Journal of Science, Engineering and Technology, Vol. 10 (1), March 2023: pages 98-105

PRAISE FOR SOFT COMPUTING

Duke, Stephen Orok

Department of Computer Science, University of Cross River State, Calabar, Nigeria.

Email addresses: orokduke2003@yahoo.com, orokduke2003@unicross.edu.ng

ABSTRACT

In this paper, we attempt to praise soft computing for its enormous contributions to changing the landscape of computing. Soft computing, is a paradigm shift in problem-solving from the classical or traditional hard computing technique. While soft computing utilizes multi-valued logic stochastically to provide solutions to real-world problems with less computational time, hard computing handles deterministic and linear problems utilizing unnoisy data and two-valued logic. Research evidences abound to show the successful applications of soft computing in resolving scientific, industrial, engineering, domestic, commercial, agriculture, medicine, etc. issues.

Keywords: Hard computing, soft computing, Artificial Neural Networks, Fuzzy Logic, Expert Systems, Genetic Algorithms.

1.0 INTRODUCTION

Problem-solving has two techniques: Hard Computing (HC) and Soft Computing (SC). Modern control systems are associated with complex problems that the classical HC and procedural mathematical models cannot describe accurately. HC, which was coined by Lotfi Zadeh, is the conventional approach used in computing and requires an accurate analytical model. HC depends on binary logic and predefined instructions, like numerical analysis and brisk software, and uses two-valued logic as well as crisp systems. It is extremely difficult to control complex plants using the HC methodology because complex plants are associated with imprecision, vagueness, ambiguity, and uncertainty.

SC methodologies can best solve these complex problems precisely. SC deals with imprecision, vagueness, partial truth, approximation, ambiguity, and uncertainty when solving complex problems. The principle guiding SC is to utilize tolerance towards partial truth, uncertainty, and imprecision to actualize low solution costs, robustness, better rapport, and tractability amidst reality. In fact, the role model for SC is a human mind. SC has important features that make it an intelligent control technique, including optimization, non-linear programming, and decision-making support techniques. These features have made SC very popular and have attracted a lot of research interest from people with diverse backgrounds (Jang et al., 1997).

SC is a new problem-solving horizon, the hottest topic in problem-solving, and it is changing the landscape of computing. SC is NOT a one- whole methodological approach for solving problems. It comprises of different related techniques, which, fascinatingly, are not competing with one another but are complementary to each other and can be combined to form a hybrid methodology to provide solutions to given complex problems (Buckley and Hayashi, 1994). The principal components or constituents of SC are Fuzzy Logic (FL), Genetic Algorithms (GAs), Neural Artificial Networks (ANNs), Probabilistic Reasoning (PR), and Neurocomputing.

Using the HC methodology to solve and control the ever-increasing complexity of modern machinery has become very difficult. HC cannot easily control and stabilize the

many non-linear and time-variant plants with bulky time delays. Today, SC methodology has proven to be the most efficient method of controlling complex plants and solving complex problems through the exploitation of uncertainty and imprecision in decision-making manipulations or processes.

2.0 HC VERSUS SC METHODOLOGIES

Generally, the main or dominant problem-solving techniques are HC and SC. Figure 1 shows the activities of the HC and SC problem-solving techniques.

