EDITORIAL NOTE

Convergence of Software Science and Computational Intelligence: A New Transdisciplinary Research Field

Yingxu Wang, Editor-in-Chief, IJSSCI

ABSTRACT

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 editorial addresses the objectives of the International Journal of Software Science and Computational Intelligence (IJSSCI), and explores the domain of the emerging discipline. The historical evolvement of software and intelligence sciences and their theoretical foundations are elucidated. The coverage of this inaugural issue and recent advances in software and intelligence sciences are reviewed. This editorial demonstrates that the investigation into software and intelligence sciences will result in fundamental findings toward the development of future generation computing theories, methodologies, and technologies, as well as novel mathematical structures.

Keywords: artificial intelligence; brain science; computational intelligence; cognitive science; cybernetics; denotational mathematics; intelligence science; knowledge engineering; natural intelligence; neural informatics; new computing methodologies; software engineering; software science

INTRODUCTION

The latest developments in computer science, theoretical software engineering, cognitive science, cognitive informatics, and intelligence science, and the crystallization of accumulated knowledge by the fertilization of these areas, have led to the emergence of a transdisciplinary and convergence field known as software and intelligence sciences. The coverage of the International Journal of Software Science and Computational Intelligence (IJSSCI) includes theories, methodologies, technologies of software science and engineering, denotational mathematics, and their applications in engineering and industries.

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 [Wang, 2007a]. 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 [Wilson and Frank, 1999; Wang, 2008a, 2009a]. The convergence of software science and intelligent science forms the transdisciplinary field of computational intelligence, which provides a coherent set of fundamental theories, contemporary denotational mathematics, and engineering applications.

This editorial addresses the objectives of the International Journal of Software Science and Computational Intelligence (IJSSCI), and explores the domain of the emerging discipline. The historical evolvement of software and intelligence sciences and their theoretical foundations are elucidated. The coverage of this inaugural issue and recent advances in software and intelligence sciences are highlighted. This editorial demonstrates that the investigation into software and intelligence sciences will result in fundamental findings toward the development of future generation computing theories, methodologies, and technologies, as well as novel mathematical structures.

THE EMERGENCE OF 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 the laws constrain it 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 will software science emerge and grow in the field of software, computer, information, knowledge, and system engineering [Wang, 2007a].

This section provides perspectives on the emerging discipline of software science along with the maturity of software engineering theories and methodologies in fundamental research. The architecture and roadmap of software science are presented. The theoretical framework, mathematical foundations, and basic methodologies of software science will be briefly introduced.

What is Software Science?

**Definition 1.** Software Science 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 the engineering discipline that applies software science theories and methodologies to efficiently, economically, and reliably organize and develop large-scale software systems.

The relationship between software science and software engineering can be analogized to those of theoretical physics and applied physics, or dynamics and mechanical engineering. Without theoretical physics there would be no matured applied physics; without dynamics there would be no matured mechanical engineering. So is software science with software engineering.

The phenomena 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.

The disciplines of mathematics and physics are successful paradigms that adopt the formal framework of theoretical knowledge, which have the advantages of *stability* and *efficiency*. The former is a property of formal knowledge that once it is established and proven, users who refer to it will no longer need to reexamine or reprove it. The latter is a property of formal knowledge that is exclusively true or false that saves everybody’s time to argue a proven theory. In contrasting the nature of theoretical and empirical knowledge, the following principle can be derived.
Theorem 1. The rigorous levels of empirical and theoretical knowledge states that an empirical truth is a truth based on or verifiable by observations, experiments, or experiences. In contrary, a theoretical proposition is an assertion based on formal theories, logical, or mathematical inferences.

Based on Theorem 1, a corollary on application domains of theoretical and empirical knowledge is stated as follows.

Corollary 1. The validation scope of theoretical knowledge is universal in its domain such as \( \forall x \in S \Rightarrow p(x) \); while the validation scope of empirical knowledge is based on limited observations such as \( \exists x \in S \Rightarrow p(x) \), where \( S \) is the discourse of a problem \( x \) under study, and \( p \) a proven proposition or derived theory on \( x \).

