

Fusion of Soft Computing and Hard Computing: Computational Structures and Characteristic Features

Seppo J. Ovaska, *Senior Member, IEEE*, Akimoto Kamiya, *Senior Member, IEEE*,
and YangQuan Chen, *Senior Member, IEEE*

Abstract—Soft computing (SC) and hard computing (HC) methodologies are fused together successfully in numerous industrial applications. The principal aim is to develop computationally intelligent hybrid systems that are straightforward to analyze, with highly predictable behavior and stability, and with computational burden that is no more than moderate. All these goals are particularly important in embedded real-time applications. This paper is intended to clarify the present vagueness related to the fusion of SC and HC methodologies. We classify the different fusion schemes to 12 core categories and six supplementary categories, and discuss the characteristic features of SC and HC constituents in practical fusion implementations. The emerging fusion approach offers a natural evolution path from pure hard computing toward dominating soft computing.

Index Terms—Computationally intelligent hybrid systems, evolutionary computation (EC), fuzzy logic (FL), hard computing (HC), neural network (NN), soft computing (SC).

I. INTRODUCTION

SOFT COMPUTING (SC) methodologies are penetrating steadily into industrial and commercial applications, not only in Japan and South Korea, but also in the U.S. and Europe. In the majority of such applications, SC is hidden inside systems or subsystems, and the end user does not necessarily know that soft computing methods are being used for control, fault diagnosis, pattern recognition, signal processing, etc. This is the case when SC is utilized for complementing the performance of traditional hard computing (HC) algorithms, or even replacing them. Another class of applications uses soft computing for implementing computationally intelligent and user-friendly features that could not be realized competitively by hard computing alone. Professor Zadeh's original definition of soft computing is quoted below from <http://www.cs.berkeley.edu/~mazlack/BISC/BISC-DBM-soft.html>:

Soft computing differs from conventional (hard) computing in that, unlike hard computing, it is tolerant of imprecision, uncertainty, partial truth, and approximation. In effect, the role model for soft computing is the human mind. The guiding principle of soft computing is: Exploit the

Manuscript received June 16, 2004; revised November 17, 2004. This paper was recommended by Associate Editor N. O. Attoh-Okine.

S. J. Ovaska is with the Helsinki University of Technology, Department of Electrical and Communications Engineering, FI-02150 Espoo, Finland (e-mail: ovaska@ieee.org).

A. Kamiya is with the Koshiro National College of Technology, Department of Information Engineering, Koshiro 084-0916, Japan (e-mail: kamiya@kushiroct.ac.jp).

Y. Chen is with the Department of Electrical and Computer Engineering, Utah State University, Logan, UT 84322-4120 USA (e-mail: yqchen@ece.usu.edu).

Digital Object Identifier 10.1109/TSMCC.2005.855528

tolerance for imprecision, uncertainty, partial truth, and approximation to achieve tractability, robustness and low solution cost.

At this juncture, the principal constituents of soft computing (SC) are fuzzy logic (FL), neural computing (NC), genetic computing (GC) and probabilistic reasoning (PR), with the latter subsuming belief networks, chaos theory and parts of learning theory. What is important to note is that soft computing is not a mélange. Rather, it is a partnership in which each of the partners contributes a distinct methodology for addressing problems in its domain. In this perspective, the principal constituent methodologies in SC are complementary rather than competitive.

On the other hand, hard computing is defined as the antipode of soft computing, i.e., a heterogeneous collection of traditional computing methods.

One important reason behind the growing industrial acceptance of SC is the existence of advantageous combinations of individual soft computing methodologies such as fuzzy-neuro, genetic algorithm (GA) like fuzzy, and neuro-GA [1], [2]. From the applications point of view, those combined constituents of soft computing are often more efficient than pure evolutionary computation, fuzzy logic, or neural networks. In addition, soft computing methods are commonly fused to traditional hard computing techniques in industrial products, instead of using SC alone. While academic researchers typically prefer *either* SC *or* HC methodologies, the complementary fusion of soft computing and hard computing is, increasingly, a natural way of thinking for practicing engineers, who are facing demanding real-world problems which must be solved competitively.

Goldberg presented three principal requirements to the research community that are essential in making soft computing widespread among different applications [24]:

- 1) scalable results;
- 2) practicality;
- 3) little models or theory.

In addition, he emphasized the importance of more complex integration of individual soft computing methods in developing autonomously intelligent systems. We would definitely include the complementary fusion of SC and HC in that requirements list.

There are several ideas and informal definitions of the concept "fusion of soft computing and hard computing" in the researchers' and engineers' minds, but none of the published ones has yet obtained a dominating position. Our specific definition of this emerging concept can be stated as follows.

An algorithm-level or system-level union of particular soft computing and hard computing methodologies, in which either methodology is signal-, parameter-, or structure-wise dependent upon and receives functional reinforcement from the other.

The emerging role of the fusion of SC and HC was highlighted by Ovaska and VanLandingham [3]. A summarizing overview [4] and eight original contributions [5]–[12] discussing advanced applications were presented in [3]. The feedback from the research and development community that the guest editors received during and after the editing process was mostly positive. Criticism of this special issue focused primarily on the *vagueness* of the entire fusion concept, because, at that time, there was not available any systematical treatment on different fusion categories or their characteristic features in the context of integrated SC and HC. The work of Agarwal on combining neural and conventional paradigms for modeling, prediction, and control is an early step toward that direction [33]. On the other hand, in the context of pure soft computing, a landmark discussion on various fusion categories was presented by Hayashi and Umano in 1993 [13] (in Japanese); and on the characteristic features for knowledge acquisition by Furuhashi in 2001 [14]. An extended discussion of Hayashi's and Umano's fusion categories, including also genetic algorithms, is available in [1] (in English).

To strengthen the theoretical basis for the fusion of soft computing and hard computing, we will analyze both the structural categories and characteristic features of the core fusion topologies, and thus provide a similar framework for the fusion of SC and HC that already exists for the fusion or hybridization of individual soft computing methods [1], [14]. Our aim is to remove the criticized vagueness aspect mentioned above. The fusion concept needs to be recognized explicitly, because in that way we can highlight that there is much unused research and development (R&D) potential in the *interface section* between SC and HC methodologies. In the highly successful human brain, the left and right hemispheres are linked closely together. By following such a reasonable analogy, it could be anticipated that a high degree of interaction between soft and hard computing would lead to better implementation economy of intelligent systems, improved tolerance against incomplete or uncertain sensor data, enhanced learning and adaptation capabilities, as well as increased machine IQ. All those characteristics are needed for developing autonomously intelligent systems. Thus, the ultimate motivation behind this fusion discussion is to advance the world-wide use of SC in real-world applications.

Our paper is organized as follows. In Section II, we introduce the structural categories for the fusion of soft computing and hard computing, as well as provide a concise discussion on the procedure how these categories were identified. In Section III, we discuss the characteristic features of SC and HC constituents in a few representative fusion applications. Section IV gives a demonstrative classification of several published fusions of soft computing and hard computing in industrial applications. Finally, Section V closes this paper with a conclusion and a brief discussion on the acceptance of SC methodologies for industrial use.