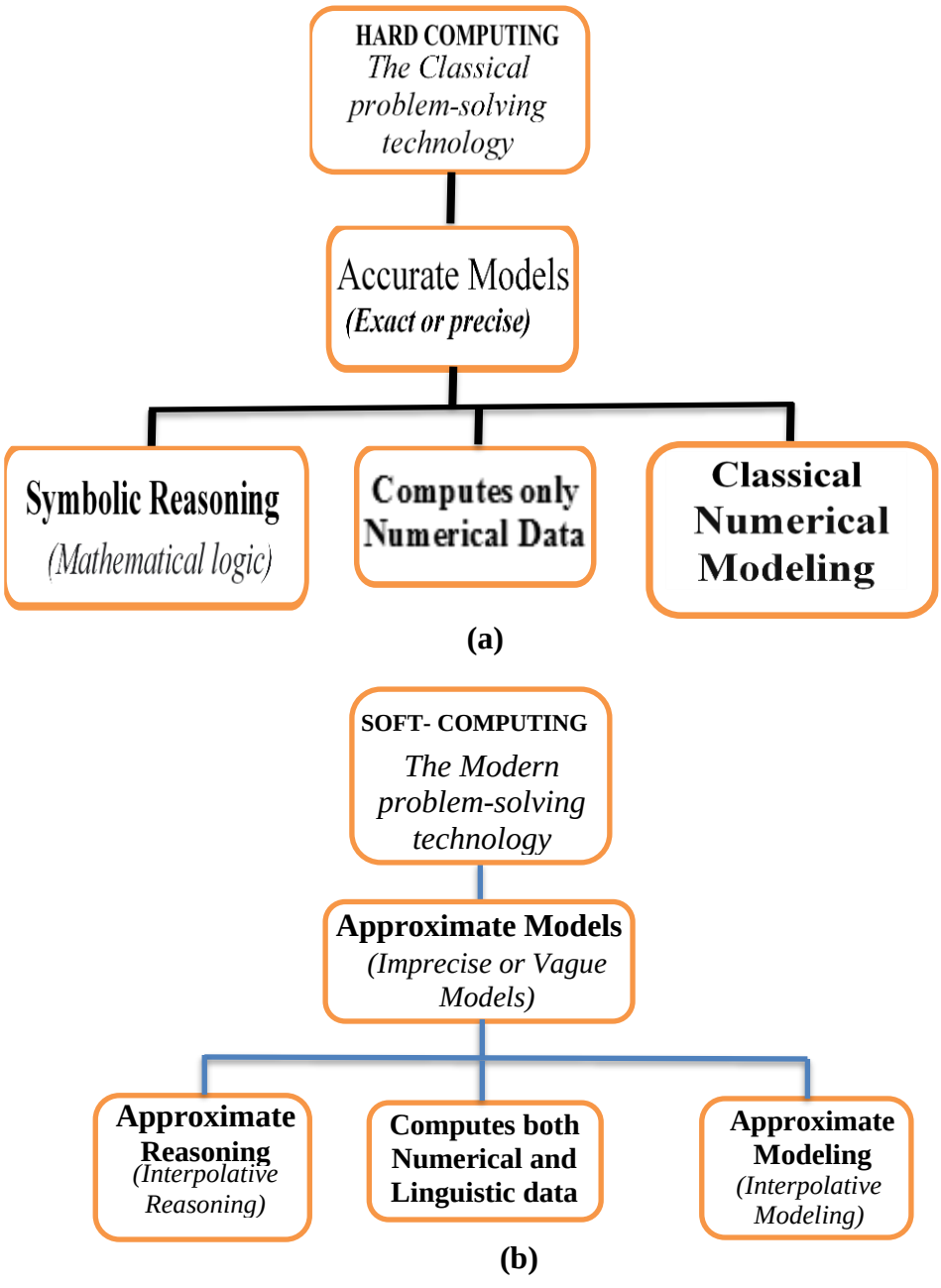

Figure 1: Problem-Solving Methodologies: HC versus SC

Figure 1**(a)** shows the classical HC problem-solving method. This method adopts accurate (exact or precise) models in providing solutions to both simple and complex problems via symbolic reasoning or mathematical logic and numerical modeling. This technique can only compute numerical data perfectly. On the other hand, figure 1**(b)**

indicates SC problem-solving method or approach. This approach adopts approximate models in solving complex problems through approximate reasoning or interpolating reasoning and approximate modeling. It also has the capability of computing with both numerical and linguistic data. Table 1 shows further distinctions between these problem-solving methodologies.