The differences of the validated domains between theoretical and empirical knowledge indicates the levels of refinements of a given form of knowledge and its reliability. Huge empirical knowledge were reported and then disappeared over time. For example, there are tons of empirical knowledge on software engineering published each year in the last decades. However, those that would be included in a textbook on software engineering theories as proven and general truth, rather than specific cases partially working on certain given or nonspecified constraints, would be no more than a few handful pages. According to Corollary 1, the major risk of empirical knowledge is its uncertainty when applying in a different environment, even the same environment but at different time, because empirical knowledge and common sense are often error-prone. The differences of the validated domains between theoretical and empirical knowledge indicate the levels of refinements of different forms of knowledge and their reliability.

Empirical knowledge answers how; while theoretical knowledge reveals why. Theoretical knowledge is a formalization of generic truth and proven empirical knowledge. Although the discovery and development of a theory or a law may take decades even centuries, its acquisition and dissemination are much easier and faster with ordinary effort. However, empirical knowledge is very difficult to be indirectly gained. One may acquire knowledge in multiple scientific disciplines such as those offered at a university, but may not be an expert in multiple engineering disciplines such as in all areas of electrical, mechanical, chemical, and computer engineering. The reasons behind this are that each engineering area requires specific empirical knowledge, skills, and tools. All of them need a long period of training and practice to be an expert.

Architecture of Software Science

The architecture of software science can be classified into four categories namely theories and methodologies, denotational mathematics, cognitive informatics, and organizational theories as shown in Fig. 1.

In the framework of software science, theories and methodologies encompass system modeling and refinement methodologies, computing theories, formal linguistic theories, and software code generation theories. All forms of imperative, autonomic, and cognitive computing theories as well as their engineering applications are explored in this category.

Denotational mathematics [Wang, 2002a, 2008b] for software science is the enquiry for its mathematical foundations in the forms of formal inference methodologies, concept algebra, system algebra, and Real-Time Process Algebra (RTPA). In the contemporary mathematics for software science and software engineering, concept algebra is designed to deal with the to be problems and knowledge manipulation [Wang, 2008c]. System algebra is developed to formally treat the to have problems in terms of dynamic relations and possessions beyond set theory [Wang, 2008d]. RTPA is adopted to formalize the to do problems such as system architectures, static and dynamic behaviors [Wang, 2002b, 2007a, 2008e]. Further discussion on denotational mathematics for software science may be referred to [Wang, 2008b].

Cognitive informatics for software science encompasses intelligence science, neural informatics, knowledge science, and computational intelligence [Wang, 2002a, 2006, 2007b; Wang et al., 2006]. Cognitive informatics explains the fundamental mechanisms of natural intelligence and its products in terms of information and knowledge. It also studies the software implemen-
tation of intelligent behaviors by computational intelligence. Advances in cognitive informatics will help to overcome the cognitive barriers and inherited complicities in software engineering, which is called the intellectually manageability by Dijkstra (1976) and the essential difficulties by Brooks (1975) in software engineering.

Organizational theories of software science encompass coordinative work organization theories, management theories, economics theories, and system/sociology theories. The organizational facet of software science studies how large-scale software engineering projects may be optimally organized and what the underpinning laws are at different levels of coordinative complexities [Wang, 2007a].

Software Science: Theoretical Foundations for Software Engineering and Computational Intelligence

It is recognized that theoretical software engineering focuses on foundations and basic theories of software engineering; whilst empirical software engineering concentrates on heuristic principles, tools/environments, and best practices by case studies, experiments, trials, and benchmarking. It is noteworthy that, because software is the most abstract instructive information, software engineering is one of the most complicated branches of engineering, which requires intensive theoretical investigations rather than only empirical studies. Due to the widely impacted and applicable objects and the complicated theories in software engineering, a scientific discipline known as software science is emerged.

The discipline of software science enquires the common objects in the abstract world such as software, information, data, knowledge, instruction, executable behavior, and their processing by natural and machine intelligence. In other words, software science studies instructive and behavioral information and the mechanism of its translation into system behaviors. It is noteworthy that cognitive informatics perceives information as anything that can be inputted into and processed by the brain; while software science perceives software as any instructive information that can be executed and transformed into computational behaviors by computers. This forges a relationship between cognitive informatics and software science, which indicates that the former is the foundation for natural intelligence science, and the latter is the foundation for artificial intelligence science and software engineering. With the perception as applied software science, software engineering is an engineering discipline that applies software science theories and methodologies to efficiently, economically, and reliably organize and develop large-scale software systems.
FROM ARTIFICIAL INTELLIGENCE TO COMPUTATIONAL INTELLIGENCE

Intelligence science is naturally interdisciplin-
ary in vertical based on reductionism and is transdisciplinary in horizontal based on holism.
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.

