Preface

Software Science is a discipline that studies the theoretical framework of software as instructive and behavioral information, which can be embodied and executed by generic computers in order to create expected system behaviors and machine intelligence. *Intelligence Science* is a discipline that studies the mechanisms and theories of abstract intelligence and its paradigms such as natural, artificial, machinable, and computational intelligence. The convergence of software and intelligent sciences forms the transdisciplinary field of computational intelligence, which provides a coherent set of fundamental theories, contemporary denotational mathematics, and engineering applications.

This book entitled *Breakthroughs in Software Science and Computational Intelligence* is the second volume in the IGI Series of Advances in Software Science and Computational Intelligence. The book encompasses 25 chapters of expert contributions selected from the *International Journal of Software Science and Computational Intelligence* during 2010. The book is organized in four sections on: (1) Computational intelligence; (2) Cognitive computing; (3) Software science; and (4) Applications in computational intelligence and cognitive computing.

SECTION 1: COMPUTATIONAL INTELLIGENCE

Intelligence science studies theories and models of the brain at all levels, and the relationship between the concrete physiological brain and the abstract soft mind. Intelligence science is a new frontier with the fertilization of biology, psychology, neuroscience, cognitive science, cognitive informatics, philosophy, information science, computer science, anthropology, and linguistics. A fundamental view developed in software and intelligence sciences is known as abstract intelligence, which provides a unified foundation for the studies of all forms and paradigms of intelligence such as natural, artificial, machinable, and computational intelligence. *Abstract Intelligence* (αI) is an enquiry of both natural and artificial intelligence at the neural, cognitive, functional, and logical levels from the bottom up. In the narrow sense, αI is a human or a system ability that transforms information into behaviors. However, in the broad sense, αI is any human or system ability that autonomously transfers the forms of abstract information between *data, information, knowledge,* and *behaviors* in the brain or intelligent systems.

Computational Intelligence (CoI) is an embodying form of Abstract Intelligence (αI) that implements intelligent mechanisms and behaviors by computational methodologies and software systems, such as expert systems, fuzzy systems, cognitive computers, cognitive robots, software agent systems, genetic/evolutionary systems, and autonomous learning systems. The theoretical foundations of computational intelligence root in cognitive informatics, software science, and denotational mathematics.

This section on computational intelligence encompasses the following five chapters:

- Chapter 1: Perspectives on Cognitive Computing and Applications
- Chapter 2: Granular Computing and Human-Centricity in Computational Intelligence
- Chapter 3: A General Knowledge Representation Model for the Acquisition of Skills and Concepts
- Chapter 4: Feature Based Rule Learner in Noisy Environment Using Neighbourhood Rough Set Model
- Chapter 5: A High Level Model of a Conscious Embodied Agent

Chapter 1, "Perspectives on Cognitive Computing and Applications," by Yingxu Wang, Witold Pedrycz, George Baciu, Ping Chen, Guoyin Wang, and Yiyu Yao, presents *Cognitive Computing* (CC) as an emerging paradigm of intelligent computing theories and technologies based on cognitive informatics, which implements computational intelligence by autonomous inferences and perceptions mimicking the mechanisms of the brain. The development of *Cognitive Computers* (CogC) is centric in cognitive computing methodologies. A CogC is an intelligent computer for knowledge processing as that of a conventional von Neumann computer for data processing. This chapter summarizes the presentations of a set of 6 position chapters presented in the ICCI 2010 *Plenary Panel on Cognitive Computing and Applications* contributed from invited panelists who are part of the world's renowned researchers and scholars in the field of cognitive informatics and cognitive computing.