Table 1: SC versus HC

Soft computing	Hard computing
i. Tolerates imprecision, uncertainty, and approximations (adopts the human brain as role model to solve problems). Can provide different answers or results different times when the problems are solved different times.	Intolerance of imprecision, uncertainty, and approximations (needs precise stated analytical models to solve problems).
ii. Utilizes multi-valued logic (fuzzy logic) to solve problems.	Utilizes two-valued logic to solve problems
iii. Consumes much less computation time	Consumes much more computation time
iv. Stochastically in nature (Best candidate for solving real-world problems).	Deterministic in nature and best candidate for solving mathematical problems. Hard computing cannot solve real-world problems with accuracy.
v. Evolves its own program (learn itself) to compute.	Does not evolve its own program. Needs program to be written to compute a particular problem.
vi. Is regarded as computation intelligence	Is regarded as conventional intelligence
vii. Programs are not connected directly or linked to hardware such as database and spreadsheet	Programs are linked or connected to hardware.
iii. Depends on fuzzy logic, neural networks, probabilistic reasoning.	Depends on binary logic, crisp inputs, crisp systems, numerical analysis.
ix. Deals with noisy data	cannot deal with noisy data, but exact data
x. Provides approximate solutions or answers	Provides exact solutions or answers
xi. Robustness (can cope with errors during the time of execution as well as with erroneous inputs).	Lack of robustness (cannot cope with errors during execution time as well as erroneous inputs). Requires accuracy.

3.0 CONSTITUENTS OF SOFT COMPUTING

Complex control problems have been solved by means of fuzzy control techniques over the past few decades (Driankov et al., 1993). Furthermore, Fuzzy Logic paradigms or concepts are now in use to solve many instrumental-based problems (Russo, 1996). A lot of people today are using the newer notion or concept of SC, or neural networks, to provide solutions to complex automatic control problems as well as servo-based problems (Gao and Ovaska, 1999). Beside solving automatic control problems, SC also finds useful applications in diverse areas, namely: fields of signal processing (Cichocki and Unbehauen, 1998), pattern recognition (Pedrycz, 1990), heavy current systems (Dote and Hoft, 1998), intelligent speech recognition (Komori, 1992), design and manufacturing (Roy et al., 1998), communications (Yuhas, 1994), etc.

Some of the constituents of the SC paradigms applied in solving complex problems are presented in the subsections below, including: Genetic Algorithms (GAs), Artificial Neural Networks (ANNs), Fuzzy Logic (FL), and Expert Systems (ESs).

3.1 Expert Systems (ESs)

An Expert System (ES) is a computer-based system as well as an Artificial Intelligence (AI) that mimics the decision-making abilities of human experts. Human experts design ESs to solve complex problems by reasoning through bodies of knowledge. The bodies of knowledge that make up an ES are represented mainly by the IF-THEN rule statements rather than via the conventional procedural codes.

An ES has the capacity or ability to change its own decisions and make new ones based on external factors. While some expert systems are designed to take the place of humans in some applications, others are designed to assist or aid humans. ESs have found useful applications in such areas as engineering design, legal matters, online medical systems for diagnosing diseases in humans and livestock, robotics, and financial loan and credit decisions.

Components of an ES

The main components of every ES are the Knowledge Base (KB), Working Memory (WM), Inference Engine (IE), and User Interface (UI).

i. **KB:** The KB is the most important component of every expert system. KB stores the intelligence of the system in the form of an IF-THEN-ELSE construct; that is, it is the system's intelligence repository (the special heuristics or rules that direct the use of the knowledge, facts, etc.). It contains the knowledge necessary for understanding, formulating, and problem-solving. Expert systems in general can acquire new knowledge through their sensors or through training and expand their knowledge based on how they should easily respond to new problems.

ii. **WM:** The WM is the Blackbox of an expert system. It describes the current problems and records the intermediate results.

iii. **IE:** This is the deduction system or part of the system that infers the results from the user input and KB. IE is the brain of the expert system, the rule interpreter (control structure). It gives the methodology for reasoning.

iv. **UI:** The UI interfaces the users with the system through Natural Language Processing (NLP), menus, and graphics. The UI is found between the KB and the user. It also acts as a language processor for friendly, problem-oriented communication. The interface is where the decisions are made by following the conditions and requirements before coming to an outcome and presenting solutions to the users. The system user interface is normally in the form of the natural language we use daily in our everyday lives. Expert systems are best developed using symbolic programming languages such as Clips, Prolog, Java, OPS-5, Python, Lisp, and so on. They are also developed using algorithmic-based or problem-oriented programming languages such as FORTRAN, PASCAL, BASIC, and C. These are the traditional high-level programming languages known as procedural languages.