II. STRUCTURAL CATEGORIES

By following the pragmatic thinking of Hayashi and Umano [13], and after analyzing a substantial number of application

TABLE I
STRUCTURAL CATEGORIES OF THE FUSION OF SC AND HC

#	Notation	Description	Fusion grade
1	SC&HC	SC and HC are isolated from each other	low
2	SC/HC	SC and HC are connected in parallel	moderate
3	SC\HC	SC with HC feedback	moderate
4	HC\SC	HC with SC feedback	moderate
5	SC-HC	SC is cascaded with HC	moderate
6	HC-SC	HC is cascaded with SC	moderate
7	HC=SC	HC-designed SC	high/very high
8	SC=HC	SC-designed HC	high/very high
9	HC+SC	HC-augmented SC	very high
10	SC+HC	SC-augmented HC	very high
11	HC//SC	HC-assisted SC	very high
12	SC//HC	SC-assisted HC	very high

TABLE II
DEFINITION OF FUSION GRADES

Fusion grade	Qualitative definition
Low	SC and HC in simultaneous or alternative use without explicit connections
Moderate	Connections on input/output signal level only
High	SC (or HC) used for determining values of internal parameters
Very high	HC (or SC) used for determining internal structure or generating/manipulating internal data

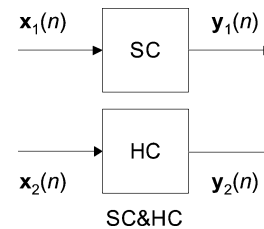


Fig. 1. Soft computing and hard computing without explicit connections.

papers, we will introduce 12 core categories for the fusion of soft computing and hard computing. They belong to the class of “little models or theory” that was emphasized by Goldberg in [24]. In his practical view, only the feature of little but sufficient models or theory is advancing the spread of soft computing, because R&D engineers prefer plain and uncomplicated solutions. Our core fusion categories are first summarized in Table I. The adopted qualitative measure, *fusion grade* [1], describes the strength of a particular connection between SC and HC constituents. Table II gives brief definitions of the employed fusion grades: low, moderate, high, and very high. Below is a short description of each core fusion category, as well as a discussion on six supplementary categories.

A. SC&HC

In SC and HC type systems (Fig. 1), the soft computing and hard computing constituents both have their independent roles, and there is no explicit connection between them. A descriptive example of such a loose union is an advanced elevator group dispatcher, where the primary call allocation algorithm is

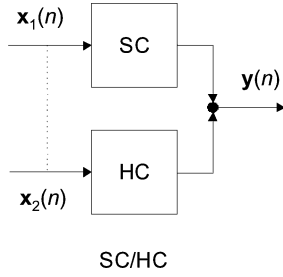


Fig. 2. Soft computing and hard computing in parallel. Dashed line shows a typical connection.

typically based on soft computing methods, while the backup algorithm, which becomes effective should the hall call interface have a serious failure, is usually a straightforward hard computing procedure; e.g., a finite state machine. In such cases, the fusion grade is naturally low, because there is no information exchange between the individual SC and HC blocks. The mathematical mappings realized by SC&HC are

$$\begin{cases} \mathbf{y}_1(n) = f_{SC}(\boldsymbol{\theta}_{SC}; \mathbf{x}_1(n)) \\ \mathbf{y}_2(n) = f_{HC}(\boldsymbol{\theta}_{HC}; \mathbf{x}_2(n)) \end{cases} \quad (1)$$

where $f_{SC}(\cdot; \cdot)$ and $f_{HC}(\cdot; \cdot)$ are arbitrary soft computing and hard computing algorithms, respectively, and $\boldsymbol{\theta}_{SC}$ and $\boldsymbol{\theta}_{HC}$ the corresponding system parameters; i.e., coefficients, weights, dimensions, etc. All the boldface symbols represent vectors in this discussion, and n is a discrete time index.

B. SC/HC

The second category, SC/HC, is presently of moderate practical interest. It offers possibilities for creating versatile combinations of SC and HC. In this parallel-connected topology, either soft computing is complementing the behavior and capabilities of a primary hard computing system, or *vice-versa* (Fig. 2). A typical example of such a configuration is an augmented linear controller (HC), where the nonlinearities of a plant are compensated by a parallel-connected SC-type controller. Such a closed-loop system could often be stabilized even without the supplementary SC controller, and the role of soft computing is merely to fine tune the control performance and provide additional robustness. Besides, such parallel soft computing and hard computing systems are often considerably easier to design and more efficient to implement than pure hard computing systems with comparable performance. In this important category, the fusion grade is moderate. The characteristic function of SC/HC is defined as

$$\mathbf{y}(n) = f_{SC}(\boldsymbol{\theta}_{SC}; \mathbf{x}_1(n)) \oplus f_{HC}(\boldsymbol{\theta}_{HC}; \mathbf{x}_2(n)) \quad (2)$$

where the merging operator, \oplus , denotes either addition or combining of data/signal vectors, i.e., $\mathbf{a} \oplus \mathbf{b}$ is either $\mathbf{a} + \mathbf{b}$ or $[\mathbf{a}\mathbf{b}]$, where \mathbf{a} and \mathbf{b} are arbitrary row vectors.

C. SC\HC and HC\SC

Also the feedback parallel connections, SC\HC and HC\SC, are currently of some applicable use. In these cases, the output

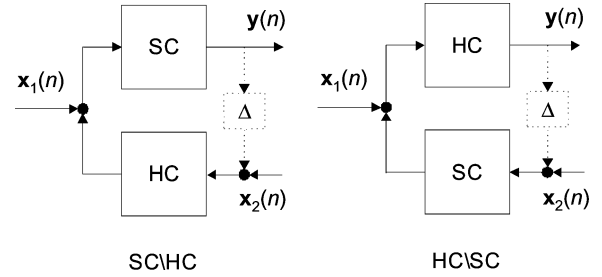


Fig. 3. Soft computing with hard computing feedback and hard computing with soft computing feedback. Dashed line shows a typical connection (Δ is a delay element).

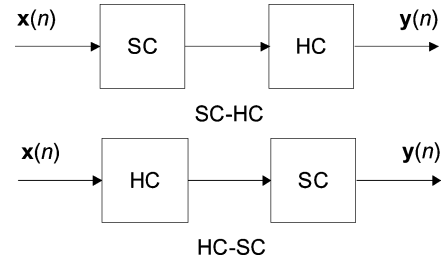


Fig. 4. Cascades of soft computing and hard computing.

of the HC feedback or SC feedback is added to the input of the primary soft computing and hard computing blocks, respectively (Fig. 3). Such reversed parallel connections are used in a variety of control applications; e.g., for improving the transient performance of a closed control loop. As with the forward parallel connection, SC/HC, the feedback parallel connections have also moderate fusion grades. The input-output mappings of SC\HC and HC\SC can be expressed in the following compact form:

$$\mathbf{y}(n) = f_{SC}(\boldsymbol{\theta}_{SC}; [\mathbf{x}_1(n) \oplus f_{HC}(\boldsymbol{\theta}_{HC}; \mathbf{x}_2(n))]) \quad (3)$$

$$\mathbf{y}(n) = f_{HC}(\boldsymbol{\theta}_{HC}; [\mathbf{x}_1(n) \oplus f_{SC}(\boldsymbol{\theta}_{SC}; \mathbf{x}_2(n))]). \quad (4)$$

D. SC-HC and HC-SC

There are two obvious alternatives to form hybrid cascades of soft computing and hard computing algorithms, SC-HC or HC-SC, depending on the sequential order of those functional blocks (Fig. 4). In these popular configurations, the first block is usually a kind of preprocessor and the second one acts as a postprocessor. A typical example of the SC-HC configuration is a dual-stage optimization procedure, where the initial optimization phase is carried out by evolutionary computation (global coarse search), and the intermediate result is then refined by some traditional gradient-based or non-gradient-based algorithm (local fine search) to yield the final optimum. This kind of complementary fusion leads to computationally efficient “global” optimization. On the other hand, a typical HC-SC configuration is an SC-based pattern recognition system, where the feature vectors