Chapter 2, "Granular Computing and Human-Centricity in Computational Intelligence," by Witold Pedrycz, presents information granules and the general framework of Granular Computing, which offers interesting opportunities to endow processing with an important facet of human-centricity. This facet directly implies that the underlying processing supports non-numeric data inherently associated with the variable perception of humans and generates results being seamlessly comprehended by users. Given that systems, which quite commonly become distributed and hierarchical, managing granular information in hierarchical and distributed architectures, are of growing interest, especially when invoking mechanisms of knowledge generation and knowledge sharing. The outstanding feature of human centricity of Granular Computing along with essential fuzzy set-based constructs constitutes the crux of our study. We elaborate on some new directions of knowledge elicitation and quantification realized in the setting of fuzzy sets. With this regard, we concentrate on an idea of *knowledge*-based clustering, which aims at the seamless realization of the data-expertise design of information granules. It is also emphasized that collaboration and reconciliation of locally available knowledge give rise to the concept of higher type information granules. The other interesting directions enhancing human centricity of computing with fuzzy sets, deals with non-numeric, semi-qualitative characterization of information granules (fuzzy sets), as well as inherent evolving capabilities of associated human-centric systems. We discuss a suite of algorithms facilitating a qualitative assessment of fuzzy sets, formulate a series of associated optimization tasks guided by well-formulated performance indexes, and discuss the underlying essence of the resulting solutions.

Chapter 3, "A General Knowledge Representation Model for the Acquisition of Skills and Concepts," by Carlos Ramirez and Benjamin Valdes, presents a cognitive model for skills and concepts representation as well as a proposal for its computational implementation. The model is intended to help bridge some of the natural problems that arise in current massive education models, through the adaptation and personalization of learning environments. The model is capable of representing rich semantic knowledge, including both skills and concepts, integrating them through a coherent network of role-based associations. The associations build an ontology that integrates on itself different domain taxonomies to represent the knowledge acquired by a student keeping relevant context information. The model is based on a constructivist approach.

Chapter 4, "Feature Based Rule Learner in Noisy Environment Using Neighbourhood Rough Set Model," by Yang Liu, Luyang Jiao, Guohua Bai, and Boqin Feng, presents a perspective of cognitive informatics where cognition can be viewed as the acquisition of knowledge. In real-world applications, information systems usually contain some degree of noisy data. A new model, which combines the neighbourhood approximation and variable precision rough set model, is proposed to deal with hybrid-feature selection problem. Then rule induction algorithm can learn from selected features in order to reduce the complexity of rule sets. Through proposed integration, the knowledge acquisition process becomes insensitive to the dimensionality of data with a pre-defined tolerance degree of noise and uncertainty for misclassification. When we apply the method to a Chinese diabetic diagnosis problem, the hybrid-attribute reduction method selected only five attributes from totally thirty-four measurements. Rule learner produced eight rules with average two attributes in the left part of an IF-THEN rule form, which is a manageable set of rules. The demonstrated experiment shows that the present approach is effective in handling real-world problems.

Chapter 5, "A High Level Model of a Conscious Embodied Agent," by Jirí Wiedermann, presents a schematic yet cognitively powerful architecture of an embodied conscious agent. The architecture incorporates a mechanism for mining, representing, processing, and exploiting semantic knowledge. This mechanism is based on two complementary internal world models, which are built automatically. One model (based on artificial mirror neurons) is used for mining and capturing the syntax of the recognized part of the environment, while the second one (based on neural nets) is used for its semantics. Jointly, the models support algorithmic processes underlying phenomena similar in important aspects to higher cognitive functions such as imitation learning and the development of communication, language, thinking and consciousness.

SECTION 2: COGNITIVE COMPUTING

Computing systems and technologies can be classified into the categories of *imperative*, *autonomic*, and *cognitive* computing from the bottom up. The imperative computers are a passive system based on stored-program controlled mechanisms for data processing. The autonomic computers are goal-driven and self-decision-driven machines that do not rely on instructive and procedural information. Cognitive computers are more intelligent computers beyond the imperative and autonomic computers, which embody major natural intelligence behaviors of the brain such as thinking, inference, and learning. The increasing demand for non von Neumann computers for knowledge and intelligence processing in the high-tech industry and everyday lives require novel cognitive computers for providing autonomous computing power for various cognitive systems mimicking the natural intelligence of the brain.