3.2 Genetic Algorithms (GAs)

GAs (Whitley, 1994; Holland, 1992), as a part of artificial intelligence and fuzzy computing, are mainly used to solve problems based on optimization encountered in real-life applications. The basic idea behind genetic algorithms is to imitate or mimic the natural selection in nature to find a good selection for an application. Basically, GAs is a machine learning model that is

inspired by the action of evolution in nature. We can utilize it to find solutions to complex search problems encountered in engineering applications. For example, genetic algorithms can search through a lot of designs and components to find the best combination that will lead to or result in an overall better and cheaper design.

Areas of application of GAs

Primarily, people use GAs to solve optimization problems of diverse types to maximize or minimize a specified objective function value within a specified set of constraints. They are also frequently made use of in several other fields, such as biomedical engineering, control engineering, design, climatology, code-breaking, games theory, electronic design, automated manufacturing, the traveling salesman problem and its applications, Multimodal Optimization, DNA Analysis, Parametric Design of Aircraft, Robot Trajectory Generation, Machine Learning, Scheduling applications, Vehicle routing problems, Image Processing, Parallelization, Neural Networks, Economics, etc.

The Processes in GAs

GAs has the following basic processes:

- Initialization Process: This is where we create an initial population randomly.
- Evaluation Process: This is where each member of the population is evaluated, and the fitness of the individuals is assessed based on how well they fit the desired requirements.
- Selection Process: This is where only the ones that fit the desired requirements are reselected.
- Crossover Process: This is where new individuals are created by combining the best aspects of the existing individuals.

3.3 Fuzzy Logic (FL)

Figure 2 demonstrates the basic architecture and configuration of a typical FL system. The concept of FL represents the imprecise nature of human knowledge. There are four interfaces in the FL systems: the fuzzification interface, the inference engine interface, the fuzzy rule base interface, and the defuzzification interface. and the inputs to the FL system are always crisp. FL systems can handle both numerical and linguistic input data perfectly. The first operation in the FL systems is to transform the crisp input value into a fuzzy linguistic value, called fuzzy input in the fuzzification interface. Next, the inference engine captures the fuzzy input and fuzzy rule in the form of IF...THEN statements using linguistic variables to generate fuzzy outputs. The last operation in the FL systems transforms the fuzzy output into crisp output. We call this process defuzzification. The defuzzification is carried out in the defuzzification interface.

One of the major advantages of FL perhaps, is that it provides practical ways to designing nonlinear control systems, which are difficult to design and stabilize sing traditional methods.

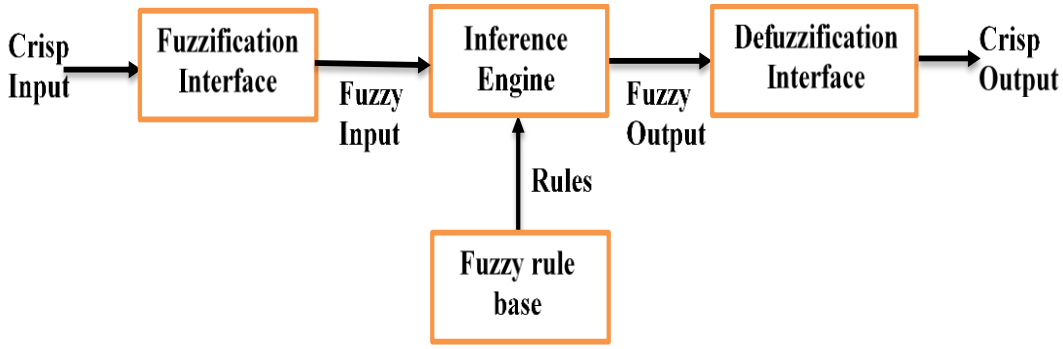

Figure 2: A Typical Architecture of Fuzzy Logic

3.4 Artificial Neural Networks (ANNs)

An ANN is an information processing system or a computing system that is made up of a lot of simple but highly interconnected processing elements or components, called neurons. An ANN manipulates or processes information through its dynamic state response to externally clarified or defined inputs.