Approaches to Embody Abstract Intelligence

A fundamental view developed in software and
intelligence sciences is known as abstract intel-
ligen [Wang, 2008a], which provides a unified
foundation for the studies of all forms and para-
digms of intelligence such as natural, artificial,
machinable, and computational intelligence.

Definition 2. Abstract intelligence, αI, is a human
enquiry of both natural and artificial intelligence
at the embody levels of neural, cognitive, func-
tional, and logical from the bottom up.

In the narrow sense, αI is a human or a
system ability that transforms information into
behaviors. While, in the broad sense, αI is any
human or system ability that autonomously trans-
fers the forms of abstract information between
data, information, knowledge, and behaviors in
the brain or systems.

With the clarification of the intension and
extension of the concept of the generic abstract
intelligence, its paradigms or concrete forms in
the real-world can be derived as summarized
in Table 1. As shown in Table 1, computational
intelligence can be defined below.

Definition 3. 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, autonomous computing, intelligent
agent systems, genetic/evolutionary systems, and
autonomous learning systems.

Cognitive Informatics Foundations
of Computational Intelligence

It is recognized that the theoretical foundations
of computational intelligence root in cognitive
informatics, software science, and denotational
mathematics [Wang, 2002a, 2008b]. Cognitive
informatics is a cutting-edge and multidisci-
plinary research field that tackles the fundamental
problems shared by modern informatics, compu-

Table 1. Taxonomy of abstract intelligence and its embodying forms

<table>
<thead>
<tr>
<th>No.</th>
<th>Form of intelligence</th>
<th>Embodying Means</th>
<th>Paradigms</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Natural intelligence (NI)</td>
<td>Naturally grown biological and physiological organisms</td>
<td>Human brains and brains of other well developed species</td>
</tr>
<tr>
<td>2</td>
<td>Artificial intelligence (AI)</td>
<td>Cognitively-inspired artificial models and man-made systems</td>
<td>Intelligent systems, knowledge systems, decision-making systems, and distributed agent systems</td>
</tr>
<tr>
<td>3</td>
<td>Machinable intelligence (MI)</td>
<td>Complex machine and wired systems</td>
<td>Computers, robots, autonomic circuits, neural networks, and autonomic mechanical machines</td>
</tr>
<tr>
<td>4</td>
<td>Computational intelligence (CoI)</td>
<td>Computational methodologies and software systems</td>
<td>Expert systems, fuzzy systems, autonomous computing, intelligent agent systems, genetic/evolutionary systems, and autonomous learning systems</td>
</tr>
</tbody>
</table>
Cognitive informatics is the transdisciplinary enquiry of cognitive and information sciences that investigates into the internal information processing mechanisms and processes of the brain and natural intelligence, and their engineering applications via an interdisciplinary approach [Wang, 2002a, 2006, 2007b; Wang et al., 2006].


The key application areas of cognitive informatics can be divided into two categories. The first category of applications uses informatics and computing techniques to investigate cognitive science problems, such as memory, learning, and reasoning. The second category including the remainder areas uses cognitive theories to investigate problems in informatics, computing, software/knowledge engineering and computational intelligence. Cognitive informatics focuses on the nature of information processing in the brain, such as information acquisition, representation, memory, retrieve, generation, and communication. Through the interdisciplinary approach and with the support of modern information and neuroscience technologies, mechanisms of the brain and the mind may be systematically explored within the framework of cognitive informatics.

Recent advances in cognitive informatics reveal an entire set of cognitive functions of the brain [Wang, 2007b; Wang and Wang, 2006] and their cognitive process models [Wang et al., 2006]. LRMB [Wang et al., 2006] provides a reference model for the design and implementation of computational intelligence, which provides a systematical view toward the formal description and modeling of architectures and behaviors of computational intelligence. The LRMB model explains the functional mechanisms and cognitive processes of the natural intelligence with 39 cognitive processes at seven layers known as the sensation, memory, perception, action, meta-cognitive,meta-inference, and higher cognitive layers from the bottom up. LRMB elicits the core and highly repetitive recurrent cognitive processes from a huge variety of life functions, which may shed light on the study of the fundamental mechanisms and interactions of complicated mental processes and computational intelligence, particularly the relationships and interactions between the inherited and the acquired life functions as well as those of the subconscious and conscious cognitive processes. The cognitive model of the brain can be used as a reference model for goal- and inference-driven technologies in computational intelligence and autonomous agent systems [Wang, 2009b].