Cognitive Computing (CC) is a novel paradigm of intelligent computing methodologies and systems based on *Cognitive Informatics* (CI), which implements computational intelligence by autonomous inferences and perceptions mimicking the mechanisms of the brain. CC emerged and developed based on transdisciplinary research in cognitive informatics, abstract intelligence, and *Denotational Mathematics* (DM). The latest advances in CI, CC, and DM enable a systematic solution for the future generation of intelligent computers known as *Cognitive Computers* (CogCs) that think, perceive, learn, and reason. A

CogC is an intelligent computer for knowledge processing just as a conventional von Neumann computer is for data processing. CogCs are designed to embody *machinable intelligence* such as computational inferences, causal analyses, knowledge manipulations, machine learning, and autonomous problem solving.

This section on cognitive computing encompasses the following six chapters:

- Chapter 6: Cognitive Memory: Human-Like Memory
- Chapter 7: Multi-Fractal Analysis for Feature Extraction from DNA Sequences
- Chapter 8: Agent-Based Middleware for Advanced Communication Services in a Ubiquitous Computing Environment
- Chapter 9: Relevant and Non-Redundant Amino Acid Sequence Selection for Protein Functional Site Identification
- Chapter 10: Role-Based Autonomic Systems
- Chapter 11: Combining Ontology with Intelligent Agent to Provide Negotiation Service

Chapter 6, “Cognitive Memory: Human-Like Memory,” by Bernard Carlos Widrow and Juan Aragon, presents a new form of computer memory inspired from life experience. Certain conjectures about human memory are keys to the central idea. We present the design of a practical and useful “cognitive” memory system. The new memory does not function like a computer memory where specific data is stored in specific numbered registers and retrieval is done by reading the contents of the specified memory register, or done by matching key words as with a document search. Incoming sensory data would be stored at the next available empty memory location, and indeed could be stored redundantly at several empty locations. The stored sensory data would neither have key words nor would it be located in known or specified memory locations. Retrieval would be initiated by a prompt signal from a current set of sensory inputs or patterns. The search would be done by a retrieval system that makes use of auto-associative artificial neural networks. In this chapter we present a practical application of this cognitive memory system to human facial recognition.

Chapter 7, “Multi-Fractal Analysis for Feature Extraction from DNA Sequences,” by Witold Kinsner and Hong Zhang, presents estimations of multiscale (multifractal) measures for feature extraction from Deoxyribonucleic Acid (DNA) sequences, and demonstrates the intriguing possibility of identifying biological functionality using information contained within the DNA sequence. We have developed a technique that seeks patterns or correlations in the DNA sequence at a higher level than the local base-pair structure. The technique has three main steps: (1) transforms the DNA sequence symbols into a modified Lévy walk, (2) transforms the Lévy walk into a signal spectrum, and (3) breaks the spectrum into subspectra and treats each of these as an attractor from which the multifractal dimension spectrum is estimated. An optimal minimum window size and volume element size are found for estimation of the multifractal measures. Experimental results show that DNA is multifractal, and that the multifractality changes depending upon the location (coding or noncoding region) in the sequence.

Chapter 8, “Agent-Based Middleware for Advanced Communication Services in a Ubiquitous Computing Environment,” by Takuo Suganuma, Hideyuki Takahashi, and Norio Shiratori, presents a ubiquitous computing (ubicomp) environment where system components of different types including hardware elements, software components, and network connections must cooperate mutually to provide services that fulfill user requirements. Consequently, advanced and flexible characteristics of software that are specialized for a ubicomp environment are needed. This chapter presents a proposal of an agent-based middleware for a ubicomp environment comprising computers and home electric appliances. The middleware, called AMUSE, can take quality of service (QoS) in a ubicomp environment into consideration by coping not only with user context but also with the resource context, using agent-based computing technology. Herein, we describe the concept, design, and initial implementation of AMUSE. Simulation results of an experimental ubiquitous service using AMUSE demonstrate the effectiveness of our proposed scheme. Additionally, to confirm our scheme's feasibility and effectiveness, we describe the initial implementation of a multimedia communication application based on AMUSE.