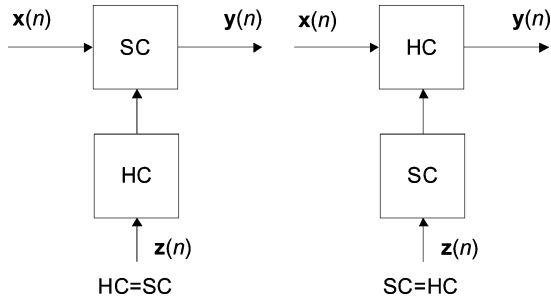


Fig. 5. Hard computing-designed soft computing and soft computing-designed hard computing.

are first constructed by HC-based linear/nonlinear filters or inter-domain transforms. Soft computing techniques are then used for automatic pattern recognition or classification tasks. Also these cascade configurations have moderate fusion grades. Equations (5) and (6) describe the cascade structures SC-HC and HC-SC, respectively

$$y(n) = f_{HC}(\theta_{HC}; f_{SC}(\theta_{SC}; x(n))) \quad (5)$$

$$y(n) = f_{SC}(\theta_{SC}; f_{HC}(\theta_{HC}; x(n))). \quad (6)$$

E. SC = HC and HC = SC

The next two combinations, SC = HC and HC = SC, have high or very high fusion grades. This is because either soft computing is explicitly assisting the design or configuring of a hard computing system, or *vice-versa* (Fig. 5). A usual example of SC = HC is a simple linear controller with SC-based gain scheduling for improved handling of varying dynamical characteristics. Such a hybrid controller could be tuned coarsely by experienced human operators; and the gain scheduling extension further adjusts the control parameters within the pre-defined stability and performance range. In contrast, the conventional back-propagation training of layered neural networks (NNs) is actually an HC = SC type fusion. Another example of the HC = SC type fusion is a dynamic neural network model with the computational structure and possibly some physical parameters taken from a traditional differential or difference equations-based model. The SC = HC and HC = SC mappings can be expressed conveniently by the following formulas:

$$y(n) = f_{SC} \left(\overbrace{\tilde{\theta}_{SC}, f_{HC}(\theta_{HC}; z(n))}^{\text{Parameters}}; x(n) \right) \quad (7)$$

$$y(n) = f_{HC} \left(\overbrace{\tilde{\theta}_{HC}, f_{SC}(\theta_{SC}; z(n))}^{\text{Parameters}}; x(n) \right). \quad (8)$$

Here, the principal soft computing (or hard computing) algorithm has two classes of parameters: internal parameters, $\tilde{\theta}_{\bullet C}$, and externally supplied parameters, $f_{\bullet C}(\theta_{\bullet C}; z(n))$. They both are needed for computing the final output. In some applications, however, the principal SC (HC) algorithm may not have

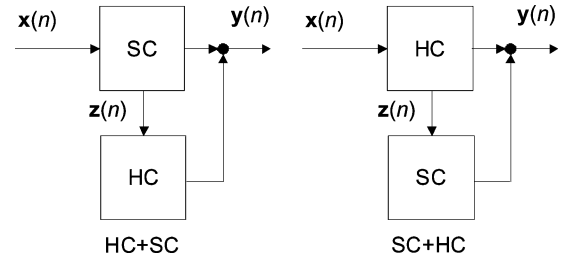


Fig. 6. Hard computing-augmented soft computing and soft computing-augmented hard computing.

any internal parameters, but all the necessary parameters are provided by an external HC (SC) algorithm.

F. HC + SC and SC + HC

The next intimate structures, HC + SC and SC + HC, have very high fusion grades (Fig. 6). In the former case, the HC block is first used for extracting some internal data, $z(n)$, from the SC algorithm and, after some information processing, the resulting HC output is added or combined to the principal SC output. Moreover, in the latter case, the SC block is for processing certain internal data from the HC algorithm, and the final output is the sum or combination of the individual HC and SC outputs. These two categories are not yet widely used in practical applications, but they offer a great potential for developing advanced fusions of soft computing and hard computing methodologies. In Section III, we will identify one promising HC + SC type fusion with emerging application interest. HC + SC and SC + HC structures are formulated as

$$y(n) = f_{SC}(\theta_{SC}; x(n)) \oplus f_{HC}(\theta_{HC}; z(n)) \quad (9)$$

$$y(n) = f_{HC}(\theta_{HC}; x(n)) \oplus f_{SC}(\theta_{SC}; z(n)). \quad (10)$$

G. HC//SC and SC//HC

HC-assisted SC and SC-assisted HC are master-slave type structures that are used moderately/sparingly in real-world applications. In HC//SC, some internal data is first extracted from the SC algorithm (master), processed by the specific HC algorithm (slave), and fed back into the SC algorithm, which completes the primary task. Thus, the hard computing constituent is an integral part of the computational procedure. An example of this kind of fusion is a hybrid genetic algorithm with Lamarckian or Baldwin local search strategies [50], where a local search is following the selection-crossover-mutation chain in the repeated evolution loop; and the role of local search is to refine the chromosome population yielding a new population for fitness evaluation and further genetic manipulations. SC//HC, on the other hand, is an analogous symbiosis between a principal HC algorithm and an assisting SC algorithm. Both of these structures (Fig. 7) have very high fusion grades, and they carry

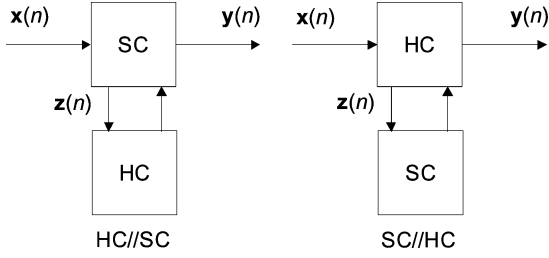


Fig. 7. Hard computing-assisted soft computing and soft computing-assisted hard computing.

out the following input-output mappings:

$$\mathbf{y}(n) = f_{SC} \left(\overbrace{\tilde{\theta}_{SC}; \mathbf{x}(n), f_{HC}(\theta_{HC}; \mathbf{z}(n))}^{\text{Inputs}} \right) \quad (11)$$

$$\mathbf{y}(n) = f_{HC} \left(\overbrace{\tilde{\theta}_{HC}; \mathbf{x}(n), f_{SC}(\theta_{SC}; \mathbf{z}(n))}^{\text{Inputs}} \right). \quad (12)$$

H. Supplementary Categories

In classifying the different kind of fusions of neural networks and fuzzy logic, Hayashi *et al.* proposed 11 structural categories [1]. This is about the same as our 12 core categories for the fusion of soft computing and hard computing. On the other hand, Agarwal identified only the three most obvious fusion structures of neural networks and conventional paradigms; i.e., NN/HC, NN-HC, and HC-NN, in his paper published in 1995 [33]. Another interesting discussion on integration architectures, related to symbolic artificial intelligence (AI), FL, GA, and NN, was presented by Fu in his 1996 conference tutorial [36]. He identified five integration architectures with three coupling strengths. Those hybrid architectures share similarities with our corresponding core categories.

As the traditional field of hard computing is much wider and more heterogeneous than the field of soft computing, we decided to classify, first, only the most universal categories. If the very details of specific hard computing algorithms were taken into consideration, the number of fusion categories would increase considerably. Since the fusion structures of Figs. 1–7 have the minimum number of SC and HC blocks—only one of each kind—they are said to be canonic. It should be pointed out that more complex, large-scale systems are naturally modeled as interconnections of these canonic building blocks.

