A Swarm Intelligence inspired Autonomic Routing Scenario in Ubiquitous Sensor Networks

Jin Wang, Brian J. d'Auriol, Young-Koo Lee, Sungyoung Lee*

Department of Computer Engineering,
Kyung Hee University, Korea
{wangjin, dauriol, sylee}@oslab.khu.ac.kr
{yklee}@khu.ac.kr

Abstract

Autonomic computing has attracted large amount of attention as a novel computing paradigm in the past few years. In this paper, we explore the inherent accordance between autonomic computing and swarm intelligence. Then, we propose a swarm intelligence inspired autonomic routing scenario with a targeting application area in ubiquitous sensor network. This scenario covers most of the characteristics of autonomic computing. The working flow and steps of our SI inspired autonomic routing scenario are explained in detail together with some preliminary simulation results, such as the power consumption, delivery ratio etc.

Keywords: Autonomic Computing, Swarm Intelligence, Routing, Sensor Network, Power-aware.

1. Introduction

With an ever-increasing usage of the distributed and heterogeneous computing devices in a ubiquitous environment, the need has been to manage them unsupervised and autonomously [1]. This autonomous style of management is referred to as autonomic computing [2] which includes at least four basic characteristics, namely self-configuration, self-optimization, self-healing, and self-protection. The purpose is that autonomic computing system will make the job of human administrator more relaxed, since the complexity of managing heterogeneous computing units is getting too large to be handled in current scenarios.

A novel alternative approach to reduce system complexity is to make use of the Swarm Intelligence (SI) mechanism. A SI inspired system is a self-optimizing system with four main ingredients, which are positive feedback, negative feedback, amplification of fluctuations and multiple interactions among multi-agents [9]. Strangely, it also covers the four aspects of autonomic computing to some degree. In SI inspired systems, many insect like agents are interacting *locally* with one another and with their environment, just like the relationship between autonomic elements and their managing environment, and finally some global optimized performance can be achieved, such as to find the shortest path from the nest to the forage place. Usually, the agents are simple and small in size, just like the insects which have only few hundreds or thousands of neurons. So, the software packets can also be small in size. Also, the SI inspired systems are robust and the tasks can be fulfilled through the interaction among the multiple agents with faster convergence. In the next section, a detailed comparison between the SI inspired mechanism and autonomic computing will be presented.

Due to the inherent accordance between autonomic computing and swarm intelligence, it is reasonable to combine them together and to apply them to some distributed and ubiquitous environment, such as the Ubiquitous Sensor Network (USN). To the best of our knowledge, little work has been done to combine the mechanism of SI with autonomic computing and then to apply them to some ubiquitous applications.

Our contribution in this paper lies in the following three aspects. First, we explore the inherent accordance between autonomic computing and swarm intelligence mechanism. Then, we propose a SI inspired autonomic routing scenario in a ubiquitous context aware application environment. Finally, we present our preliminary simulation results and analysis of some performance metrics, such as agent number, packet delivery ratio and power consumption.

* Professor Sungyoung Lee is the corresponding author.

2. Comparison between Autonomic Computing and Swarm Intelligence

Autonomic computing [1-3] system is a self-managed computing system with minimum human consciousness or involvement, and with an ultimate goal to free administrators from the burden of system operation and maintenance. It has at least four basic ingredients, which are self-configuration, self-optimization, self-healing, and self-protection. Different aspects of autonomic computing are explained in [4-6] with specific application emphasis. Opportunities and possible research directions of autonomic computing in the system engineering field are well explained in [7], and [8] gives a concise and comprehensive introduction about autonomic computing.

The principles of Swarm Intelligence (SI) [9] were originally inspired by the observation of various natural and social phenomena. Even though these social insects have only few hundreds or thousands of neurons and they simply collaborate with their neighbours, some high level intelligence is observed. Thus, the idea that subcomponents are very simple while the overall system manages itself adaptively is very tempting and promising from an engineering perspective. Ant colonies apply this principle, which has inspired a new research direction called Ant Colony Optimization (ACO) [10].