Chapter 9, "Relevant and Non-Redundant Amino Acid Sequence Selection for Protein Functional Site Identification," by Chandra Das and Pradipta Maji, presents a pattern recognition algorithm to predict functional sites in proteins where amino acids cannot be used directly as inputs since they are nonnumerical variables. They, therefore, need encoding prior to input. In this regard, the bio-basis function maps a nonnumerical sequence space to a numerical feature space. It is designed using an amino acid mutation matrix. One of the important issues for the bio-basis function is how to select a minimum set of bio-basis strings with maximum information. In this chapter, an efficient method to select bio-basis strings for the bio-basis function is described integrating the concepts of the Fisher ratio and "degree of resemblance." The integration enables the method to select a minimum set of most informative bio-basis strings. The concept of asymmetric biological distance is introduced to compute the Fisher ratio, which is more effective for selection of most relevant bio-basis strings. The "degree of resemblance" enables efficient selection of a set of distinct bio-basis strings. In effect, it reduces the redundant features in numerical feature space. Some quantitative indices are proposed for evaluating the quality of selected bio-basis strings. The effectiveness of the proposed bio-basis string selection method, along with a comparison with existing methods, is demonstrated on different data sets.

Chapter 10, "Role-Based Autonomic Systems," by Haibin Zhu, presents autonomic computing as an emerging computing paradigm in order to create computer systems capable of self-management and overcome the rapidly growing complexity of computing systems management. To possess self-* properties, there must be mechanisms to support self-awareness, i.e., an autonomic system should be able to perceive the abnormality of its components. After abnormality is checked, processes of self-healing, self-configuration, self-optimization, and self-protection must be completed to guarantee the system works correctly and continuously. In Role-Based Collaboration (RBC), roles are the major media for interaction, coordination, and collaboration. A role can be used to check if a player behaves well or not. This chapter investigates the possibility of using roles and their related mechanisms to diagnose the behavior of agents, and facilitate self-* properties of a system. The chapter asserts that role-based systems are autonomic.

Chapter 11, "Combining Ontology with Intelligent Agent to Provide Negotiation Service," by Qiumei Pu, Yongcun Cao, Xiuqin Pan, Siyao Fu, and Zengguang Hou, presents agent and ontology as distinct technologies which arose independent of each other, having their own standards and specifications. At the same time, the semantics Web is one of the popular research areas these days. It is based on the current Web, and adds more semantics to it, for the purpose of building the Ontology of Web content. So, application programs on the Web can make the purpose of cross-platform calculation come true by taking advantage of Ontology. On the other side of the coin, agent is a theory able to enhance the abstraction of software itself. Negotiation protocol is the basic principle in the electronic commerce, this principle has a direct impact on the efficiency of the negotiation. This study brings up the communication architecture with negotiation protocol on the Semantic Web. Precisely speaking, agents make computing with Ontology, and we define an agent's communication ontology for this communication framework. Semantic Web uses Ontology to describe the negotiation protocol, which will enable an agent to gain the necessary knowledge from the market. This will give agents the ability to understand each other enough to carry out their objectives. In this way, buyer or seller is able to improve semantic cognitive in the process of negotiation. At the same time, it can provide an intelligent platform for the information exchange on the same understanding about the content of communication in the electronic negotiation service.

SECTION 3: SOFTWARE SCIENCE

Software as instructive behavioral information has been recognized as an entire range of widely and frequently used objects and phenomena in human knowledge. Software science is a theoretical inquiry of software and its constraints on the basis of empirical studies on engineering methodologies and techniques for software development and software engineering organization. In the history of science and engineering, a matured discipline always gave birth to new disciplines. For instance, theoretical physics was emerged from general and applied physics, and theoretical computing was emerged from computer engineering. So does software science that emerges from and grows in the fields of software, computer, information, knowledge, and system engineering.