Most of the fusion categories discussed above were created straightforwardly by following the synergy between the fusion of SC and HC, and the fusion of individual soft computing methods [1], [13]. In addition, a large number of journal and conference papers was carefully analyzed to verify the coverage of a few intermediate category sets. After identifying a relevant structural category, like HC\SC, its reverse category, SC\HC, was also considered for the final category set. This particular reverse category, although not in wide use among

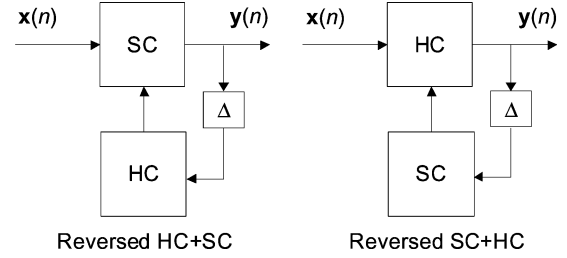


Fig. 8. Reversed HC + SC and reversed SC + HC structures.

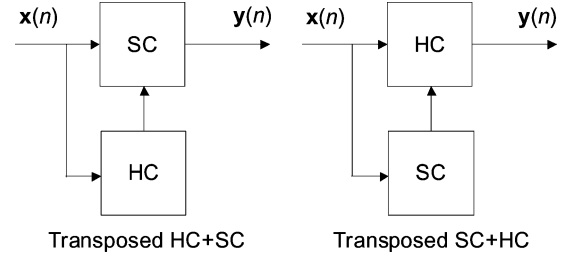


Fig. 9. Transposed HC + SC and transposed SC + HC structures.

practicing engineers, offers apparent potential for developing novel fusions of soft computing and hard computing. Moreover, Fig. 8 illustrates the reverse categories of HC + SC and SC + HC. Equations (13) and (14) give the mappings of the reversed SC + HC and HC + SC structures, respectively

$$\mathbf{y}(n) = f_{SC} \left(\overbrace{\theta_{SC}; \mathbf{x}(n), f_{HC}(\theta_{HC}; \mathbf{y}(n-1))}^{\text{Inputs}} \right) \quad (13)$$

$$\mathbf{y}(n) = f_{HC} \left(\overbrace{\theta_{HC}; \mathbf{x}(n), f_{SC}(\theta_{SC}; \mathbf{y}(n-1))}^{\text{Inputs}} \right). \quad (14)$$

It should be noted here that the fusion structures contain an external feedback and, therefore, the value of $\mathbf{y}(n)$ must be delayed (at least) by one sampling period to make the feedback realizable. These reversed structures have two kinds of inputs: the primary input, $\mathbf{x}(n)$, and the auxiliary input, $f_{\bullet C}(\theta_{\bullet C}; \mathbf{y}(n-1))$.

Structural transposition or flow-graph reversal of the defined core categories leads to two additional structures (Fig. 6 → Fig. 9) that offer innovation potential. Their mathematical mappings are formulated in (15) and (16)

$$\mathbf{y}(n) = f_{SC} \left(\overbrace{\theta_{SC}; \mathbf{x}(n), f_{HC}(\theta_{HC}; \mathbf{x}(n))}^{\text{Inputs}} \right) \quad (15)$$

$$\mathbf{y}(n) = f_{HC} \left(\overbrace{\theta_{HC}; \mathbf{x}(n), f_{SC}(\theta_{SC}; \mathbf{x}(n))}^{\text{Inputs}} \right). \quad (16)$$

For completeness of our category classification, reversed versions of the transposed HC + SC and SC + HC are depicted in

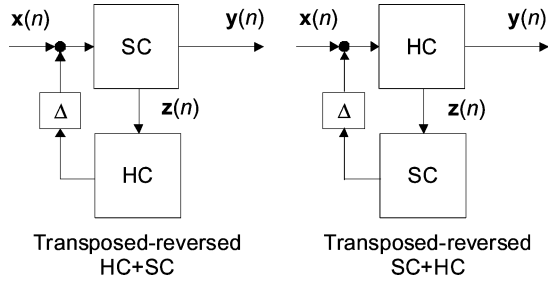


Fig. 10. Transposed-reversed HC + SC and transposed-reversed SC + HC structures.

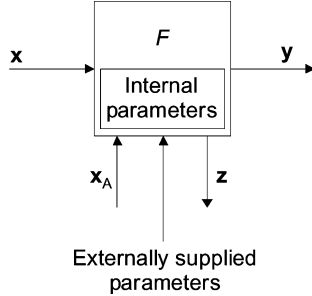


Fig. 11. General SC or HC system with data/signal inputs and outputs, as well as externally supplied system parameters.

Fig. 10. They can be expressed mathematically as

$$\mathbf{y}(n) = f_{SC}(\theta_{SC}; [\mathbf{x}(n) \oplus f_{HC}^{\Delta}(\theta_{HC}; \mathbf{z}(n))]) \quad (17)$$

$$\mathbf{y}(n) = f_{HC}(\theta_{HC}; [\mathbf{x}(n) \oplus f_{SC}^{\Delta}(\theta_{SC}; \mathbf{z}(n))]). \quad (18)$$

In (17) and (18), the necessary loop delay, Δ , is shown as a superscript of the hard computing and soft computing algorithms, respectively.

It should be pointed out that, similar to the core HC + SC and SC + HC structures of Fig. 6, the six supplementary fusion structures of Figs. 8–10 are intimate fusions of soft computing and hard computing and, therefore, have very high fusion grades.

1. General SC and HC Mapping Functions

We will next define mathematically the general mapping functions, $f_{SC}(\cdot; \cdot)$ and $f_{HC}(\cdot; \cdot)$, used in (1)–(18). Our definition follows the operational structure and notations of Fig. 11. It should be noticed, however, that only SC = HC and HC = SC type fusions use the externally supplied parameters that are shown in Fig. 11. The nonlinear or linear mapping function, $F \in \{f_{SC}, f_{HC}\}$, from the primary input vector, $\mathbf{x} \in \mathbb{R}^N$, and the auxiliary input vector (optional), $\mathbf{x}_A \in \mathbb{R}^K$, to the internal output vector (optional), $\mathbf{z} \in \mathbb{R}^L$, and the primary output vector, $\mathbf{y} \in \mathbb{R}^M$, can be expressed as

$$F : \langle \mathbf{x} \in \mathbb{R}^N, \mathbf{x}_A \in \mathbb{R}^K \rangle \rightarrow \langle \mathbf{z} \in \mathbb{R}^L, \mathbf{y} \in \mathbb{R}^M \rangle \quad (19)$$

where $\mathbf{x} = [x_1, x_2, \dots, x_N]$, $\mathbf{x}_A = [x_{A,1}, x_{A,2}, \dots, x_{A,K}]$, $\mathbf{z} = [z_1, z_2, \dots, z_L]$, and $\mathbf{y} = [y_1, y_2, \dots, y_M]$. Here, the auxiliary input vector, \mathbf{x}_A , and the internal output vector, \mathbf{z} , can be considered as partial or full state data of the dynamical mapping function, F . All the input and output spaces may naturally have different dimensions; therefore, this general

TABLE III
CHARACTERISTIC FEATURES OF EC, FL, AND NN

SC methodology	Characteristic features
EC	<ul style="list-style-type: none"> • Search methods inspired by the Neo-Darwinian paradigm • No gradient of the error surface needed • Avoidance of local optima through random perturbations • Computationally intensive
FL	<ul style="list-style-type: none"> • Rule-based approach • Tolerance for imprecision, partial truth, and uncertainty • Capability to approximate any continuous function or system • Possibility to utilize human intuition
NN	<ul style="list-style-type: none"> • Black box approach • Massively parallel distributed structure • Supervised/unsupervised learning from data • Generalization ability • Capability to approximate any continuous function or system