The SI inspired system is a self-organizing system, which includes positive feedback, negative feedback, amplification of fluctuations and multiple interactions among multi-agents [9]. The relationship between autonomic computing and swarm intelligence is tightly coupled according to Table 1. The middle column in Table 1 is a concise interpretation which bridges both autonomic computing and swarm intelligence. For example, through the tuning of system parameters, the self-configuration aspect can be achieved autonomically rather than manually. In the mean time, the pheromone can get evaporated, reinforced, thresholded according to [9], which represents the actual parameters under dynamic system situation. From this, we can see that both of them are tightly coupled. It is obvious that the system performance can get optimized through the local interaction among the multiple ant agents. Also, the system robustness can be guaranteed through this kind of parallel multi-agent interaction. Finally, the security and load-balancing can be fulfilled with careful parameter and rule design. In the next section, the readers can get a deeper understanding of their inherent accordance.

Table 1. Relationship between autonomic computing and swarm intelligence

Autonomic Computing	Relationship	Swarm Intelligence
Self-Configuration	System parameters tuning etc.	Pheromone based operator
Self-Optimization	Optimized system performance etc.	Global optimal performance through local interaction
Self-Healing	Robustness, convergence rate etc.	Multi-agent interaction, (parallelism)
Self-Protecting	Security, load-balancing etc.	Pheromone threshold, sequence number etc.

3 A-CAMUS Architecture

Our work in this paper is influenced by the earlier work relating to the Autonomic Context-Aware Middleware for Ubiquitous Systems (A-CAMUS) architecture for ubiquitous sensor networks [11-13]. This project been undertaken for more than 3 years and a "Smart Office" test-bed has been built. We include a brief discussion about A-CAMUS here as background and related work.

In Fig.1 lists our A-CAMUS architecture.

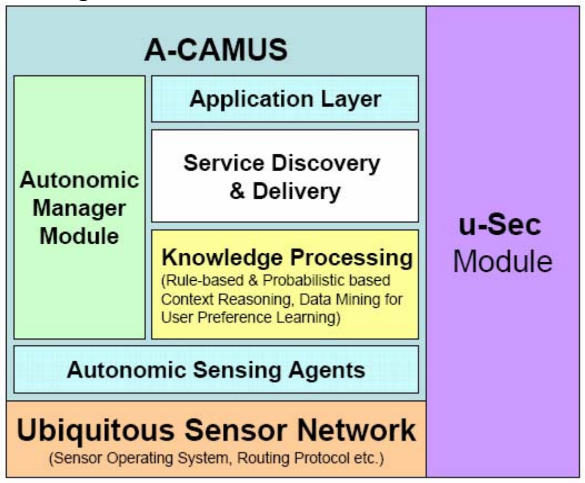
A-CAMUS
Application Layer
Autonomic Manager Module
Service Discovery & Delivery
Knowledge Processing
(Rule-based & Probabilistic based Context Reasoning, Data Mining for User Preference Learning)
Autonomic Sensing Agents
Ubiquitous Sensor Network
(Sensor Operating System, Routing Protocol etc.)
u-Sec Module

Fig. 1. An A-CAMUS Architecture

The ubiquitous sensor network lies in the bottom layer. Some sensing devices, such as Berkeley-mote, RFID devices etc. are deployed in the real environment, which form a ubiquitous sensor network. These devices can collect the original information and then send this raw data to the upper knowledge processing layer for further processing with the help of autonomic sensing agents, which is defined as software. During this process, routing is an utmost important task since it should not only send the related data from the source to the destination correctly, but that it should meet the constraints and provide certain QoS requirement. In this paper, we try to solve this routing problem by using swarm intelligence mechanism.

4. Swarm Intelligence inspired Autonomic Routing Scenario

Based on the inherent accordance between autonomic computing and swarm intelligence, we now provide a SI inspired autonomic routing scenario in this section.

4.1 Routing Scenario

Fig. 2 shows three modules of our agent based autonomic routing scenario which are Autonomic Agent (AA) manager, Service Manager and Context Manager. We will add more functionality into the modules and verify it based on our "Smart Office" test bed later.