Software Science (SS) is a discipline of enquiries that studies the theoretical framework of software as instructive and behavioral information, which can be embodied and executed by generic computers in order to create expected system behaviors and machine intelligence. The discipline of software science studies the common objects in the abstract world such as software, information, data, concepts, knowledge, instructions, executable behaviors, and their processing by natural and artificial intelligence. From this view, software science is theoretical software engineering; while software engineering is applied software science in order to efficiently, economically, and reliably develop large-scale software systems. The phenomena shows that almost all the fundamental problems, which could not be solved in the last four decades in software engineering, simply stemmed from the lack of coherent theories in the form of software science. The vast cumulated empirical knowledge and industrial practice in software engineering have made this possible to enable the emergence of software science.

This section on software science encompasses the following six chapters:

- Chapter 12: Design and Implementation of an Autonomic Code Generator Based on RTPA
- Chapter 13: The Formal Design Model of a Real-Time Operating System (RTOS+): Conceptual and Architectural Frameworks
- Chapter 14: The Formal Design Model of a Real-Time Operating System (RTOS+): Static and Dynamic Behaviors
- Chapter 15: The Formal Design Model of an Automatic Teller Machine (ATM)
- Chapter 16: The Formal Design Models of a Set of Abstract Data Types (ADTs)
- Chapter 17: A Least-Laxity-First Scheduling Algorithm of Variable Time Slice for Periodic Tasks

Chapter 12, **“**Design and Implementation of an Autonomic Code Generator Based on RTPA,” by Yingxu Wang, Xinming Tan, and Cyprian F. Ngolah, presents a denotational mathematics, Real-Time Process Algebra (RTPA), for the algebraic modeling and manipulations of software system architectures and behaviors by the Unified Data Models (UDMs) and Unified Process Models (UPMs). On the basis of the RTPA specification and refinement methodologies, automatic software code generation is enabled toward improving software development productivity. This chapter presents the work on designing and developing the RTPA-Based Software Code Generator (RTPA-CG) that transfers system models in RTPA architectures and behaviors into C++ or Java. A two-phrase strategy has been employed in the design of the code generator. The first phrase analyzes the lexical, syntactical, and type specifications of a software system modeled in RTPA, which results in a set of Abstract Syntax Trees (ASTs). The second phrase translates the ASTs into C++ or Java based on predesigned mapping strategies and code generation rules. The toolkit of RTPA code generator encompasses an RTPA lexer, parser, type-checker, and a code builder. Experimental results show that system models in RTPA can be rigorously processed and corresponding C++/Java code can be automatically generated using the toolkit. The code generated is executable and effective under the support of an RTPA run-time library.

Chapter 13, "The Formal Design Model of a Real-Time Operating System (RTOS+): Conceptual and Architectural Frameworks," by Yingxu Wang, Cyprian F. Ngolah, Guangping Zeng, Philip C. Y. Sheu, C. Philip Choy, and Yousheng Tian, presents a Real-Time Operating System (RTOS+) as platform for the design and implementation of a wide range of applications in real-time systems, embedded systems, and mission-critical systems. This chapter describes a formal design model for a general RTOS known as RTOS+ that enables a specific target RTOS to be rigorously and efficiently derived in real-world applications. The methodology of a denotational mathematics, Real-Time Process Algebra (RTPA), is described for formally modeling and refining architectures, static behaviors, and dynamic behaviors of RTOS+. The conceptual model of the RTOS+ system is introduced as the initial requirements for the system. The architectural model of RTOS+ is created using RTPA architectural modeling methodologies and refined by a set of Unified Data Models (UDMs). The static behaviors of RTOS+ are specified and refined by a set of Unified Process Models (UPMs). The dynamic behaviors of the RTOS+ system are specified and refined by the real-time process scheduler and system dispatcher. This work is presented in two chapters in serial due to its excessive length. The conceptual and architectural models of RTOS+ is described in this chapter; while the static and dynamic behavioral models of RTOS+ will be elaborated in Chapter 14.

