Soft Computing as a solution to Time/ Cost Distributor

Nabil M. Hewahi
Department of Computer Science
Islamic University Of Gaza- Palestine
nhewahi@mail.iugaza.edu

Abstract. In this paper we present a theoretical model based on soft computing to distribute the time/cost among the industry/machine sensors or effectors based on the type of the application. One of the most unstudied significant work is to recognize which sensor in an industry for example has higher priority than others. This is important to know which sensor to be checked first and within time limits of the system response. The problem of such systems is their variant environmental situations. Based on these varied situations, the priority of the importance of each sensor might change from time to another. Due to this uncertainty and lack of some information, soft computing is considered to be one of the plausible solutions. The presented idea is based on initially training of the system and continuously exploiting the system experience of the degree of importance of the sensors. The proposed system has three main stages, the first stage is concerned with training the system to obtain the necessary system time to respond, the necessary time allocated to recognize which sensors to check (or which has higher priority), and the initial importance value for each sensor, which indicates the initial judgment about the sensor importance. The second stage is to use the system experience about the importance of the sensor using fuzzy logic to decide the final values of each sensor 's importance. Based on the output of the second stage and the output of the first stage, the system distributes the time/cost among the sensors (some sensors with lower priority might be neglected). The main idea of the proposed work is based on neurofuzzy

Keywords: soft computing, neural networks, fuzzy logic

Soft computing as a solution for Time/Cost Distributor

Abstract. In this paper we present a theoretical model based on soft computing to distribute the time/cost among the industry/machine sensors or effectors based on the type of the application. One of the most unstudied significant work is to recognize which sensor in an industry for example has higher priority than others. This is important to know which sensor to be checked first and within time limits of the system response. The problem of such systems is their variant environmental situations. Based on these varied situations, the priority of the importance of each sensor might change from time to another. Due to this uncertainty and lack of some information, soft computing is considered to be one of the plausible solutions. The presented idea is based on initially training of the system and continuously exploiting the system experience of the degree of importance of the sensors. The proposed system has three main stages, the first stage is concerned with training the system to obtain the necessary system time to respond, the necessary time allocated to recognize which sensors to check (or which has higher priority), and the initial importance value for each sensor, which indicates the initial judgment about the sensor importance. The second stage is to use the system experience about the importance of the sensor using fuzzy logic to decide the final values of each sensor 's importance. Based on the output of the second stage and the output of the first stage, the system distributes the time/cost among the sensors (some sensors with lower priority might be neglected). The main idea of the proposed work is based on neurofuzzy

Keywords: soft computing, neural networks, fuzzy logic

1 Introduction

We introduce in this section the soft computing and its applications. On the other hand we define the system which we present in this paper. It is time/cost distributor system.

1.1 Soft Computing

Soft computing (SC) is a term originally expressed by Lotfi Zadeh [18][19] to denote systems "exploit the tolerance for imprecision, uncertainty, and partial truth to achieve tractability, robustness, low solution cost, and better rapport with reality" [19]. Soft computing differs from conventional (hard) computing, unlike hard computing, it is tolerant of impression, uncertainty, partial truth and approximation. The human mind is the way in which soft computing work. SC techniques are a natural way of handling the inherent flexibility with which humans communicate, request information, describe events or perform actions. Soft computing has been divided into two groups namely knowledge driven reasoning such as fuzzy logic and probabilistic reasoning, and data driven search and optimization approaches such as neuro computing and evolutionary computing[18][19]. Soft computing is a partnership in which each of the partners contributes a distinct methodology for addressing problems in its domain. Based on this vision, the main constituent methodologies in SC are complementary rather than competitive. At present, the research activities of SC applications are focused in the areas of structural engineering, environmental engineering, geo-technical engineering, intelligent interfaces, information retrieval and intelligent

assistants. One of the good examples of a particularly effective combination is what has come to be known as "neurofuzzy systems". Such systems are becoming increasingly visible as consumer products ranging from air conditioners and washing machines to photocopiers and camcorders[11][16][17]. Other combinations could be a neural networks and genetic algorithms which is termed by "neuroevolution". Neuroevolution has proven very high capabilities in various applications and in reinforcement learning tasks [6-10,12-15]. In difficult real-world learning tasks such as controlling robots, playing games, or pursuing or evading an enemy, there are no direct targets that would specify correct actions for each situation. In such problems, optimal behavior must be learned by exploring different actions, and assigning credit for good decisions based on sparse reinforcement feedback. Comparing neuroevolution to the standard reinforcement learning, neuroevolution is often more robust against noisy and incomplete input, and allows continuous states and action naturally. Much of the research in neuroevolution is on control tasks such as pole balancing and mobile robot control. Some other applications are related to industry controllers. Other existing combinations is the combination of neural networks, genetic algorithms and fuzzy logic. Such systems area used in industry, medicine, prediction and game playing [2][6][9][12][13][15].