ANNs, or Neural Computing (NC), or Artificial Neural Network Model (ANNM), as it is today tends to be called in some fields like artificial intelligence, statistics, cognitive psychology, etc., as an emerging model, is one of the swiftly and proliferating growing fields of research study that has aroused, inspired, and attracted researchers from a wide array or spectrum of disciplines in the sciences, arts, and engineering, like electronic engineering, control engineering, and software engineering.

ANNs are unique information processing systems that are inspired by or influenced essentially by the way the biological nervous system and the brain work. ANNs consist of interconnected elements of large networks of artificial neurons inspired by natural human neurons. In the ANNs, every neuron is made to perform a little operation, and the weighted sum of these operations is the overall operation.

An ANN is normally designed, modeled, or configured for certain or distinct applications like stock market prediction, pattern recognition, data recognition, security and loan applications, image compression, and weather prediction.

ANNM aims at bringing conventional computers a bit closer to the way human brains work and is the best candidate for dealing with problems in which the relationship between the inputs and outputs is extremely non-linear. Figure 3 shows a simple ANN configuration or schematic consisting of three layers, namely, an input layer, one Hidden layer, and an output layer.

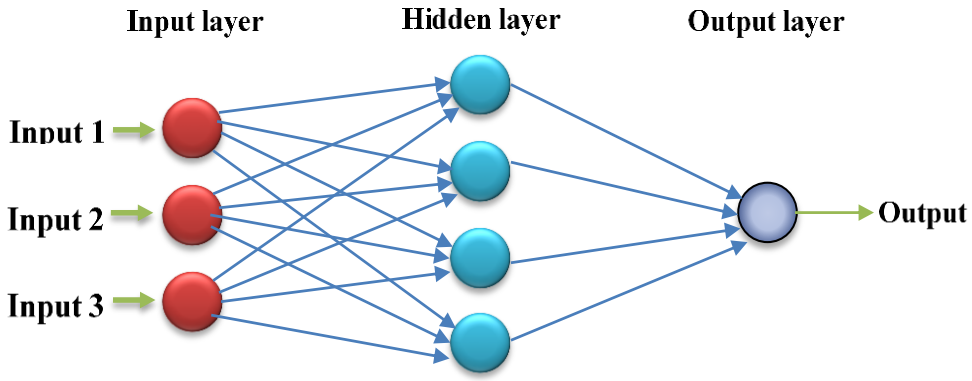

Figure 3: A three inputs single layer, one hidden layer, and one neuron output ANNs schematic

ANNs are modeling tools normally adopted in modeling non-linear statistical data and complex relationships between inputs and outputs as well as used in finding patterns in data. ANNs are indistinguishable from the concepts of connectionism in biological neural networks in that functions are carried out bunched together, collectively and in parallel, and in units instead of having a distinct delineation or description of sub-tasks. Connectionism is a method of studying human cognition in the fields of cognitive science that uses mathematical models called connectionist networks, or ANNs, to describe or elucidate mental phenomena in the form of highly interconnected neuron-like processing units.

ANNMs are designed based on emulation of the central nervous system in mind and are subjects of computational neuroscience and theoretical neuroscience.

Training the ANNs

For ANNs to produce or generate the desired outputs from a known set of inputs, we need to train them. Normally, the training is carried out by passing on or feeding teaching patterns to the network and allowing the network to evolve or change its weighting function based on some previously definitive or defined learning rules, which can either be supervised or unsupervised. In most ANNs, the backpropagation algorithm is adopted as a model in training the network (Goh, 1995; Saerens, 2002). When an ANN under investigation is placed under supervised learning, it means that inputs are giving it inputs and matching output patterns. In this approach, the results or outcomes are already known for specified or specific inputs. On the other hand, when ANNs under investigation are put under unsupervised learning, it means that the output of the network is trained to react or respond to input patterns. In this case, the outcomes are not known for specified or specific inputs beforehand.