**Denotational Mathematics**

**Foundations of Computational Intelligence**

Applied mathematics can be classified into two categories known as analytic and denotational mathematics [Wang, 2007a, 2008b]. The former are mathematical structures that deal with functions of variables as well as their operations and behaviors; while the latter are mathematical structures that formalize rigorous expressions and inferences of system architectures and behaviors with abstract concepts, complex relations, and dynamic processes.

**Definition 5.** Denotational mathematics is a category of expressive mathematical structures that deals with high-level mathematical entities beyond numbers and simple sets, such as abstract objects, complex relations, behavioral
information, concepts, knowledge, processes, intelligence, and systems.

Typical paradigms of denotational mathematics are comparatively presented in Table 2, where their structures, mathematical entities, algebraic operations, and usages are contrasted. The paradigms of denotational mathematics as shown in Table 2 are concept algebra [Wang, 2008c], system algebra [Wang, 2008d], and Real-Time Process Algebra (RTPA) [Wang, 2002b, 2008e].

The emergence of denotational mathematics is driven by the practical needs in cognitive informatics, computational intelligence, computing science, software science, and knowledge engineering, because all these modern disciplines study complex human and machine behaviors and their rigorous treatments. Among the new forms of denotational mathematics, concept algebra is designed to deal with the abstract mathematical structure of concepts and their representation and manipulation in knowledge engineering. System algebra is created to the rigorous treatment of abstract systems and their algebraic relations and operations. RTPA is developed to deal with series of behavioral processes and architectures of human and systems.

Denotational mathematics provides a powerful mathematical means for modeling and formalizing computational intelligent systems. Not only the architectures of computational intelligent systems, but also their dynamic behaviors can be rigorously and systematically manipulated by denotational mathematics. A wide range of applications of denotational mathematics have been demonstrated in software science and computational intelligence, which demonstrate that denotational mathematics is an ideal mathematical means for dealing with concepts, knowledge, behavioral processes, and human/machine intelligence with real-world problems.

<table>
<thead>
<tr>
<th>No.</th>
<th>Paradigm</th>
<th>Structure</th>
<th>Mathematical entities</th>
<th>Usage</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Concept algebra</td>
<td>(\mathcal{C}(\mathcal{O}, \mathcal{A}, \mathcal{R}_c, \mathcal{R}_i, \mathcal{R}_o))</td>
<td>({\cdot\mid \mathcal{O}, \mathcal{A}, \mathcal{R}_c, \mathcal{R}_i, \mathcal{R}_o})</td>
<td>Algebraic manipulations on abstract concepts</td>
</tr>
<tr>
<td>2</td>
<td>System algebra</td>
<td>(\mathcal{S}(\mathcal{C}, \mathcal{R}_c, \mathcal{R}_i, \mathcal{R}<em>o, \mathcal{B}, \mathcal{R}</em>+, \mathcal{B}_p, \mathcal{B}_q))</td>
<td>({\cdot\mid \mathcal{C}, \mathcal{R}_c, \mathcal{R}_i, \mathcal{R}_o, \mathcal{B}_p, \mathcal{B}_q})</td>
<td>Algebraic manipulations on abstract systems</td>
</tr>
<tr>
<td>3</td>
<td>Real-time process algebra (RTPA)</td>
<td>(\mathcal{R}(\mathcal{O}, \mathcal{P}, \mathcal{E}))</td>
<td>({\cdot\mid \mathcal{O}, \mathcal{P}, \mathcal{E}})</td>
<td>Algebraic manipulations on abstract processes</td>
</tr>
</tbody>
</table>
A NEW TRANSDISCIPLINARY RESEARCH FIELD OF SOFTWARE AND INTELLIGENCE SCIENCES

The transdisciplinary field between software science and computational intelligence brings two logically and interactively related disciplines together. This new field of enquiry will be helpful to explain how natural intelligence is generated on the basis of fundamental biological and physiological structures; How intelligence functions logically and physiologically; How natural and machine intelligence are converged on the basis of software and intelligence sciences.