1.2 Time/Cost Distributor System

Some of the very common systems for applications is applying Neural networks, genetic algorithms, fuzzy systems, evolutionary computing or combination of them in the real time industry system. In all the applications, the used technology works to simulate, control or improve the performance of the industry. None of these trials considered the system response time. Due to some factors, the industry should take a certain action. The problem is to know which sensors should take more /less time to be checked to allow the system to take the proper action within the time limit. Based on the industry situation, the needed sensors to be checked differ from time to time. We shall define the Time/Cost Distributor System (T/CDS) as a system that is responsible for distributing the given time to the sensors based on their importance to give the system the opportunity to respond within the time limit. This means, in certain cases all the sensors might be checked, whereas in some other cases some of them are checked. Figure 1 depicts T/CDS. As shown in Figure 1, the surrounding environment is the input of the T/CDS. T/CDS distributes the time to be obtained by one of its stages among the sensors to be checked based on their importance according to the current situation. The output of T/CDS is the the time slot allocated to each sensor to be checked. TS_i in the figure means the slot of time allocated to the ith sensor.

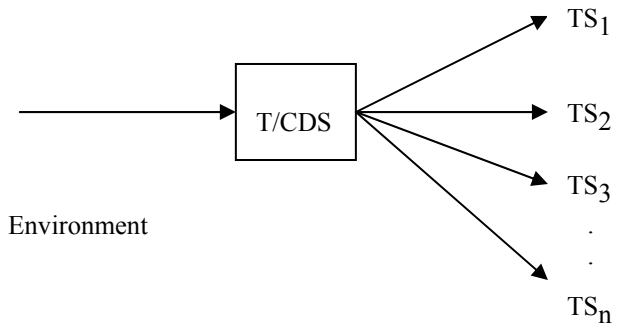

Figure 1. Time/Cost Distributor Systems

Some of the applications that might use T/CDS are:

1. Robots that play soccer. At certain position(mostly), the robot has to know where to pass the ball very quickly (might not check all his surroundings), otherwise, one of his opponents might come and get the ball.

2. Automatic pilot in cases of emergency. A very fast response is required based on the situation or the plane might get crashed.

3. Games where players should do some action or otherwise destroyed by other player.

4. Industries and controllers

Another version of the same time distribution for controllers is the cost distributors. Cost distributors can be used in economic and commercial applications. It can also be used in information retrieval based on speed, memory and the size of the databases.

2 Research Objectives

Designing T/CDS which is able to decide which sensor should be given more/less priority in a given environmental situation, or even which is to be neglected is the main objective of this research. This increases the ability to take the appropriate action within time limits. The T/CDS should be able to decide the necessary time limit for the whole system to respond, and the time limit necessary to check the sensors with higher priority. To simplify this process, we consider T1 as the time limit for the system to respond. T2 is the time to be lost to check the selected sensors. T3 is T1-T2 which is the remaining time for the system response. The proposed T/CDS is based on soft computing and more specifically on neurofuzzy system. Soft computing is used here to overcome the problem of uncertainty, partial truth and approximation. In [10], T/CDS system based on neuroevolution for real time system controllers is proposed. The proposed system has four main stages, the first one is to decide the time constraints based on the given environment surroundings, the second stage is to distribute the time/cost to determine the importance of each behavior based on the decided time by stage one. Stage three is to take the output of stage two to place appropriate controller which finally applied to the forth stage to recognize the final action of the system. It is shown how the proposed system can be applied on a soccer robot example. The main difference between this approach and the approach which we propose is that the system in [10] is based on neuroevolution and fitness function to decide the degree of the importance of the sensor, whereas in our proposed system as we shall see, the degree of the importance of a sensor is based on the system experience which finally uses the neurofuzzy to decide it**.** In general, neuroevolution technique is good when no enough examples can be provided, its performance depends highly on the fitness function which is in common not easy to have an optimum one. On the other hand, neurofuzzy is basically based on uncertainties and lack of information, moreover, it is in general faster than neuroevolution in such kind of problems.

