* Singapore: World Scientific Publishing.

206. Ngan, P. S., Wong, M. L., Lam, W., Leung, K. S., & Cheng, J. C. Y. (1999). Medical data mining using evolutionary computation. *Artificial Intelligence in Medicine*, *16*, 73–96.

207. Pena-Reyes, C. A., & Sipper, M. (2000). Evolutionary computation in medicine: An overview. *Artificial Intelligence in Medicine*, *19*, 1–23.

208. Calabretta, R., Nolfi, S., & Parisi, D. (1995). An artificial life model for predicting the tertiary structure of unknown proteins that emulates the folding process. In F. Morán, A. Moreno, J. J. Merelo Cuervós, & P. Chacón (Eds.), *Advances in Artificial Life* (pp. 862–875). New York: Springer-Verlag.

209. Iba, H., & Mimura, A. (2002). Inference of a gene regulatory network by means of interactive evolutionary computing. *Information Sciences*, *145*, 225–236.

210. Kim, K.-J., & Cho, S.-B. (2004). Prediction of colon cancer using an evolutionary neural network. *Neurocomputing*, *61*, 361–379.

211. Parpinelli, R. S., Lopes, H. S., & Freitas, A. A. (2002). Data mining with an ant colony optimization algorithm. *IEEE Transactions on Evolutionary Computation*, *6*(4), 321–332.

212. Shelokar, P. S., Jayaraman, V. K., & Kulkarni, B. D. (2004). An ant colony classifier system: Application to some process engineering problems. *Computers & Chemical Engineering*, *28*, 1577–1584.

213. Liu, B., Abbass, H. A., & McKay, R. I. (2004). Classification rule discovery with ant colony optimization. *IEEE Computational Intelligence Bulletin*, *3*(1), 31–35.

214. Abraham, A., & Ramos, V. (2003). Web usage mining using artificial ant colony clustering and linear genetic programming. In *IEEE Congress on Evolutionary Computation* (pp. 1384–1391).

215. Tsai, C.-F., Tsai, C.-W., Wu, H.-C., & Yang, T. (2004). ACODF: A novel data clustering approach for data mining in large databases. *The Journal of Systems and Software*, *73*(1), 133–145.

216. Lu, W. Z., Fan, H. Y., & Lo, S. M. (2003). Application of evolutionary neural network method in predicting pollutant levels in downtown area of Hong Kong. *Neurocomputing*, *51*, 387–400.

217. Sim, K. M., & Sun, W. H. (2002). Multiple ant-colony optimization for network routing. In *Proceedings of the First International Symposium on Cyber Worlds* (pp. 277-281). Piscataway, NJ: IEEE Computer Society.

218. Bonabeau, E., Henaux, F., Guerin, S., Snyers, D., Kuntz, P., & Theraulaz, G. (1998). Routing in telecommunications networks with "Smart" ant-like agents (Working paper 98-01-003). Santa Fe, NM: Santa Fe Institute.

219. Di Caro, G., & Dorigo, M. (1998). AntNet: Distributed stigmergetic control for communications networks. *Journal of Artificial Intelligence Research*, *9*, 317–365.

220. Sim, K. M., & Sun, W. H. (2003). Ant colony optimization for routing and load-balancing: Survey and new directions. *IEEE Transactions on Systems, Man and Cybernetics (Part A)*, *33*(5), 560–572.

221. Denby, B., & Hegarat-Mascle, S. L. (2003). Swarm intelligence in optimization problems. *Nuclear Instruments & Methods in Physics Research A*, *502*, 364–368.

222. Subrata, R., & Zomaya, A. Y. (2003). A comparison of three artificial life techniques for reporting cell planning in mobile computing. *IEEE Transactions on Parallel and Distributed Systems*, *14*(2), 142–153.

223. Subrata, R., & Zomaya, A. Y. (2003). Evolving cellular automata for location management in mobile computing networks. *IEEE Transactions on Parallel and Distributed Systems*, *14*(1), 13–26.

224. Dorigo, M., Gambardella, L. M., Middendorf, M., & Stutzle, T. (2002). Special section on ant colony optimization. *IEEE Transactions on Evolutionary Computation*, *6*(4).

225. Dasgupta, D. (2002). Special issue on artificial immune systems. *IEEE Transaction on Evolutionary Computation*, *6*(3).

226. Tesfatsion, L. (2001). Special issue on agent-based modeling of evolutionary economic systems. *IEEE Transactions on Evolutionary Computation*, *5*(5).