TABLE IV
REPRESENTATIVE CHARACTERISTIC FEATURES OF HC METHODOLOGIES

Specific HC methodology	Characteristic features
PID controller	<ul style="list-style-type: none"> • Simplified linear approximation • Manual tuning possible • Computationally efficient
<i>s</i> -domain models with physical origin	<ul style="list-style-type: none"> • Solid physical basis (differential equations) • Sensitivity to parameter variation • Tedious derivation of high-order models
Fletcher-Powell optimization algorithm	<ul style="list-style-type: none"> • Makes use of derivatives of the error surface • Variable metric method • Efficient unconstrained optimization
General parameter adaptation method	<ul style="list-style-type: none"> • Reduced-rank adaptation • Computationally efficient • Applicable with many computational structures
Stochastic system simulation	<ul style="list-style-type: none"> • Suitable for various kind of complex systems • Demanding phase of model building • Time-consuming brute force technique

definition covers both single-input single-output (SISO) and multiple-input multiple-output (MIMO) type systems.

III. CHARACTERISTIC FEATURES

A. General

The central motivation behind fusing elemental soft computing and hard computing algorithms is the aim to overcome the limitations of individual methodologies. This applies to all fusion categories; the resulting hybrid systems are expected to have such valuable characteristics that would not be either possible or practical to obtain by using a single methodological class only. Table III contains the most important characteristic features of evolutionary computation [15], fuzzy logic [16], and neural networks [17]. All these constituents of SC have functional characteristics that are potentially needed in real-world applications, because the conceptual structure of hard computing is overly precise in relation to the imprecision of the world around us. In addition, there is a growing demand for computationally intelligent and self-organizing systems.

The HC field, on the other hand, is more diverse than soft computing, and it is not feasible to create a comprehensive table that would contain the characteristic features of all specific classes of hard computing methods. Therefore, we built Table IV, which lists only five representative HC methods with their primary characteristic features. The fusion needs and potential of these hard computing methods are discussed as follows.

B. PID Controller

Proportional integral derivative (PID) controllers are undoubtedly the most widely used control algorithms in industrial applications. Thus, practicing engineers favor them in developing future products. However, the requirements of control performance are continuously increasing, and they cannot always be met by using a fixed linear approximation alone. This problem has been solved successfully by complementary EC = HC, FL = HC, and NN = HC type fusions of soft computing and hard computing [18]–[20]. In such hybrid systems, the SC constituent is adjusting the PI(D) control parameters either incrementally or continuously. To ensure the stability of autotuning PI(D) systems, the range and tuning interval of control parameters must be constrained appropriately.

C. Physical Model

A variety of process models for estimation, control, and simulation is developed by modeling the entire physical system using its characteristic differential or difference equations. This requires a thorough understanding of the physical system to be modeled. On the other hand, such core knowledge is commonly available in matured research and development organizations. Nevertheless, differential equation models and their *s*-domain counterparts become cumbersome to deal with if the system to be modeled is considerably nonlinear or time-variant. Otherwise, a high order model, including also critical secondary effects, is needed to obtain the required accuracy. In these cases, the physics-oriented low or moderate order linear models can be complemented by neural networks or fuzzy logic-based nonlinear models [21]. The linear approximation remains the backbone and its behavior is enhanced by NN/HC or FL/HC type fusions, where the soft computing constituent compensates for system nonlinearities, parameter uncertainty, or other inadequacies of the primary hard computing model.

D. Optimization Using Local Information

The gradient-type Fletcher–Powell optimization algorithm [34] and the non-gradient-type Simplex algorithm [44] are widely used unconstrained optimization methods that make use of local information of the error surface. These traditional algorithms work usually well when the error surface is not more than moderately complex. However, with highly peaky error landscapes that are typical for complex nonlinear optimization problems, all the local-based methods have a notable possibility of getting stuck on some (poor) local optima. This undesired possibility can be reduced substantially in advanced EC-HC or HC//EC type dual-phase optimization procedures [12], [22].

E. General Parameter Method

Various NN models are used frequently for time series prediction in embedded real time systems; e.g., in automatic fault detection and system state estimation. In many applications, however, the predicted time series has specific time-varying characteristics that would actually need an adaptive model for maximizing the prediction accuracy. Unfortunately, fully adap-

tive neural networks have high computational burden. Thus, they are rarely used in (embedded) real-time applications. With radial basis function networks (RBFNs), this time variance problem was relieved by introducing an intimate HC+NN type fusion structure [23]. In that hybrid structure, the hard computing constituent is a simple reduced-rank adaptation scheme, the general parameter (GP) method, which tunes the behavior of the RBFN model around a nominal operation point. The desired input-output mapping at the nominal operation point is represented accurately by the fixed RBFN part, and the general parameter extension is able to compensate for moderate time varying characteristics in the time series. Similar intimate fusion schemes could also be applied with other layered neural networks that are used in predictive configurations. For example, the fixed output layer of multilayer perceptron (MLP) networks could be complemented by a least-mean-square (LMS) type adaptive linear combiner [32].

F. Stochastic System Simulation

Stochastic system simulation is a standard technique for analyzing and optimizing complex systems that do not have tractable analytical models. There are commercial simulation packages available with efficient model building functions and versatile visualization capabilities of simulation results. Monte Carlo method based stochastic simulation is considered as a brute force technique for optimizing parameters of complex and large scale systems. From the computation time point of view, it would be desired to keep the number of simulation cycles at a minimum. On the other hand, that would affect directly the statistical reliability of the simulation results. Such a problem could be reduced by combining evolutionary computation with stochastic system simulation. In an HC//EC type fusion, the stochastic system simulator is used to compute the values of the fitness function needed in evolutionary optimization [12], and the EC constituent provides a selective search of the enormous solution space.

G. Summary

Table V summarizes the principal characteristic features of the fusions of soft computing and hard computing that were discussed previously. The constructive fusion approaches have provided improved results over traditional techniques. These improvements are chiefly in such areas as design cost, robustness, and tractability. In addition, the SC-enhanced HC algorithms form a natural intermediate step on the methodological evolution path from pure hard computing to dominating soft computing. Such a multistep path is typically preferred in industrial R&D organizations.

As mentioned earlier in this paper, our categorization is based on a “little model approach.” In this practical approach, each SC or HC constituent is represented by a little model and the categorization is based on the types of *interconnections* of these models. However, fusions of soft computing and hard computing could be extended into “soft computing in a hard computing framework” or “hard computing in a soft computing framework.” In such extended fusion approaches, the resulting system

TABLE V
CHARACTERISTIC FEATURES OF THE CONSIDERED FUSIONS OF SC AND HC

HC constituent	Characteristic features of the fusion
PID controller	<i>General</i> <ul style="list-style-type: none"> • Automatic tuning of control parameters • Manual tuning still possible • Stability predictable <i>EC=HC</i> <ul style="list-style-type: none"> • Global search of control parameters <i>FL=HC</i> <ul style="list-style-type: none"> • Possibility to utilize human intuition <i>NN=HC</i> <ul style="list-style-type: none"> • Learning from data
s-domain models with physical origin	<i>General</i> <ul style="list-style-type: none"> • Solid physical basis • Ability to handle nonlinearities <i>FL/HC</i> <ul style="list-style-type: none"> • Possibility to utilize human intuition <i>NN/HC</i> <ul style="list-style-type: none"> • Learning from data
Fletcher-Powell optimization algorithm	<i>EC-HC</i> <ul style="list-style-type: none"> • Global search • Efficient unconstrained optimization
General parameter adaptation method	<i>HC+NN</i> <ul style="list-style-type: none"> • Partially adaptive neural network model • Computationally efficient
Stochastic system simulation	<i>HC/EC</i> <ul style="list-style-type: none"> • Optimization method for complex systems • Global search • Demanding phase of model building

contains only SC constitutes implemented in an HC framework, or only HC constitutes embedded in an SC framework. Examples of such fusions are genetic algorithms formulated in an object-oriented framework [45], fuzzy rules coded in a Petri net [46], neural networks embedded in inverse-model-based control techniques [47], and various mathematical programming methods combined in a genetic algorithm architecture [48]. Categorization based on this type of fusions is an interesting area to be studied in more detail, and will be a topic of our future research.