The architectural framework of software and intelligence science is described in Table 3, which illustrates the structure and scope of the International Journal of Software Science and Computational Intelligence (IJSSCI). Because the implementation media and embody means of computational intelligence are software or its instructive behaviors, to a certain extent, computational intelligence may be perceived as software intelligence, or shortly intelware in parallel to hardware and software. Typical paradigms of computational intelligence are expert systems, fuzzy systems, autonomous computing, intelligent agent systems, genetic/evolutionary systems, and autonomous learning systems.

As that of computing hardware is based on the mathematical foundation of Boolean algebra, the more intelligent capability of computational intelligence must be processed by more powerful mathematical structures known as denotational mathematics in the forms of concept algebra, system algebra, and RTPA as described in the preceding sections. The three new structures of contemporary mathematics extend the abstract objects under study in mathematics from basic entities such as numbers, Boolean variables, and sets to complex ones such as concepts, systems, and behavioral processes.

Computing systems and technologies can be classified into the categories of imperative, autonomic, and cognitive computers from the bottom up. The imperative computers are a traditional and passive system based on stored-program controlled behaviors for data processing [von Neumann, 1946, 1958]. The autonomic computers are goal-driven and self-decision-driven machines that do not rely on instructive and procedural information [Kephart and Chess, 2003; IBM, 2006; Wang, 2004, 2007a]. Cognitive computers are more intelligent computers beyond the imperative and autonomic computers, which embodies major natural intelligence behaviors of the brain such as thinking, inference, and learning [Wang, 2006, 2007a; Wang and Sheu, 2008].

Definition 6. A cognitive computer is an intelligent knowledge processor with the capabilities of autonomic inference and perception that mimics the mechanisms of the brain and abstract intelligence.

Cognitive computers are an expected paradigm of computational intelligence. The theories and methodologies of cognitive computers are inspired by the latest advances in cognitive informatics [Wang, 2002a, 2006, 2007b] and contemporary denotational mathematics [Wang, 2002b, 2008b]. The theoretical foundations of cognitive computers encompass cognitive informatics, neural informatics, and abstract intelligence. As that of formal logic and Boolean algebra are the mathematical foundations of conventional computers. The mathematical foundations of cognitive computers are based on denotational mathematics. Cognitive computers will provide a powerful platform to implement all facets of computational intelligence such as the perceptive, cognitive, instructive, and reflective intelligence [Wang, 2008a, 2009a].

HIGHLIGHTS OF THE INAUGURAL ISSUE

IJSSCI will not only establish an important forum for the emerging field of software and intelligence sciences, but also facilitate the dissemination of engineering and industrial applications of the latest discoveries in the fields covered by the journal. The objective of this inaugural issue of IJSSCI is to provide an informative overview on the entire structure of software science and computational intelligence, and an in-depth survey of the latest advances in these fields from multidisciplinary researchers and practitioners.
### Table 3. The structure and scopes of IJSSCI