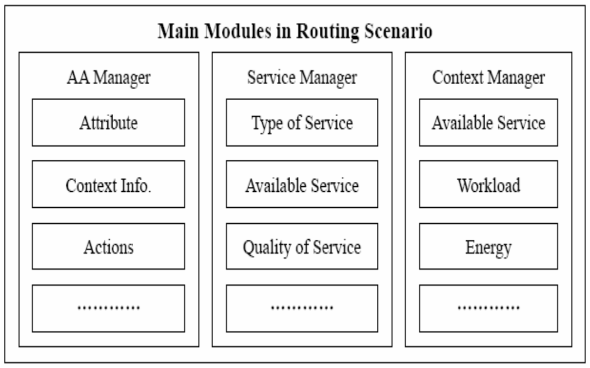

Fig. 2. Autonomic routing scenario

TABLE 2 DATA STRUCTURE OF AA MANAGER

Attributes	Unique ID, Timestamp, Remaining energy, Service type etc.
Context Information	IP, Available service, Hop number, Pheromone distribution, Humidity, Temperature, Lighteness, latency, etc
Actions	Generation, Migration, Service discovery, Pheromone evaporation, Pheromone aging, Pheromone reinforcement, etc.

Autonomic Agent (AA) manager consists of three main parts, which are attributes, context information and actions, as is shown in Table 2. Autonomic Agents (AAs) will collaborate with each other and configure themselves so as to achieve an optimal performance based on SI mechanism. While in the mean time, they will prevent the system from being over-loaded and malfunctioned to fulfill the self-protecting function. A Service Manager is in charge of different services, like routing, printing services etc. Then, through the communication among AAs, different types of services can be collected and exchanged in a localized and real-time manner. A Context Manager is like a repository to store the context and provide the best service so as to fulfill the self-optimization function.

An SI inspired routing scenario based on these three modules is proposed in Fig. 3. From which, we can conclude its basic steps as follows:

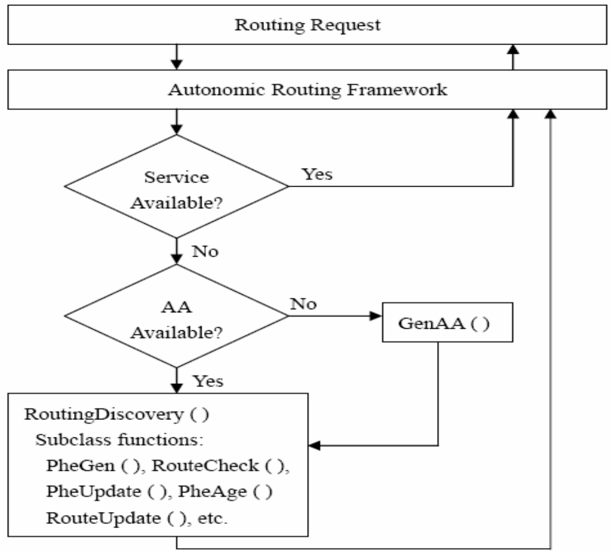

Fig. 3. Autonomic routing work flow

Step 1: Originally, each AA is piggybacked into the hello packet and periodically broadcasted to collect some information, such as available services, remaining energy, distance, next hop etc. The SI mechanism is used with pheromone evaporation, reinforcement and aging.

Step 2: When there is some specific routing request originated from one node, it will follow the autonomic routing work flow in Fig. 3, which consists of many closed control loop, as is the same with the autonomic computing.

Step 3: Then, it will first check the Service Manager in Fig. 2 to see whether the service is available, namely whether the destination node is in its routing table. If it is available, it will check the context manager to get the best candidate and return it to the original node finally.

Step 4: If the service is not available, it will contact the AA Manager rather than search in the Context Manager. If AA is attacked or malfunctioned, it will autonomically generate another one with the uniform data structure shown in Table 2. In this way, a self-healing function is fulfilled here. If AA is available, the service discovery process will be initiated so as to find the required service. Through the localized interaction between AAs, the related services can be found, like to find the shortest path between source node and destination node. During this process, some actions are taken, such as pheromone generation, updating, and routing updating etc. to fulfill the self-configuration function.

Step 5: Once specific route is found, the context information is sent back to the Service Manager and Context Manager. And finally, the best service is decided and provided to the users. Here, another closed control loop is formed.

4.2 Routing Phase

The routing phrase can be divided into three phases, which are route setup, route maintenance and route failure handling phase. Here, we will illustrate this procedure with an example depicted in Fig. 4.