IV. CLASSIFICATION OF SPECIFIC HYBRID STRUCTURES

In Table VI, we list a collection of reference papers that contain some fusion structure belonging to one of the core categories of Table I. It should be emphasized, however, that those few references are just illustrative examples of the core fusion categories; they are not meant to give any representative literature survey. To make the reference table more instructive, the soft computing constituent was divided into three subconstituents: evolutionary computation, fuzzy logic, and neural networks. Because the first category, SC&HC, was not included in this consideration, Table VI contains all together 33 possible fusions of SC and HC methodologies. The selected reference papers, as well as the recent overview paper [4], can guide the reader to the realm of practical fusion implementations.

As shown in Table VI, the number of reported applications with respect to HC + SC, SC + HC, HC//SC, and SC//HC is still very small. However, as research and development on various applications with respect to the fusion of soft computing and hard computing evolve further, these intimate structures will likely become a promising area to be explored.

One noticeable trend in the fusion of soft computing and hard computing is the simultaneous partnership of HC and *multiple*

TABLE VI
EXAMPLES OF PUBLISHED FUSIONS OF SC AND HC

Category	SC constituent	HC constituent	Reference
SC/HC	EC	Mixed-integer linear programming	[37]
	FL	P type controller	[35]
	NN	Linear feedback controller	[28]
SC&HC	EC	Linear combination of visual perception	[38]
	FL	First-order linear filter	[39]
	NN	Noise filter	[40]
HC&SC	EC	Spacecraft navigation and control	[41]
	FL	Feedback linearization controller	[27]
	NN	Linearizing control	[42]
SC-HC	EC	Powell's unconstrained optimization method	[22]
	FL	Linear quadratic Gaussian type controller	[25]
	NN	Narrow-band lowpass filters (integrators)	[30]
HC-SC	EC	Linear programming for initial population	[49]
	FL	Extended Kalman filter	[25]
	NN	Feature extraction algorithm	[5]
HC=SC	EC	Constraint satisfaction techniques	[43]
	FL	Gradient-descent method	[35]
	NN	Newton-Gauss optimization algorithm	[31]
SC=HC	EC	Two PI controllers with cross-coupling	[18]
	FL	Dynamic programming algorithm	[26]
	NN	Optimizing long-range predictive controller	[29]
HC+SC	EC	—	—
	FL	—	—
	NN	General parameter adaptation algorithm	[23]
SC+HC	EC	—	—
	FL	—	—
	NN	—	—
HC//SC	EC	Stochastic system simulator	[12]
	FL	—	—
	NN	—	—
SC//HC	EC	—	—
	FL	—	—
	NN	—	—

SC methodologies [2]. Here, the fused soft computing methodologies are targeted to provide computational intelligence or high machine IQ (MIQ) [51] for a variety of advanced systems, products, and services [52].

V. CONCLUSION AND DISCUSSION

We introduced 12 structural core categories for the fusion of soft computing and hard computing. By comparing our core categories to those related to the fusion of individual SC methodologies, and after reviewing a large amount of application papers, it can be concluded that the fusion of soft computing and hard computing is already a well-defined alternative for developing computationally intelligent hybrid systems that largely overcome the shortages and limitations of individual SC and HC methodologies. In addition to those core categories, we also identified a set of supplementary categories that offer considerable innovation potential for future applications.

Although the hard computing and soft computing research communities are quite separated, it is good to observe that many SC-oriented researchers and engineers are openly favoring the various fusion opportunities. The HC community generally seems to be more reluctant in applying the complementing SC techniques. This situation may change progressively following positive experiences, and the associated evolving emphasis in electrical and computer engineering curricula. Currently, there is usually an unfortunate division between hard computing and soft computing-oriented courses related to major application

topics such as industrial electronics, electric power systems, and telecommunications. This separation is clearly hindering the industrial acceptance of SC, because those separated courses do not typically consider the methodological fusion approach at all, but mostly neglect the other computing methodology.

ACKNOWLEDGMENT

The first author wishes to thank T. Bose (Utah State University) for providing the visiting scientist position that made it possible to complete this manuscript in the summer of 2004. In addition, the authors are grateful to Reviewer #2 for his critical comments and helpful suggestions aimed at improving the manuscript.