<table>
<thead>
<tr>
<th>Theories and Methodologies of Software and Intelligence Sciences</th>
<th>Denotational Mathematics for Software and Intelligence Sciences</th>
<th>Applications of Software and Intelligence Sciences in Engineering and Industries</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Transdisciplinary theories shared by software and intelligence science</td>
<td>• Denotational vs. analytic mathematics</td>
<td>• Future generation computers</td>
</tr>
<tr>
<td>• Functional models of the brain</td>
<td>• Mathematical structures for software modeling</td>
<td>• Cognitive computers</td>
</tr>
<tr>
<td>• Logical models of the brain</td>
<td>• Formal description of the brain</td>
<td>• Soft computing</td>
</tr>
<tr>
<td>• Software simulations of the brain</td>
<td>• Formal description of cognitive processes</td>
<td>• Intelligent software engineering</td>
</tr>
<tr>
<td>• Theories for computational intelligence</td>
<td>• Formal inference processes</td>
<td>• Autonomic/autonomous systems</td>
</tr>
<tr>
<td>• Software vs. brain processes</td>
<td>• Concept algebra for knowledge modeling</td>
<td>• Autonomous machine learning systems</td>
</tr>
<tr>
<td>• Formal language theories</td>
<td>• Process algebra for behavioral modeling</td>
<td>• Autonomous agent systems</td>
</tr>
<tr>
<td>• Cognitive informatics foundations of the brain</td>
<td>• System algebra for complex system modeling</td>
<td>• Software code generation technologies</td>
</tr>
<tr>
<td>• Intelligent behavioral foundations of software</td>
<td>• Real-Time Process Algebra (RTPA)</td>
<td>• Hyper programming</td>
</tr>
<tr>
<td>• Instructive information foundations of software</td>
<td>• Visual semantic algebra</td>
<td>• Hybrid man-machine systems</td>
</tr>
<tr>
<td>• Non-language centered programming</td>
<td>• Fuzzy/rough sets</td>
<td>• Novel computing methods</td>
</tr>
<tr>
<td>• Cognitive informatics</td>
<td>• Mathematical foundations of software</td>
<td>• Novel intelligence simulation systems</td>
</tr>
<tr>
<td>• Cognitive mechanisms of the brain and mind</td>
<td>• Universal mathematic models of software</td>
<td>• Novel memory devices</td>
</tr>
<tr>
<td>• Neural informatics</td>
<td>• Mathematical models of the brain and mind</td>
<td>• Cognitive complexity of software</td>
</tr>
<tr>
<td>• Knowledge representation methodologies</td>
<td>• Mathematical models of machine intelligence</td>
<td>• Cognitive machines that think and feel</td>
</tr>
<tr>
<td>• Autonomous computing</td>
<td>• Mathematical models of natural intelligence</td>
<td>• Granular computing</td>
</tr>
<tr>
<td>• Brain behavioral simulation</td>
<td>• Mathematical models of learning</td>
<td>• Bioinformatics</td>
</tr>
<tr>
<td>• Distributed intelligence</td>
<td>• Mathematical models of problem solving</td>
<td>• Machine inferences</td>
</tr>
<tr>
<td>• Machine perception and cognition</td>
<td>• Mathematical models of agent systems</td>
<td>• Novel memory forms and implementations</td>
</tr>
</tbody>
</table>

This inaugural issue includes eight regular research articles from prestigious scholars as highlighted below:

a) Yingxu Wang introduces the concept of abstract intelligence in the work *On Abstract Intelligence: Toward a Unifying Theory of Natural, Artificial, Machinable, and Computational Intelligence*, which presents the Generic Abstract Intelligence Model (GAIM) in order to explain the mechanisms of advanced intelligence and their denotational
mathematical foundations.

b) Witold Pedrycz elaborates *Hierarchies of Architectures of Collaborative Computational Intelligence*, which investigates into the hierarchical and collaborative properties of computational intelligence as known as Collaborative Computational Intelligence (C²I). A paradigm of information granules in granular computing are modeled and explained in the form of metastructures and metamodels of C²I.

c) Eric Bouillet, Mark Feblowitz, Zhen Liu, Anand Ranganathan, and Anton Riabov from IBM T.J. Watson Research Center address a long-standing and highly challenging problem on *Semantic Matching, Propagation and Transformation for Composition in Component-Based Systems*. The work demonstrates how application architects craft components for dynamic assembly, and how they craft semantic descriptions using ontology that captures the essence of components’ functionality to support composition in software engineering.

d) Jeffrey J.P. Tsai, Jia Zhang, Jeff J.S. Huang and Stephen J.H. Yang reported their finding in *Supporting CSCW and CSCL with Intelligent Social Grouping Sensors*. A hierarchical social network is constructed to model both knowledge and social relationships, where the former is the extent of knowledge of a participant; while the latter is the capability of a participant to share knowledge with peers. The model is adopted to measure the collaboration strength between pairs of participants in social sensor networks in the contexts of Computer Supported Cooperative Work (CSCW) and Computer-Supported Collaborative Learning (CSCL).

e) Yingxu Wang, Lotfi A. Zadeh, and Yiyu Yao develop a new approach to formally model the entities and methodologies of granular computing using system algebra in the work *On the System Algebra Foundation for Granular Computing*. It focuses not only on the architectural and relational modeling of system granules in granular computing, but also their computing behavioral modeling, which enables implementation of computable and distributed granules and granular computing systems.