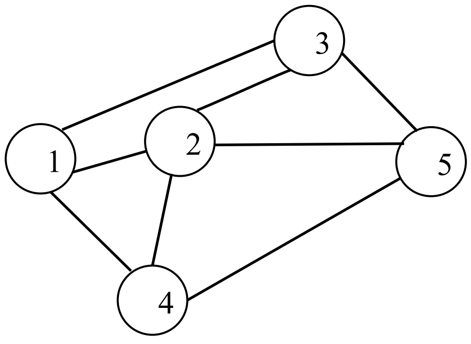

Fig. 4. 5 nodes network topology

During the route setup phase, each node will select its next hop node to forward the traffic packet according to its routing table and pheromone table. Taking node 1 as an example, its pheromone table is shown as in Table 3. The row in pheromone table represents its neighboring neighbors and column represents the destination node. The entry means the probability to get to the destination node through the neighboring nodes. Once node 1 has packet to send to node 5 and it has no pheromone information about node 5, it will initiate route setup phase. If the intermediate node has no information about destination node, it will continually broadcast until the broadcast packet reaches the destination node. Finally, the route is setup on reverse direction. It is worth noting that, the pheromone value is only updated during the backward route setup phase so as to represent the latest update of dynamic network. The selection of the probability entry $p_{i,j}$ can be found in [9].

Table 3 Pheromone table of node 1

	2	3	4	5
2	$p_{2,2}$	$p_{2,3}$	$p_{2,4}$	$p_{2,5}$
3	$p_{3,2}$	$p_{3,3}$	$p_{3,4}$	$p_{3,5}$
4	$p_{4,2}$	$p_{4,3}$	$p_{4,4}$	$p_{4,5}$
5	$p_{5,2}$	$p_{5,3}$	$p_{5,4}$	$p_{5,5}$

During the route maintenance phase, the source-destination pair is maintained either through periodical hello packets or data packets to save routing overhead. In the meantime, the pheromone value is either reinforced or evaporated to represent the real network situation. Since the selection of next hop is probability based, the work load on one route can be shared by the other route to make load-balancing. In this way, the functionalities of self-configuration, self-optimization and self-protecting are achieved.

Finally, once there is a link failure due to overload or out of power, the local repair can be adopted so as to search for alternative route. If impossible, a new route setup phase will be initiated.

5. Experimental Setup and Results

5.1 Experimental setup

Based on our previous experience [14], we set our simulation environment as follows. There are N nodes randomly placed within a range of 100 by 100 m^2, with a uniform transmission range R of 30m. A Random Waypoint mobility model is adopted here and their velocity vary from 0 to 20 m/s. Taking N=20 as an example, we will see the deployment as in Fig.5.

Unlike the broadcasting mechanism, we set our neighbor selection criteria as follows:

$$p_{ij}(t) = e^{-\tau_i(t)} * e^{-d_{ij}/R},\ (d_{ij} < R) \qquad (2)$$

$$\tau_i(t+1) = \tau_i(t) + \Delta(t,t+1),\ \tau \in [0,1] \qquad (3)$$

$$\Delta(t,t+1) = \begin{cases} K * \dfrac{1}{2 \cdot N} & reinforced \\ (\rho - 1) * \tau_i(t) & aging \end{cases} \qquad (4)$$

here, $\tau_i(t)$ is the pheromone distribution of node i at time t, d_{ij} is the distance of <i, j> and R is the maximum transmission range. From (2), we can infer that the smaller d_{ij} and $\tau_i(t)$ is, the higher $p_{ij}(t)$ will be and we will choose higher $p_{ij}(t)$ as the candidate node. In other words, this algorithm will prefer those nodes which are either nearer to the transmission node or being less visited with less pheromone so as to save energy. After each hop, the pheromone will automatically be updated according to (3) and (4), where the pheromone will either be reinforced according to the number of visiting agents K or aged by a factor of ρ (normally between [0.9, 1]) and N is the neighbor number.

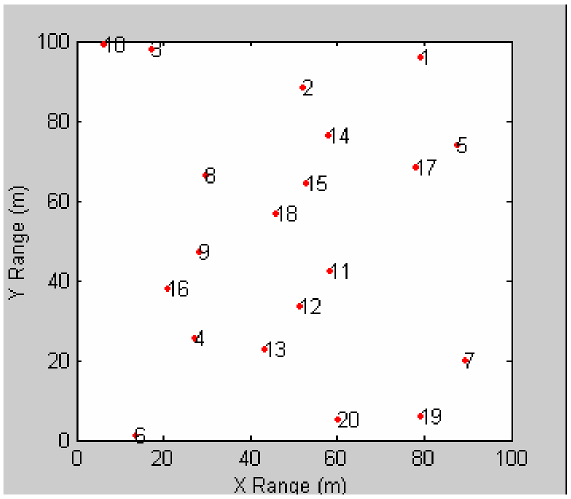

Fig. 5. N (20 in this case) node random deployment

5.2 Study of Agent Number

We will first study how many autonomic agents is need for each node so as to fulfill the routing task. If the agent number is too small, it will not find the related route or the response time will be too large to accept. If the agent number is too large, the routing overhead will be a burden and it is similar to the broadcasting mechanism, which we try to avoid in this paper.