REFERENCES

- [1] I. Hayashi, M. Umamo, T. Maeda, A. Bastian, and L. C. Jain, "Acquisition of fuzzy knowledge by NN and GA—A survey of the fusion and union methods proposed in Japan," in *Proc. 2nd Int. Conf. Knowledge-Based Intelligent Electronic Systems*, Adelaide, Australia, Apr. 1998, pp. 69–78.
- [2] H. Takagi, "R&D in intelligent technologies: Fusion of NN, FS, GA, chaos, and human," in *IEEE Int. Conf. Systems, Man, and Cybernetics*, Orlando, FL, Oct. 11, 1997, half-day tutorial/workshop.
- [3] S. J. Ovaska and H. F. VanLandingham, "Guest editorial: Special issue on fusion of soft computing and hard computing in industrial applications," *IEEE Trans. Syst., Man, Cybern. C: Appl. Rev.*, vol. 32, no. 2, pp. 69–71, May 2002.
- [4] S. J. Ovaska, H. F. VanLandingham, and A. Kamiya, "Fusion of soft computing and hard computing in industrial applications: An overview," *IEEE Trans. Syst., Man, Cybern. C: Appl. Rev.*, vol. 32, no. 2, pp. 72–79, May 2002.
- [5] B. Sick, "Fusion of hard and soft computing techniques in indirect, online tool wear monitoring," *IEEE Trans. Syst., Man, Cybern. C: Appl. Rev.*, vol. 32, no. 2, pp. 80–91, May 2002.
- [6] C. Shi and A. D. Cheok, "Performance comparison of fused soft control/hard observer type controller with hard control/hard observer type controller for switched reluctance motors," *IEEE Trans. Syst., Man, Cybern. C: Appl. Rev.*, vol. 32, no. 2, pp. 99–112, May 2002.
- [7] G. D. Buckner, "Intelligent bounds on modeling uncertainties: Applications to sliding mode control," *IEEE Trans. Syst., Man, Cybern. C: Appl. Rev.*, vol. 32, no. 2, pp. 113–124, May 2002.
- [8] R. Sterritt and D. W. Bustard, "Fusing hard and soft computing for fault management in telecommunications systems," *IEEE Trans. Syst., Man, Cybern. C: Appl. Rev.*, vol. 32, no. 2, pp. 92–98, May 2002.
- [9] M. Oosterom, R. Babuska, and H. B. Verbruggen, "Soft computing applications in aircraft sensor management and flight control law reconfiguration," *IEEE Trans. Syst., Man, Cybern. C: Appl. Rev.*, vol. 32, no. 2, pp. 125–139, May 2002.
- [10] J. J. Murray, C. J. Cox, G. G. Lendaris, and R. Saeks, "Adaptive dynamic programming," *IEEE Trans. Syst., Man, Cybern. C: Appl. Rev.*, vol. 32, no. 2, pp. 140–153, May 2002.
- [11] S.-B. Cho, "Incorporating soft computing techniques into a probabilistic intrusion detection system," *IEEE Trans. Syst., Man, Cybern. C: Appl. Rev.*, vol. 32, no. 2, pp. 154–160, May 2002.
- [12] R. H. Kewley and M. J. Embrechts, "Computational military tactical planning system," *IEEE Trans. Syst., Man, Cybern. C: Appl. Rev.*, vol. 32, no. 2, pp. 161–171, May 2002.
- [13] I. Hayashi and M. Umamo, "Perspectives and trends of fuzzy-neural networks," *J. Jpn. Soc. Fuzzy Theory Syst.*, vol. 5, no. 2, pp. 178–190, 1993. (in Japanese).
- [14] T. Furuhashi, "Fusion of fuzzy/neuro/evolutionary computing for knowledge acquisition," *Proc. IEEE*, vol. 89, no. 9, pp. 1266–1274, Sep. 2001.
- [15] D. B. Fogel, *Evolutionary Computation: Toward a New Philosophy of Machine Intelligence*. Piscataway, NJ: IEEE Press, 1999.
- [16] B. Kosko, *Fuzzy Engineering*. Upper Saddle River, NJ: Prentice-Hall, 1996.
- [17] S. Haykin, *Neural Networks: A Comprehensive Foundation*. Upper Saddle River, NJ: Prentice-Hall, 1999.
- [18] Y. Zhao, R. M. Edwards, and K. Y. Lee, "Hybrid feedforward and feedback controller design for nuclear steam generators over wide range operation using genetic algorithm," *IEEE Trans. Energy Convers.*, vol. 12, no. 1, pp. 100–105, Mar. 1997.
- [19] K. C. Jeong, S. H. Kwon, D. H. Lee, M. W. Lee, and J. Y. Choi, "A fuzzy logic-based gain tuner for PID controllers," in *Proc. FUZZ-IEEE*, Anchorage, AK, May 1998, pp. 551–554.
- [20] S. Omatu, T. Iwasa, and M. Yoshioka, "Skill-based PID control by using neural networks," in *Proc. IEEE Int. Conf. Systems, Man, and Cybernetics*, San Diego, CA, Oct. 1998, pp. 1972–1977.
- [21] Y. Tipsuwan and M.-Y. Chow, "Analysis of training neural compensation model for system dynamics modeling," in *Proc. Int. Joint Conf. Neural Networks*, Washington, DC, Jul. 2001, pp. 1250–1255.
- [22] T. S. Kim and G. S. May, "Optimization of via formation in photosensitive dielectric layers using neural networks and genetic algorithms," *IEEE Trans. Electron. Packag. Manuf.*, vol. 22, no. 2, pp. 128–136, Apr. 1999.
- [23] D. F. Akhmetov, Y. Dote, and S. J. Ovaska, "Fuzzy neural network with general parameter adaptation for modeling of nonlinear time-series," *IEEE Trans. Neural Netw.*, vol. 12, no. 1, pp. 148–152, Jan. 2001.
- [24] D. E. Goldberg, "A meditation on the application of soft computing and its future," in *Soft Computing and Industry: Recent Applications*, R. Roy, M. Köoppen, S. Ovaska, T. Furuhashi, and F. Hoffmann, Eds. London, U.K.: Springer-Verlag, 2002, pp. XV–XVIII.
- [25] A. Ben-Abdenour and K. Y. Lee, "An autonomous control system for boiler turbine units," *IEEE Trans. Energy Convers.*, vol. 11, no. 2, pp. 401–406, Jun. 1996.
- [26] F.-C. Lu and Y.-Y. Hsu, "Fuzzy dynamic programming approach to reactive power/voltage control in a distribution substation," *IEEE Trans. Power Syst.*, vol. 12, no. 2, pp. 681–688, May 1997.
- [27] L.-C. Lin and T.-B. Gau, "Feedback linearization and fuzzy control for conical magnetic bearings," *IEEE Trans. Contr. Syst. Technol.*, vol. 5, no. 4, pp. 417–426, Jul. 1997.
- [28] Y. Q. Chen, K. L. Moore, and V. Bahl, "Learning feedforward control using a dilated B-spline network: Frequency domain analysis and design," *IEEE Trans. Neural Netw.*, vol. 15, no. 2, pp. 355–366, Mar. 2004.
- [29] G. Prasad, E. Swidenbank, and B. W. Hogg, "A neural net model-based multivariable long-range predictive control strategy applied in thermal power plant control," *IEEE Trans. Energy Convers.*, vol. 13, no. 2, pp. 176–182, Jun. 1998.
- [30] M. G. Simões and B. K. Bose, "Neural network based estimation of feedback signals for a vector controlled induction motor drive," *IEEE Trans. Ind. Appl.*, vol. 31, no. 3, pp. 620–629, May/Jun. 1995.
- [31] D. Gorinevsky, A. Kapitanovsky, and A. Goldenberg, "Neural network architecture for trajectory generation and control of automated car parking," *IEEE Trans. Contr. Syst. Technol.*, vol. 4, no. 1, pp. 50–56, Jan. 1996.
- [32] B. Widrow and S. D. Stearns, *Adaptive Signal Processing*. Englewood Cliffs, NJ: Prentice-Hall, 1985.
- [33] M. Agarwal, "Combining neural and conventional paradigms for modeling, prediction, and control," in *Proc. 4th IEEE Conf. Control Applications*, Albany, NY, Sep. 1995, pp. 566–571.
- [34] R. Fletcher and M. J. D. Powell, "A rapidly convergent descent method for minimization," *Comput. J.*, vol. 6, no. 2, pp. 163–168, 1963.
- [35] J. Abonyi, L. Nagy, and F. Szeifert, "Adaptive Sugeno fuzzy control: A case study," in *Advances in Soft Computing: Engineering Design and Manufacturing*, R. Roy, T. Furuhashi, and P. K. Chawdhry, Eds. London, U.K.: Springer-Verlag, 1999, pp. 135–146.
- [36] L. M. Fu, "Tutorial 5: Knowledge and neural heuristics," in *IEEE Int. Conf. Neural Networks*, Washington, DC, Jun. 2, 1996, half-day tutorial/workshop.
- [37] T. Nishi, T. Inoue, and Y. Hattori, "Development of a decentralized supply chain optimization system," in *Proc. Int. Symp. Design, Operation and Control of Next Generation Chemical Plants*, Kyoto, Japan, Dec. 2000, pp. 141–151.
- [38] S. Schaal and D. Sternad, "Learning of passive motor control strategies with genetic algorithms," in *Lectures in Complex Systems*, L. Nadel and D. Stein, Eds. Boston, MA: Addison-Wesley, 1992, pp. 631–643.
- [39] J. Abonyi, R. Babuska, M. A. Botto, F. Szeifert, and L. Nagy, "Identification and control of nonlinear systems using fuzzy Hammerstein models," *Ind. Eng. Chemistry Res.*, vol. 39, pp. 4302–4314, 2000.
- [40] Q. Zhang and S. J. Stanley, "Real-time water treatment process control with artificial neural networks," *J. Environ. Eng.*, vol. 125, pp. 153–160, Feb. 1999.
- [41] S. McClintock, T. Lunney, and A. Hashim, "A genetic algorithm environment for star pattern recognition," *J. Intell. Fuzzy Syst.*, vol. 6, pp. 3–16, 1998.