f) Marina Gavrilova presents an intensive survey on *Adaptive Computation Paradigm in Knowledge Representation: Traditional and Emerging Applications*. The author finds that many fundamental needs in computing drive the investigation into convergence of software science and intelligence science, as well as computational sciences and their applications. A number of emerging paradigms of adaptive computation, such as the algorithmic models of intelligence, biometric technologies, evolutionary computing, swarm intelligence, knowledge representation, and geometric computing, are comparatively elaborated.

h) Janusz Kacprzyk and Sławomir Zadrożny investigate into *Protoforms of Linguistic Database Summaries as a Human Consistent Tool for Using Natural Language in Data Mining*. The authors advocate the use of Zadeh’s concept of the prototypical form, shortly protoform, in linguistic database and their extensions as a general tool for consistent summarization of large data sets. They present an extended interactive approach to fuzzy linguistic summaries based on fuzzy logic and fuzzy database queries with linguistic quantifiers. This work shows how fuzzy queries are related to linguistic summaries by a hierarchy of protoforms. An implementation of the proposed technology for the summarization of Web server logs is demonstrated.

g) Du Zhang’s work on *Machine Learning and Value-Based Software Engineering* synergizes a new link between computational intelligence and software intelligence. The author perceives that machine learning plays an increasingly important role in helping develop and maintain large and complex software systems. A set of machine learning methods and algorithms to enable value-based software engineering are proposed, which integrates value considerations into the full range of existing and emerging software engineering practices.

As an introductory orientation, the editorial of this inaugural issue highlights the architecture and a coherent framework of the convergence of software science and computational intelligence,
which form a new transdisciplinary field that investigates into a unifying theory for abstract intelligence and its paradigms in forms of natural, artificial, machinable, computational, and web-based distributed intelligence.

CONCLUSION

The emerging field of software and intelligence sciences investigates into the theoretical foundations and denotational mathematical structures of computational intelligence and software engineering. This editorial has provided insightful perspectives on the convergent field of software science and computational intelligence. The coverage of the inaugural issue has been reviewed and highlighted. The Editor-in-Chief expects that readers of the International Journal of Software Science and Computational Intelligence (IJSSCI) will benefit from the set of articles presented in this inaugural issue in order to aware the recent advances and groundbreaking studies in the transdisciplinary field of software and intelligence sciences.

ACKNOWLEDGMENT

Many people have contributed their dedicated work to the establishment of IJSSCI and the publication of its inaugural issue. The Editor-in-Chief would like to thank all authors, the associate editors, and the editorial board members, particularly Loﬁ A. Zadeh, Witold Pedrycz, James Anderson, Janusz Kacprzyk, Witold Kinsner, Philip Sheu, Jeffrey Tsai, Du Zhang, Yiyu Yao, Dilip Patel, Christine Chan, Keith Chan, Ling Guan, and Zhen Liu, for their great contributions to IJSSCI. I would like to acknowledge the publisher of IJSSCI, IGI Publishing, USA, and to thank Mehdi Khosrow-Pour (President of IGI), Jan Travers (Managing Director), Kristin M. Klinger (Managing Acquisitions Editor), and the editorial staff of IGI, Heather Probst and Elizabeth C. Duke, for their excellent professional support.

REFERENCES


Wang, Y. (2007c), The OAR Model of Neural Informatics for Internal Knowledge Representation in the


Yingxu Wang is professor of cognitive informatics and software engineering, director of International Center for Cognitive Informatics (ICICI), and director of Theoretical and Empirical Software Engineering Research Center (TESEREC) at the University of Calgary. He received a PhD in software engineering from The Nottingham Trent University, UK, in 1997, and a BSc in electrical engineering from Shanghai Tiedao University in 1983. He was a visiting professor in the Computing Laboratory at Oxford University and Dept. of Computer Science at Stanford University during 1995 and 2008, respectively, and has been a full professor since 1994. He is founding editor-in-chief of International Journal of Cognitive Informatics and Natural Intelligence (IJ-CINI), founding editor-in-chief of International Journal of Software Science and Computational Intelligence (IJSSCI), associate editor of IEEE TSMC(A), and editor-in-chief of CRC Book Series in Software Engineering. He has published over 300 peer reviewed journal and conference papers and 11 books in cognitive informatics, software engineering, and computational intelligence. He has won dozens of research achievement, best paper, and teaching awards in the last 28 years.