Deciding the necessary time (changeable) to perform the action which is changeable in real time applications, is one of the very challenging and not yet widely tackled

problems. This problem is difficult to solve using neural networks alone because in many different situations the time needed and the action to be taken is changeable. This research is a continuous research started in [10]. It is to explore and investigate a solution to the posed problem of T/CDS using neurofuzzy technique. The purpose of this research is to help other researchers to tackle the problem of T/CDS in a near future.

3 The Proposed System

The proposed system is based on three stages :

1. Time/ sensor decider : This stage is concerned with deciding the time limit of the system and the time needed for the sensors to be checked. In addition this stage is concerned with specifying an initial value for each sensor. This value is to indicate an initial impression about the importance of the system. This stage is based on Backpropagation algorithm since several examples can be provided. The input for this neural network is the environment inputs and the output is the T1, T2 and the initial importance value for each sensor based on the given environment inputs. This stage is firstly used alone to train the system, then used as apart of the T/CDS system to get the values of the sensors, and T1 and T2.

2. Sensor priority decider : Deciding the final value of the importance of each sensor is the main goal of this stage. In this stage soft computing is used. In this stage we take each sensor importance value obtained from stage 1 as input to a fuzzy membership function (fuzzy set) and the old experience about the importance of this sensor in various environment situations. The experience importance value is obtained by getting always the average value of the importance value of the sensor. The experience value is passed to a fuzzy set. The initial value for the experience sensor value is 0. The stage uses these two inputs to produce the final value of importance for the sensor in the current situation. This is done by a constructed fuzzy rules and defuzzifying by using center of gravity or Sugeno-style inference. Using the system previous experience of the importance of the sensor is very significant. This will help the system to scale up always the importance of the sensor and how often it is used.

3. Selecting the sensor: One of the important inputs for the T/CDS is the time needed by each sensor which is known. Knowing the importance of each sensor, the sensors are ordered based on their importance. The sensors to be checked are selected based on their priority order and T2.

Figure 2 shows the stages of TC/DS. A detailed explanation about the proposed system is provided in the next sections.

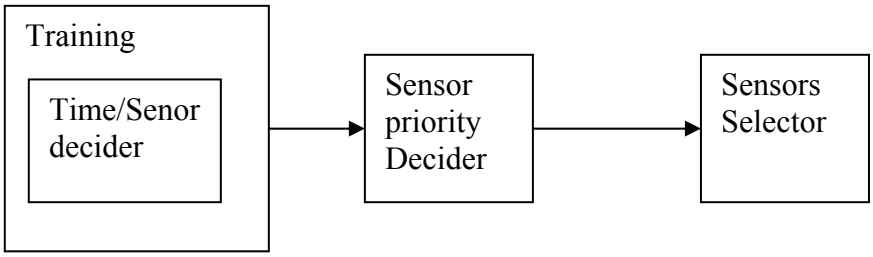

Figure 2. TC/DS proposed system

3.1 Time/sensor decider

In this stage, time needed for the system to respond, the time needed to check the sensors and the values the specify the initial importance of the censors are produced. This output is based on the current situation of the environment. The current situation of the environment is the description of the situation. To make it clear, if the situation we have may produce a voice and a shape might be seen, the sensors related to voice recognition and vision are necessary, whereas a sensor related to touch might not be important in this case. Another situation is based on touch only which means the touch sensor is the only (the most important) sensor needed in this case. Knowing this, we can train a neural net using backpropagation. In our training, in addition to the sensor importance, we can also provide the time needed for the system to respond and the time needed to check the higher priority sensors. Figure 3 shows the Time/ sensor decider. The importance sensor in the figure and later in the text is termed as IS_j to indicate the importance of the jth sensor. In case where no enough examples can be provided, neuroevolution technique can be used to find the values of T1 and T2 [10]. This needs a good fitness function based on the factors related to the problem domain. In addition, certain genetic operators are to be used properly.