- [42] K. Fregene and D. Kennedy, "Control of a high-order power system by neural adaptive feedback linearization," in *Proc. IEEE Int. Symp. Intelligent Control/Intelligent Systems and Semiotics*, Cambridge, MA, Sep. 1999, pp. 34–39.
- [43] N. Barnier and P. Brisset, "Optimization by hybridization of a genetic algorithm with constraint satisfaction techniques," in *Proc. IEEE Int. Conf. Evolutionary Computation*, Anchorage, AK, May 1998, pp. 645–649.
- [44] L. Yang, J. Yen, A. Rajesh, and K. D. Kihm, "A supervisory architecture and hybrid GA for the identifications of complex systems," in *Proc. Cong. Evolutionary Computation*, Washington, DC, Jul. 1999, pp. 862–869.
- [45] C. S. Krishnamoorthy, P. P. Venkatesh, and R. Sudarshan, "Object-oriented framework for genetic algorithms with application to space truss optimization," *J. Comput. Civil Eng.*, vol. 16, pp. 66–75, Jan. 2002.
- [46] W. Chiang, K. F. R. Liu, and J. Lee, "Bridge damage assessment through fuzzy Petri net based expert system," *J. Comput. Civil Eng.*, vol. 14, pp. 141–149, Apr. 2000.
- [47] M. A. Hussain, "Review of the applications of neural networks in chemical process control—Simulation and online implementation," *Artif. Intell. in Eng.*, vol. 13, pp. 55–68, 1999.
- [48] J. Rachlin, R. Goodwin, S. Murthy, R. Akkiraju, F. Wu, S. Kumaran, and R. Das, "A-teams: An agent architecture for optimization and decision support," in *Lecture Notes in Artificial Intelligence (Vol. 1555): Intelligent Agents V*, J. Mueller, M. Singh, and A. Rao, Eds. Heidelberg, Germany: Springer-Verlag, 1999, pp. 261–276.
- [49] G. R. Raidl, "An improved genetic algorithm for the multiconstrained 0-1 knapsack problem," in *Proc. IEEE Int. Conf. Evolutionary Computation*, Anchorage, AK, May 1998, pp. 207–211.
- [50] D. Newth, M. Kirley, and D. G. Green, "An investigation of the use of local search in NP-hard problems," in *Proc. 26th Ann. Conf. IEEE Industrial Electronics Soc.*, vol. 4, Nagoya, Japan, Oct. 2000, pp. 2710–2715.
- [51] H.-J. Park, B. K. Kim, and K. Y. Lim, "Measuring the machine intelligence quotient (MIQ) of human-machine cooperative systems," *IEEE Trans. Syst., Man, Cybern. A: Syst. Humans*, vol. 31, pp. 89–96, Mar. 2001.
- [52] P. P. Bonissone, Y.-T. Chen, K. Goebel, and P. S. Khedkar, "Hybrid soft computing systems: Industrial and commercial applications," *Proc. IEEE*, vol. 87, pp. 1641–1667, Sep. 1999.



Seppo J. Ovaska (M'85–SM'91) received the M.Sc. degree in electrical engineering from Tampere University of Technology, Tampere, Finland, the Lic.Sc. degree in computer science and engineering from Helsinki University of Technology, Espoo, Finland, and the D.Sc. degree in electrical engineering from Tampere University of Technology in 1980, 1987, and 1989, respectively.

He is currently a Professor in the Department of Electrical and Communications Engineering, Helsinki University of Technology. Before joining Helsinki University of Technology in 1996, he was an Associate/Full Professor in the Department of Information Technology, Lappeenranta University of Technology, Lappeenranta, Finland. From 1980 to 1992, he held engineering, research, and R&D management positions with Kone Elevators and Nokia Research Center, both in Finland and in the U.S. In the summer of 1999, he was a Visiting Scientist at Muroran Institute of Technology, Muroran, Japan; in the summers of 2000 and 2001, at Virginia Polytechnic Institute and State University, Blacksburg, in the summers of 2002–2004, at Utah State University, Logan; and in the summer of 2005, at the University of Passau, Passau, Germany. His research interests are in soft computing, fault diagnosis, signal processing, and control. He has authored or coauthored over 200 papers in peer-reviewed journals and international conferences. He edited *Computationally Intelligent Hybrid Systems: The Fusion of Soft Computing and Hard Computing* (Hoboken, NJ: Wiley—IEEE Press, 2004), and holds nine patents in the area of systems and control.

Dr. Ovaska is an Associate Editor of the IEEE TRANSACTIONS ON SYSTEMS, MAN, AND CYBERNETICS—PART C: APPLICATIONS AND REVIEWS and the IEEE TRANSACTIONS ON NEURAL NETWORKS. He was the Founding General Chair of the 1999 IEEE Midnight-Sun Workshop on Soft Computing Methods in Industrial Applications. In addition, he was the General Chair of the 5th Online World Conference on Soft Computing in Industrial Applications (2000). He is a recipient of two Outstanding Contribution Awards of the IEEE Systems, Man, and Cybernetics Society.



Akimoto Kamiya (M'92–SM'01) received the B.S. degree in electrical engineering from Nagasaki University, Nagasaki, Japan, in 1975 and the Ph.D. degree in engineering from Tokyo Institute of Technology, Yokohama, Japan, in 1997.

He joined Toshiba Corporation, Tokyo, in 1975, and was a Senior Specialist in the Energy Systems Engineering Group, Toshiba Corporation Power and Industrial Systems Research and Development Center. Since 2002, he has been a Professor in the Department of Information Engineering, Kushiro National College of Technology, Kushiro, Japan. His research and development interests include evolutionary computation, neural networks, and reinforcement learning.

Dr. Kamiya was the Technical Program Chair of the 2005 IEEE Mid-Summer Workshop on Soft Computing in Industrial Applications, and the Program Co-Chair of the IEEE International Workshop on Soft Computing as Transdisciplinary Science and Technology (2005). He is a member of JSAI, IEE, and SICE.



YangQuan Chen (S'95–SM'98) received the B.S. degree in industrial automation from the University of Science and Technology of Beijing, China, in 1985, the M.S. degree in automatic control from Beijing Institute of Technology in 1989, and the Ph.D. degree in control and instrumentation, Nanyang Technological University, Singapore, in 1998.

He is currently an Assistant Professor of Electrical and Computer Engineering at Utah State University, Logan and the Acting Director of the Center for Self-Organizing and Intelligent Systems (CSOIS). His current research interests include robust iterative learning and repetitive control, identification and control of distributed parameter systems with networked movable actuators and sensors, autonomous ground mobile robots, fractional order dynamic systems and control, computational intelligence, intelligent mechatronic systems, and visual servoing/tracking. He holds 12 granted and 2 pending U.S. patents in various aspects of hard-disk drive servomechanics. He published more than 150 papers in refereed journals and conferences and more than 50 industrial technical reports. He coauthored a research monograph *Iterative Learning Control: Convergence, Robustness and Applications* (with C. Wen, Lecture Notes Series in Control and Information Science, New York: Springer-Verlag, 1999) and two textbooks: *System Simulation: Techniques and Applications Based on MATLAB/Simulink* (with D. Xue, Tsinghua Univ. Press, Beijing, China, 2002), and *Solving Advanced Applied Mathematical Problems Using Matlab* (with D. Xue, Tsinghua Univ. Press, 2004).

Dr. Chen is now an Associate Editor on the Conference Editorial Board of the Control Systems Society of IEEE, and an Associate Editor on the ISA Editorial Board for the American Control Conference.