Chapter 14, "The Formal Design Model of a Real-Time Operating System (RTOS+): Static and Dynamic Behaviors," by Yingxu Wang, Guangping Zeng, Cyprian F. Ngolah, Philip C. Y. Sheu, C. Philip Choy, and Yousheng Tian, presents a Real-Time Operating System (RTOS+), which is the second part of the work succeeding Chapter 13. In this chapter, the static behaviors of RTOS+ are specified and refined by a set of Unified Process Models (UPMs). The dynamic behaviors of the RTOS+ system are specified and refined by the real-time process scheduler and system dispatcher. RTOS+ provides a platform for the design and implementation of a wide range of applications in real-time systems, embedded systems, and mission-critical systems.

Chapter 15, "The Formal Design Model of an Automatic Teller Machine (ATM)," by Yingxu Wang, Yanan Zhang, Philip C. Y. Sheu, Xuhui Li, and Hong Guo, presents an Automated Teller Machine (ATM) as a safety-critical and real-time system that is highly complicated in design and implementation. This chapter presents the formal design, specification, and modeling of the ATM system using a denotational mathematics known as Real-Time Process Algebra (RTPA). The conceptual model of the ATM system is introduced as the initial requirements for the system. The architectural model of the ATM system is created using RTPA architectural modeling methodologies and refined by a set of Unified Data Models (UDMs), which share a generic mathematical model of tuples. The static behaviors of the ATM system are specified and refined by a set of Unified Process Models (UPMs) for the ATM transition processing and system supporting processes. The dynamic behaviors of the ATM system are specified and refined by process priority allocation, process deployment, and process dispatch models. Based on the formal design models of the ATM system, code can be automatically generated using the RTPA Code Generator (RTPA-CG), or be seamlessly transformed into programs by programmers. The formal models of ATM may not only serve as a formal design paradigm of real-time software systems, but also a test bench for the expressive power and modeling capability of exiting formal methods in software engineering.

Chapter 16, "The Formal Design Models of a Set of Abstract Data Types (ADTs)," by Yingxu Wang, Xinming Tan, Cyprian F. Ngolah, and Philip Sheu, presents type theories as one of the fundamental theories underpinning data object modeling and system architectural design in computing and software engineering. Abstract Data Types (ADTs) are a set of highly generic and rigorously modeled data structures in type theory. ADTs are not only important in type modeling, but also play a key role in Object-Oriented (OO) technologies for software system design and implementation. This chapter presents a formal modeling methodology for ADTs using the Real-Time Process Algebra (RTPA), which allows both architectural and behavioral models of ADTs and complex data objects to be rigorously designed and specified in a top-down approach. Formal architectures, static behaviors, and dynamic behaviors of a set of ADTs, such as stack, queue, sequence, and record, are comparatively studied. The architectural models of the ADTs are created using RTPA architectural modeling methodologies known as the Unified Data Models (UDMs). The static behaviors of the ADTs are specified and refined by a set of Unified Process Models (UPMs) of RTPA. The dynamic behaviors of the ADTs are modeled by process dispatching technologies of RTPA. This work has been applied in a number of real-time and nonreal-time system designs such as a Real-Time Operating System (RTOS+), a Cognitive Learning Engine (CLE), an ADT library for an RTPA support tool, and the automatic code generator based on RTPA.

Chapter 17, "A Least-Laxity-First Scheduling Algorithm of Variable Time Slice for Periodic Tasks," by Shaohua Teng, Wei Zhang, Haibin Zhu, Xiufen Fu, Jiangyi Su, and Baoliang Cui, presents a Least-Laxity-First (LLF) scheduling algorithm that assigns a priority to a task according to its executing urgency. The smaller the laxity value of a task is, the sooner it needs to be executed. When two or more tasks have the same or approximative laxity values, the LLF scheduling algorithm leads to frequent switches among tasks, causes extra overhead in a system, and therefore, restricts its application. The least switch and laxity first scheduling algorithm is proposed in the chapter by searching out an appropriate common divisor in order to improve the LLF algorithm for periodic tasks.