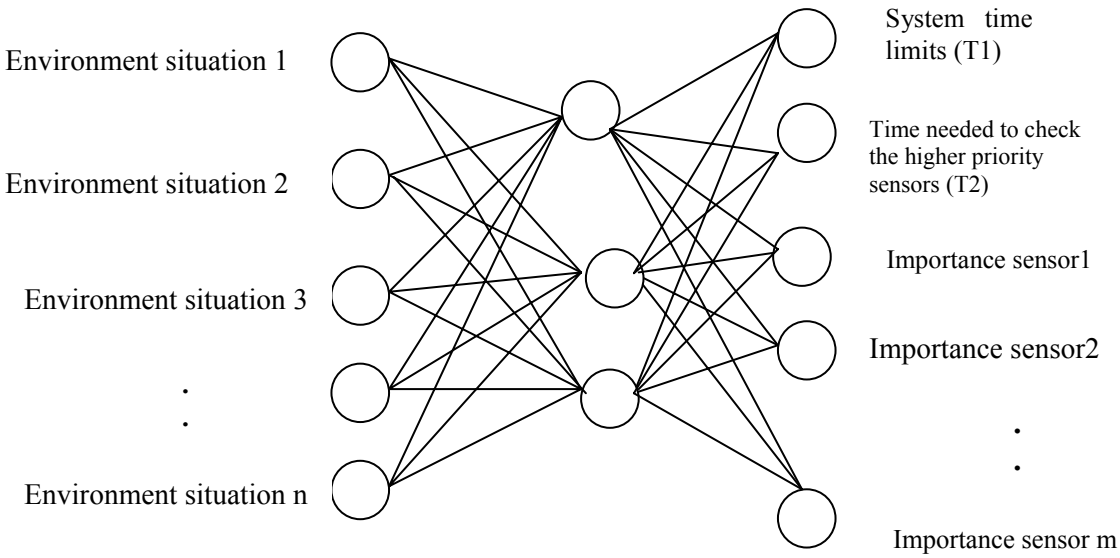

Figure 3 . Time/sensor decider

3.2 Sensor priority decider

This stage is to decide the final value for the sensor importance. The inputs of this stage are obtained from the first stage. For each sensor, there is a subsystem to obtain its final importance value. To find the final value of the sensor importance, inputs of the subsystem are inputs for a membership functions. The output of the membership functions are passed to fuzzy rules and finally defuzzifying is applied to get the final value of the sensor importance. The first input is SI_j, where the second input is the average value of the jth sensor in each environment situation and indicated by AVG_j. The initial value of AVG_j is 0.To clarify this point, let us assume that we have tried n arbitrary number of environment situations, and let us assume further that IS_{jk} is the importance value of the jth sensor in the kth environmental situation. Then

$$AVG_j = \left(\sum_{k=1}^{n} IS_{jk} \right) / n$$

The computation of the AVG_j is considered to be a main factor of the final decision to reflect the importance of the jth sensor over various situations. This step will help the system to learn from its experience.

An example of some of the rules that might be used in such as a system :

IF (IS_{jk} is low) and (AVG_j is low) Then (IS_{jk} is low)
IF (IS_{jk} is low) and (AVG_j is high) Then (IS_{jk} is moderate)
IF (IS_{jk} is high) and (AVG_j is high) Then (IS_{jk} is high)
IF (IS_{jk} is Med.) and (AVG_j is low) Then (IS_{jk} is low)

These rules are absolutely domain dependent and based on the used membership functions. In the final stage defuzzifying is applied to obtain the final value of the sensor importance (FIS). This is done by any of the methods of defuzzifying such as Center of Gravity or Sugeno-Style inference. Based on the FIS value for each sensor, It would be very simple to order the sensors. The higher is the value of the FIS for the sensor, the more important is the sensor. Figure 4 shows the sensor priority decider stage.

3.3 Selecting the sensor

In this stage, the outputs of the first and second stages are used as inputs. The inputs of this stage are the final importance sensor values (FIS) for all the sensors and the time needed to decide the importance of the sensors (T2). The importance of this stage is to distribute the T2 among the sensors based on their importance values. Each sensor 's needed operation time is known. This will help in deducting the operation time of the chosen sensor based on the priority from T2. This process will continue until the T2 is over. Some of the special cases regarding the left time of T2 and the time of the selected sensor 's operation time might be considered. In some cases, the left time of the T2 is less than the necessary operation time for the selected sensor. In

this case, the next priority sensor 's operation time is checked. Figure 5 shows selecting the sensor stage.