Areas of application of ANNs

ANNs, as powerful models, have a broad range of application areas. ANNs have far-reaching applications that cut across many different fields in Medicine, Security, Banking/finance, agriculture, image processing, character recognition, Forecasting, Defense, Government, Stock Exchange Prediction (makes a stock market prediction), Traveling Salesman Problem (finds the shortest possible path to travel all cities in each area), engineering, etc.

4.0 CONCLUSION AND THE FUTURE OF SOFT COMPUTING

As computer technologies, sophistication, processing power, and miniaturization keep increasing, so will intelligent systems and soft computing (SC) paradigms grow in importance and acceptance. To make complex decisions in real life, the best choice of an outcome from myriad possibilities requires the use of complex algorithms and intelligent systems. Today, a great number of computers possessing high-speed processing power and enormous storage space are available in a great many research centers and universities, even at a reduced rate or cost.

The desire for SC method application and constructing systems that are intelligent has become very important, as observed in the power and recognition of the network of physical objects, aka "the Internet of Things (IoT)" concept. Today, using SC to build intelligent systems has also increased tremendously in importance.

Today, FL, ANN, GA, and ES are already used in many everyday domestic appliances, such as washing machines, cookers, and fridges; sowing machines; car washing and cleaning machines; kitchen blenders; etc. Moreover, SC has useful applications in sciences, industries, governments, the military, aviation, and commerce every day, and it is expected to continue to grow, as what we see today of SC is a drop of water and what is yet to be seen is an ocean.

REFERENCES

Buckley, J. J. and Hayashi, Y. (1994). Neural Networks. A survey. Fuzzy Sets and Systems 1994;66: 1-13.

Cichocki, A, Unbehauen, R. (1998). Neural Networks for Optimization and Signal Processing. West Sussex: UK. John Wiley & Sons.

Dote, Y., Hoft, R. G. (1998). Intelligent Control: Power Electronic Systems. Oxford: UK. Oxford University Press.

Driankov, D., Reinfrank, M., Hellendoorn, H. (1993). An Introduction to Fuzzy Control. Berlin: Germany.Springer; 1993.

Gao, X. Z and Ovaska, S. J. (1999). Friction compensation in servo motor systems using neural networks. in Proc. IEEE Midnight-Sun Workshop on Soft Computing Methods in industrial applications. Kuusamo: Finland; June 1999.

Goh, A.T.C. (1995). Back-propagation neural networks for modelling complex systems, Artificial Intelligence in Engineering 1995;9(3):143-151.

Holland, J. H. (1992). Adaptation in Natural and Artificial Systems. An Introductory Analysis with Applications to Biology, Control, and Artificial Intelligence. 2nd ed. Cambridge: MA. MIT Press;1992.

Jang, J. S. R, Sun, C T, and Mizutani, E. (1997). Neuro-Fuzzy and Soft Computing, A Computational Approach to Learning and Machine Intelligence. Upper Saddle River, NJ: Prentice-Hall; 19973.

Komori, Y. (1992). A neural fuzzy training approach for continue speech recognition improvement. in Proc. International Conference on Acoustics, Speech, and Signal Processing. San Francisco: CA; 1992: 405-408.

Pedrycz, W. (1990). Fuzzy sets in pattern recognition: Methodology and methods, Pattern Recognition 1990; vol. 23. no. 1: 121-146.

Roy, R., Furuhashi, T, Chawdhry, P. K. (1998). Advances in Soft Computing: Engineering Design and Manufacture. London.

Russo, F. (1996). Fuzzy systems in instrumentation: Fuzzy signal processing. IEEE Trans Instrumentation and Measurement. vol.45. no.2.

Saerens, M. (2002). Neural controller based on back-propagation algorithm. IE Proc. On Radar and Signal Processing 2002; 138(1): 55-62.

Whitley, D. A. (1994). Genetic algorithm tutorial. Statistics and Computing 1994; 4(2): 65-85.

Yuhas, B., Ansari, N. (1994). Neural Networks in Telecommunications. Boston: MA. Kluwer Academic; 1994.