SECTION 4: APPLICATIONS OF COMPUTATIONAL INTELLIGENCE AND COGNITIVE COMPUTING

A series of fundamental breakthroughs have been recognized and a wide range of applications has been developed in software science, abstract intelligence, cognitive computing, and computational intelligence in the last decade. Because software science and computational intelligence provide a common and general platform for the next generation of cognitive computing, some expected innovations in these fields will emerge such as cognitive computers, cognitive knowledge representation technologies, semantic searching engines, cognitive learning engines, cognitive Internet, cognitive robots, and autonomous inference machines for complex and long-series inferences, problem solving, and decision making beyond traditional logic- and rule-based technologies.

This section on applications of computational intelligence and cognitive computing encompasses the following eight chapters:

- Chapter 18: Cognitive Location-Aware Information Retrieval by Agent-Based Semantic Matching
- Chapter 19: Perceiving the Social: A Multi-Agent System to Support Human Navigation in Foreign Communities
- Chapter 20: Symbiotic Aspects in e-Government Application Development
- Chapter 21: CoPBoard: A Catalyst for Distributed Communities of Practice
- Chapter 22: Remote Conversation Support for People with Aphasia
- Chapter 23: Estimating which Object Type a Sensor Node is Attached to in Ubiquitous Sensor Environment
- Chapter 24: Delay-Range-Dependent Robust Stability for Uncertain Stochastic Neural Networks with Time-Varying Delays
- Chapter 25: Classifier Ensemble Based Analysis of a Genome-Wide SNP Dataset Concerning Late-Onset Alzheimer Disease

Chapter 18, "Cognitive Location-Aware Information Retrieval by Agent-Based Semantic Matching," by Eddie C. L. Chan, George Baciu, and S. C. Mak, presents an agent system operating in both wired and wireless networks that finds and retrieves location-aware information. Agents in the proposed system must have the full range of abilities, including perception, use of natural language, learning, and the ability to understand user queries. The speed and accuracy of retrieval and the usefulness of the retrieved data depends on a number of factors including constant or frequent changes in its content or status, the effects of environmental factors such as the weather and traffic, and the techniques that are used to categorize the relevance of the retrieved data. This chapter proposes semantic TFIDF, an agent-based system for retrieving location-aware information that makes use of semantic information in the data to develop smaller training sets, thereby improving the speed of retrieval while maintaining or even improving accuracy. This proposed method first assigns intelligent agents to gathering location-aware data, which they then classify, match, and organize to find a best match for a user query. This is done using semantic graphs in the WordNet English dictionary. Experiments will compare the proposed system with three other commonly used systems and show that it is significantly faster and more accurate.

Chapter 19, "Perceiving the Social: A Multi-Agent System to Support Human Navigation in Foreign Communities," Victor V. Kryssanov, Shizuka Kumokawa, Igor Goncharenko, and Hitoshi Ogawa, presents a system developed to help people explore local communities by providing navigation services in social spaces created by the community members via communication and knowledge sharing. The proposed system utilizes data of a community's social network to reconstruct the social space, which is otherwise not physically perceptible but imaginary, experiential, and yet learnable. The social space is modeled with an agent network, where each agent stands for a member of the community and has knowledge about expertise and personal characteristics of some other members. An agent can gather information, using its social "connections," to find community members most suitable to communicate to in a specific situation defined by the system's user. The system then deploys its multimodal interface, which operates with 3D graphics and haptic virtual environments and "maps" the social space onto a representation of the relevant physical space, to locate the potential interlocutors and advise the user on an efficient communication strategy for the given community. A prototype of the system is built and used in an experiment. The study results are discussed in a context of related work, conclusions are drawn, and implications for future research are formulated.

Chapter 20, "Symbiotic Aspects in e-Government Application Development," by Claude Moulin and Marco Luca Sbodio, presents e-Government applications where the symbiotic aspect must be taken into account at three stages: at design time in order to integrate the end-user, at delivery time when civil servants have to discover and interact with new services, and at run time when ambient intelligence could help the interaction of citizens with specific services. In this chapter, we focus on the first two steps. We show how interoperability issues must concern application designers. We also present how semantics can help civil servants when they have to deal with e-government service frameworks. We then describe an actual application developed during the European Terregov project where semantics is the key point for simplifying the role of citizens when requesting for health care services.