To make a clear idea about selecting the sensors, let us consider Table 1. It is assumed in the table that T1 is 30 and T2 is 10. Therefore, T3 = T1-T2 and is equal to 20. We assume further that we have six sensors in our system, each has a specific time to be checked (sensor requested time). The sensor priority in the table is assumed to be obtained after getting the FIS for each sensor. Based on the T2, this time has to be distributed among the sensors of the higher priority. Sensors 4,1,3, and 6 are chosen in order to be the sensors to be checked. It is to be noted that sensor 3 should be chosen instead of sensor 6, but because sensor 3 needs more time, which makes the total time (summed time of the selected sensors) exceeds T2, sensor 6 is chosen instead. A very important note can be considered here, the sensor requested time can be one of the main factors in the fuzzy rules or fuzzy sets to decide about the degree of the importance of the sensor instead of doing the procedure of exchanging the priority of the sensor 6 with that of sensor 3.

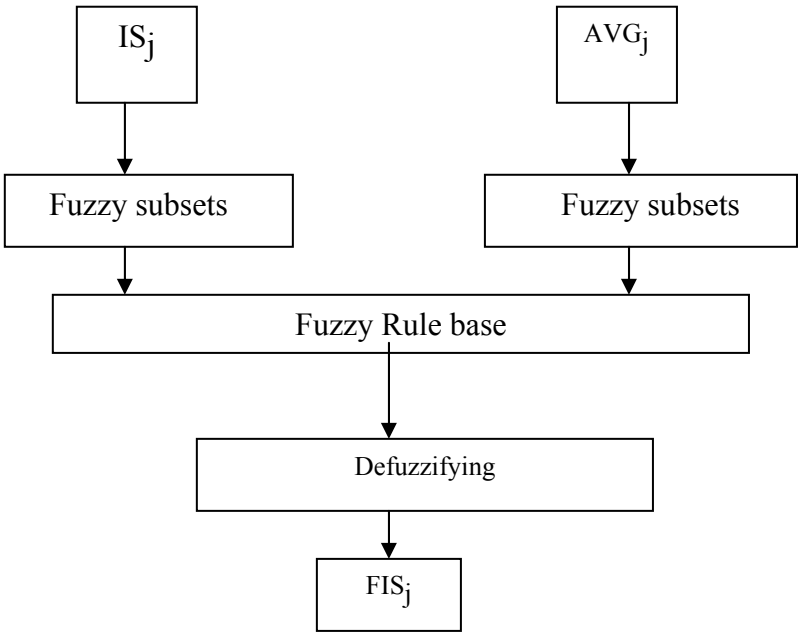

Figure 4. Sensor Priority Decider

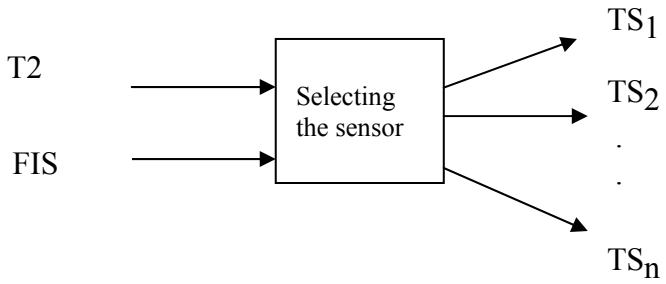

Figure 5. Selecting the sensor stage

T1	T2	T3	Sensor number.	Sensor requested time	Sensor priority	Selected sensors time
30	10	20	1	5	2	5
			2	3	4	0
			3	2	3	2
			4	1	1	1
			5	4	6	0
			6	2	5	2

Table 1. An example of the selecting the sensor stage of the T/CDS

4 Conclusions

In this paper we tried to focus on a spot of research which has not been tried extensively. A theoretical model to solve the problem of distributing a slot of time to decide which sensors in industry or controller have more importance and effect than others in system response within time limits is developed. The main problem is that, considering all the sensors to take a decision might lead to inappropriate response time. Due to the lack of information and uncertainty emerged from various environmental situations, soft computing is used as a key operator to the developed system. One of the main points of the proposed system is its dependence on its experience about the history of the degree of the importance of the sensor. The developed system is based on three stages, training and producing the initial importance values for the sensors, obtaining the final values of the importance values for the sensor, and finally determining which sensors to be checked to take the action of the system within time constraints. This paper is the first part of a sequence of continuous work. It 's main goal is to help researchers to widen their perspective towards a solution to this yet not solved problem. Some of the future directions are 1. Exploring other solutions to the time/cost distribution problem 2. Implementation of the proposed system and apply it on various applications.