Table.4 shows the average number of neighbors for each node. The total node number varies from 20 to 100 and the average node neighbor increases almost linearly with it. So, we tentatively set our agent number as K=**min** (N, 6) here and we will try to validate it later.

TABLE 4 AVERAGE NEIGHBOR FOR DIFFERENT N

N	10	20	30	40	50
Neighbor	2	4	6	8	10
N	60	70	80	90	100
Neighbor	12	14	17	20	22

5.3 Study of Packet Delivery Ratio

To verify the agent number, we randomly generate 300 routing connections between node i and j. In Fig. 6, we make a comparison between one agent approach and our approach in the aspect of packet delivery ratio. From this figure, we can see that there is no reliability guarantee if the agent number is too small. For the connection of (1, 14) and the node deployment in Fig. 5, if there is only one agent, the routing sequence will be {1, 5, 15, 6, 20, 18, 17, 13, abort}. If there are K agents like our approach, it will first take nodes {5, 16} as its first hop node. Then, node 5 will detect node 14 as the destination node and return this information to the source node 1. It only takes 2 hops. And another routing sequence can also be found through {1, 16, 19, 18, 9, 8, 14}. So, the successfulness of finding a reliable route can be ensured in our approach.

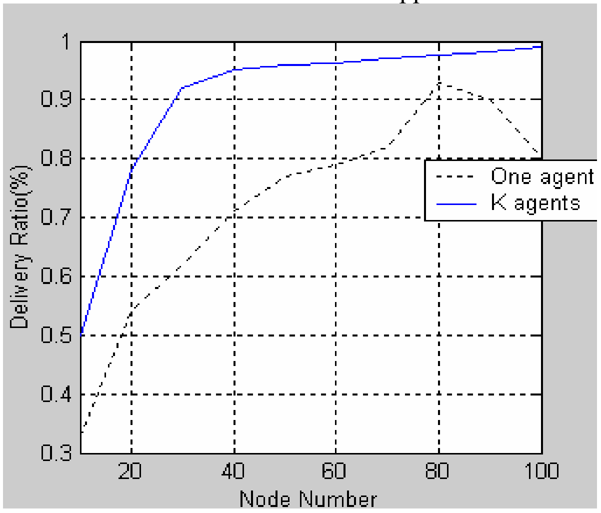

Fig. 6. Packet delivery ratio for different approaches

5.4 Study of Power Consumption

Finally, we will study the power consumption and make a comparison between the broadcasting mechanism and our approach. According to our previous work in [15], we assume the power consumption is proportional to the square of distance and the data length is the same.

Fig.7 shows the power consumption for different approaches, from which we can see that our approach is always better than the broadcasting mechanism. And the larger N is, more energy will be saved. So, the self-optimization aspect can be once again achieved.

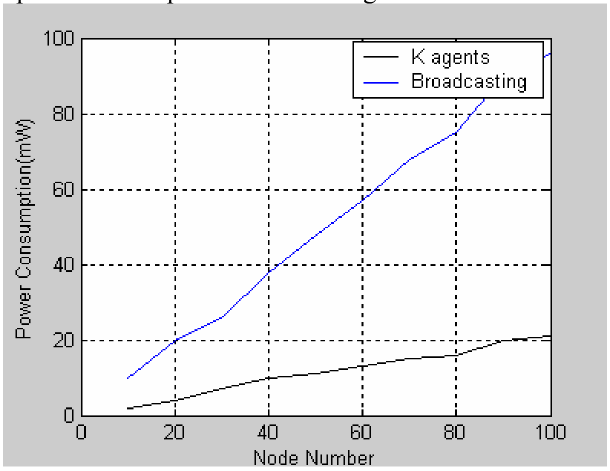

Fig. 7. Power consumption for different approaches

6. Conclusion

We compare and draw some inherent accordance between autonomic computing and swarm intelligence mechanism. Then, we introduce a Swarm Intelligence inspired autonomic routing scenario under a context-aware ubiquitous environment. The autonomic routing scenario and routing phase are explained in detail, and some preliminary experimental results are provided with satisfactory performances.