Chapter 21, "CoPBoard: A Catalyst for Distributed Communities of Practice," by Gilson Yukio Sato and Jean-Paul A. Barthès, presents symbiotic computing to a proliferation of computing devices that allow linking people, favoring the development of distributed Communities of Practice (CoPs). Their members, being dispersed geographically, have to rely strongly on technological means to interact. In this context, coordinating distributed CoPs is more challenging than coordinating their collocated counterparts. Hence, the increasing role of the coordination should be supported by an adequate set of coordination tools. In this chapter we present an approach based on multi-agent systems for coordinating distributed CoPs. It includes analyzing the exchanges among members and translating this information into a graphical format in order to help the coordinators to follow the evolution of the participation and the domain of the community.

Chapter 22, "Remote Conversation Support for People with Aphasia," by Nilar Aye, Takuro Ito, Fumio Hattori, Kazuhiro Kuwabara, and Kiyoshi Yasuda, presents a remote conversation support system for people with aphasia. The aim of our system is to improve the Quality of Lives (QoL) of people suffering cognitive disabilities. In this framework, a topic list is used as a conversation assistant in addition to the video phone. The important feature is sharing the focus of attention on the topic list between a patient and the communication partner over the network in order to facilitate distant communication. The results of two preliminary experiments indicate the potential of the system.

Chapter 23, "Estimating which Object Type a Sensor Node is Attached to in Ubiquitous Sensor Environment," by Takuya Maekawa, Yutaka Yanagisawa, and Takeshi Okadome, presents a method for inferring the type of the physical indoor objects and the states they are in with attached sensors. Assuming that an object has its own states that have transitions represented by a state transition diagram, we prepare the state transition diagrams in advance for such indoor objects as a door, a drawer, a chair, and a locker. The method determines the presumed state transition diagram from prepared diagrams that match sensor data collected from people's daily living for a certain period of time. A two week experiment shows that the method achieves high accuracy of inferring objects to which sensor nodes are attached. The method makes it easy for us to introduce ubiquitous sensor environments by simply attaching sensor nodes to physical objects around us.

Chapter 24, "Delay-Range-Dependent Robust Stability for Uncertain Stochastic Neural Networks with Time-Varying Delays," by Wei Feng and Haixia Wu, presents a robust stability analysis problem for uncertain stochastic neural networks with interval time-varying delays. By utilizing a Lyapunov-Krasovskii functional and conducting stochastic analysis, we show that the addressed neural networks are globally, robustly, asymptotically stable if a convex optimization problem is feasible. Some stability criteria are derived for all admissible uncertainties. And these stability criteria are formulated by means of the feasibility of a Linear Matrix Inequality (LMI), which can be effectively solved by some standard numerical packages. Five numerical examples are given to demonstrate the usefulness of the proposed robust stability criteria.

Chapter 25, "Classifier Ensemble Based Analysis of a Genome-Wide SNP Dataset Concerning Late-Onset Alzheimer Disease," by Lúcio Coelho, Ben Goertzel, Cassio Pennachin, and Chris Heward, presents the OpenBiomind toolkit that is used to apply GA, GP, and local search methods to analyze a large SNP dataset concerning Late-Onset Alzheimer's Disease (LOAD). Classification models identifying LOAD with statistically significant accuracy are identified, and ensemble-based important features analysis is used to identify brain genes related to LOAD, most notably the solute carrier gene SLC6A15. Ensemble analysis is used to identify potentially significant interactions between genes in the context of LOAD.

This book is intended for researchers, engineers, graduate students, senior-level undergraduate students, and instructors as an informative reference book in the emerging fields of software science, cognitive intelligence, and computational intelligence. The editor expects that readers of *Breakthroughs in Software Science and Computational Intelligence* will benefit from the 25 selected chapters of this book, which represents the latest advances in research in software science and computational intelligence and their engineering applications.

Yingxu Wang
University of Calgary, Canada