References

[1] A. Agogino, K.Stanley and R.Miikkulainen, Online interactive neuro-evolution, Neural Processing Letters, 11, pp 29-37, 2000.
[2] A.Conradie, Risto Miikkulainen and C.Aldrich, Adaptive control utilizing Neural swarming, Proceedings of the 2002 Genetic&Evolutionary Computation Conference (GECC-2002), 2002.
[3] G.Drake and J.Smith, Simulation system for real-time planning, scheduling and control, Proceedings of 28th Conference on Winter Simulation, pp 1083-090,1996.
[4] J.Fan, R.Lau and R.Miikkulainen, Utilizing domain knowledge in neuroevolution, Proceedings of the Twentieth Inter. Conf. On Machine Learning(ICML-03) Washington,Dc, 2003.
[5] R.Florian, Evolution of alternate object pushing in a simulated embodied agent: Preliminary report, Center for Cognitive and Neural Studies (Coneural), Romania, August, 2004.

[6] F.Gomez , Robust non-linear control through neuroevolution, Ph.D dissertation, The university of Texas at Austin,USA, 2003.
[7] F.Gomez and R.Miikkulainen, Transfer of Neuroevolved Controllers in Unstable Domains, Proceeding of The Genetic Evolutionary Computation Conference (GECCO 2004), 2004.
[8] F.Gomez and R.Miikulanien, Active Guidance for a Fitness Rocket using Neuroevolution, Proceedings of Genetic Evolutionary Computation Conference (GECC-03), 2003.
[9] N. Hewahi, Engineering Industry Controllers Using Neuroevolution, Journal of AIDAM: Artificial Intelligence for Engineering Design, Analysis and Manufacturing, USA,19,pp 49-57, 2005.
[10] N.Hewahi, Neuroevolution Based Time/Cost Distributor For Real Time System Controllers, Proceedings of the 2nd Inter. Conference on Information Technology, Amman, Jordan, pp 189-193, 2005.
[11] D. Hong, C.Hwang, Abrief Introduction to Soft Computing, Proceedings of the Autumn Conference, Korean Statistical Society, pp 65-66, 2004.
[12] D.Law and R.Miikkulainen, Grounding Robotic Control with Genetic Neural Networks, (Technical Report AI-94-223). Department of Computer Sciences, The University of Texas at Austin, 1994.
[13] S.Nolfi and D.Floreano, Evolutionary Robotics, MIT press, Cambridge, 2000.
[14] S.Nolfi and D.Parisi, Evoultion of Artificial Neural Networks, in M.A.Arbib(Ed.) Handbook of Brain Theory and Neural Networks, 2nd edition, Cambridge, MA: MIT press, pp 418-421, 2002.
[15] K.Stanley and R.Miikkulainen, Efficient evolution of neural network topologies,Proceedings of the 2002 Congress on Evolutionary Computation(CEC'02), 2002.
[16] S.Taheri, Trends in Fuzzy Statistics, Australian Journal of Statistics, 32,pp.239-257,2003.
[17] H.Takagi, S. Kamohara, T. TAKEDA, Introduction of Soft Computing Techniques to Welfare Devices, IEEE Midnight-Sun Workshop on Soft Computing Methods in Industrial Applications (SMCia'99), Kuusamo, Finland, June 16-18, pp 116-121, 1999.
[18] L.A.Zadeh, Fuzzy logic, Neural Networks and Soft Computing, Comm. Of ACM, Vol.37,No.3,pp. 77-84, March 1994.
[19] L.A.Zadeh, Soft Computing and Fuzzy Logic, IEEE Software, Vol.11, No.6,pp 48-58, 1994.