For the present, an agent based "Smart Office" test bed has been built in our professor's office. With the aid of various sensors, actuators and protocols, we intend to perform extensive experiments based on our SI inspired autonomic routing scenario, and introduce more modules and functionalities in the near future.

Acknowledgements

This research was supported by the MIC(Ministry of Information and Communication), Korea, under the ITRC(Information Technology Research Center) support program supervised by the IITA (Institute of Information Technology Advancement)" (IITA-2006-C1090-0602-0002).

6. References

[1] R. Murch, "Autonomic Computing," Upper Saddle River, NJ: Prentice Hall, Mar. 2004, ISBN: 013144025X.

[2] IBM, "Autonomic Computing Initiative," IBM Press, 2003, http://www.autonomic-computing.org.

[3] J. Kephart and D. Chess, "The Vision of Autonomic Computing," IEEE Computer, vol. 36, no. 1, January 2003.

[4] Belecheanu, R. A., Jawaheer, G., Hoskins, A., McCann, J. and Payne, T, "Semantic Web Meets Autonomic Ubicomp," In Proceedings of The 3rd International Semantic Web Conference, Hiroshima, Japan, 2004.

[5] F. Schintke, T. Schütt, A. Reinefeld, "A Framework for Self-Optimizing Grids Using P2P Components", 14th Intl. Workshop on Database and Expert Systems Applications (DEXA'03), September 2003, pp. 689 – 693.

[6]Apostolos Malatras, George Pavlou, et al, "Self-Configuring and Optimizing Mobile Ad Hoc Networks," Proceedings of the Second International Conference on Autonomic Computing (ICAC'05), Seattle, Washington, June 2005, pp. 372-373.

[7] H. Schmeck, "Autonomic computing - vision and challenge for system design," in Proceedings of the International Conference on Parallel Computing in Electrical Engineering (PARELEC'04), Dresden, Germany, September 2004.

[8] R. Sterritt, M. Parashar, H. Tianfield and R. Unland, "A Concise Introduction to Autonomic Computing," Journal of Advanced Engineering Informatics, Special Issue on Autonomic Computing and Automation, Elsevier Publishers, 2005, pp. 181-187.

[9] E. Bonabeau, M. Dorigo, G. Theraulaz, "Swarm Intelligence: From Natural to Artificial Systems", Oxford University Press, New York, 1999.

[10] Maniezzo V, Gambardella LM, De Luigi F, "Ant Colony Optimization, New Optimization Techniques in Engineering," by Onwubolu, GC, and BV Babu, Springer-Verlag Berlin Heidelberg, 2004, pp. 101-117.

[11] Hung Q. Ngo, Hung Le-Xuan, SungYoung Lee. A Middleware Framework for Context Acquisition in Ubiquitous Computing Systems. Second International Conference on Computer Applications (ICCA 2004), Myanmar 8th January 2004.

[12] Hung Q. Ngo, Anjum Shehzad, Saad Liaquat, Maria Riaz and Sungyoung Lee. Developing Context-Aware Ubiquitous Computing Systems with a Unified Middleware Framework. Proceedings of Embedded and Ubiquitous Computing: International Conference EUC 2004, LNCS Volume 3207, Springer Verlag 2004, pp. 672 – 681.

[13] http://uclab.khu.ac.kr

[14] Wang Jin, Shu Lei, Jinsung Cho, Young-Koo Lee, Sungyoung Lee, Yonil Zhong. "A Load-balancing and Energy-aware Clustering Algorithm in Wireless Ad-hoc Networks". The 1st International Workshop on RFID and Ubiquitous Sensor Networks, USN'2005.

[15] Xiaoling Wu, Shu Lei, Wang Jin, Jinsung Cho, Sungyoung Lee, "Energy-Efficient Deployment of Mobile Sensor Networks by PSO." Advanced Web and Network Technologies, and Applications (APWeb 2006), Harbin, China, January 16-18, 2006, pp. 373-382.