Applying Tarantula Mating based Routing Strategy on Manufacturing Network

Susmita Bandyopadhyay
Department of Business Administration
The University of Burdwan,
West Bengal, India-713104

ABSTRACT
This paper applies a routing strategy based on Tarantula Mating Behavior as proposed by the authors, Bandyopadhyay and Bhattacharya [1], on a manufacturing network. The particular behavior which is of interest is that the female Tarantula spider sometimes eats the male Tarantula just after mating for satisfying immediate need for food or for genetic purpose. This interesting behavior has been simulated with the help of a hierarchical multi-agent based framework where the master agent at the top of the hierarchy takes the final decision with the help of PROMETHEE multi-criteria decision analysis technique, based on the data for various criteria as delivered by the worker agents at the leaf level of the hierarchy. Fuzzy orientation has been added to the measurement for one of the criteria and in the calculation of PROMETHEE decision analysis technique. The strategy has been applied successfully on a manufacturing network.

Keywords
Tarantula Mating Behavior, Routing Strategy, Hierarchical Multi-Agent System; Multi-Criteria Decision Analysis Technique, Manufacturing Network

1. INTRODUCTION
In manufacturing scenario in general, the optimum point-to-point path for sending a job is decided before the journey begins. This may lead to problematic routing scenario since when a particular job reaches a particular station, there is a chance that the previously determined optimum path may not continue to be optimum anymore. Thus, it may not be proper to decide over the entire optimum path before the journey begins, since, the data, such as, the congestion data, deadlock data, shortest path data and many other conditions may change over time. Thus, instead of finding the entire point-to-point optimum path, it may be better to choose the next best node (station) to route a job towards the destination. Towards this direction, this paper proposes to apply Tarantula mating based routing strategy as proposed by Bandyopadhyay and Bhattacharya [1], which applies a hierarchical multi-agent based routing strategy and PROMETHEE multi-criteria decision analysis technique with fuzzy orientation.An agent is a computational system which is long lived, has goals, self-contained, autonomous, capable of independent decision making. The main characteristics of agents are autonomy, social ability, responsiveness, pro-activeness, adaptability, mobility, veracity, rationality. Among the benchmark multi-agent technologies, GAIA [2] is a hierarchical agent-based architecture using the concepts of object-oriented analysis and design. Wooldridge et al. [2] used some concepts from FUSION [3]. GAIA is suitable for the development of the systems like ADEPT [4], ARCHON [5]. In GAIA, every agent has a role to play and they interact with each other in a certain pre-defined way which is defined in their protocols. ROADMAP [6] is another agent-based methodology which is an extension of GAIA for complex open systems. Some of the other significant technologies include PROMETHEUS [7], TROPOS [8], PASSI [9], TAPAS [10] and so on. Some of the agent based technologies as applied in manufacturing include PROSA [11], ADACOR [12], HCBA [13] and so on.

The strategy as proposed in this paper has also used multi-criteria decision analysis technique. Multi-Criteria Decision Analysis (MCDA) techniques are basically methods to aid decision making for the cases where a decision depends on more than a single criterion. MCDA techniques can be categorized into 1) Value Measurement Models, such as, AHP (Analytic Hierarchy Process proposed by Saaty [14, 15], Simple Multi-Attribute Rating Technique (SMART) proposed by Edwards and Barron [16]; 2) Goal, Aspiration and Reference Level Models, such as, TOPSIS (Technique for Order Performance by Similarity to Ideal Solution); 3) Outranking Models, such as, ELECTRE I, II, III, IV [17-20], PROMETHEE [21-22], NAIADE [23-24]. The following section 2 describes the multi-agent strategy as proposed in this paper.

2. PROPOSED STRATEGY
This paper has used a hierarchical structure of agents (Figure 1) and PROMETHEE multi-criteria outranking method with a fuzzy orientation. The leaf level of the hierarchy contains worker agents. Each of the worker agents performs a particular task. The worker agents considered in this research study are 1) shortest path agent, 2) congestion agent, 3) deadlock agent, 4) hops agent and 5) buffer agent. The Master agent takes the final decision from top of the hierarchy. After performing the task, each of the worker agents is killed by the master agent after taking the result of the performed task from the worker agent. Thus, the hierarchical structure does not exist after all the tasks are performed by all the worker agents. The killing of the worker agents is required to save valuable computational resource and overhead for large networks. The final decision is taken by the master agent based on PROMETHEE multi-criteria decision analysis technique based on the information as provided by the worker agents. The master agent gets notification after killing each of the worker agents. The idea conveyed in this research study stems from the mating incident of a type of spider called Tarantula where the female spider eats the male one just after mating. The analogy of such interesting mating behavior with the idea in this research study is described in Figure 2.

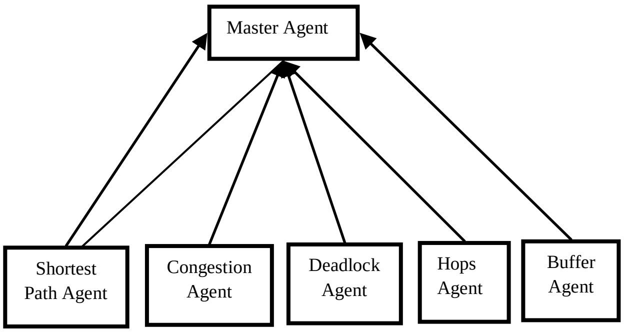

Fig 1: Hierarchy of Agents

The routing strategy considered in this research study finds the next optimum neighboring node through agent based technique, instead of finding the entire source-to-destination path. Thus the worker agents and then the master agent will start functioning whenever there will be a need to route a job to the next optimum neighboring node or whenever a new job enters the system. The master agent invokes and initiates the actions of the worker agents, just like the female spider chooses a male spider for mating. The worker agents, after performed their tasks, return the results to the master agent, just like the male spider transfers the genetic material to the female spider during mating. The master agent kills the worker agents after receiving the results from the worker agents, just like the female spider kills and eats the male spider after mating. The master agent gets the notification of the killing of the worker agents, just like the female spider takes the male spider as food. The various functions as performed by various worker agents and the master agent are described in the following subsections.

2.1 Shortest Path Agent

The shortest path agent finds the fuzzy shoretst path towards destination node from the each of the neighbors of the current node. The fuzzy shortest path is determined by the fuzzy Dijkstra's algorithm following the research study of Deng et al. [25]. The algorithm is depicted in the Figure 3. Here, *perm[]* represents Permanent node; *v[]* holds the distance to each node from current node. In this algorithm, the edge lengths are triangular fuzzy numbers from which the fuzzified edge lengths are calculated from the fuzzy numbers by using expression (1) below.

$$v_{ij} = \frac{\Pr_{ij}}{\sum_{j=1}^{C} \Pr_{ij}} \qquad (1)$$

Where, v_{ij} is the normalized value of the preference and an entry in the *i-th* row and *j-th* column in the matrix containing normalized values for the *i-th* decision maker and the *j-th* criterion; $\Pr_{ij}$ is the respective original preference value delivered by *i-th* decision maker, for the *j-th* criterion.

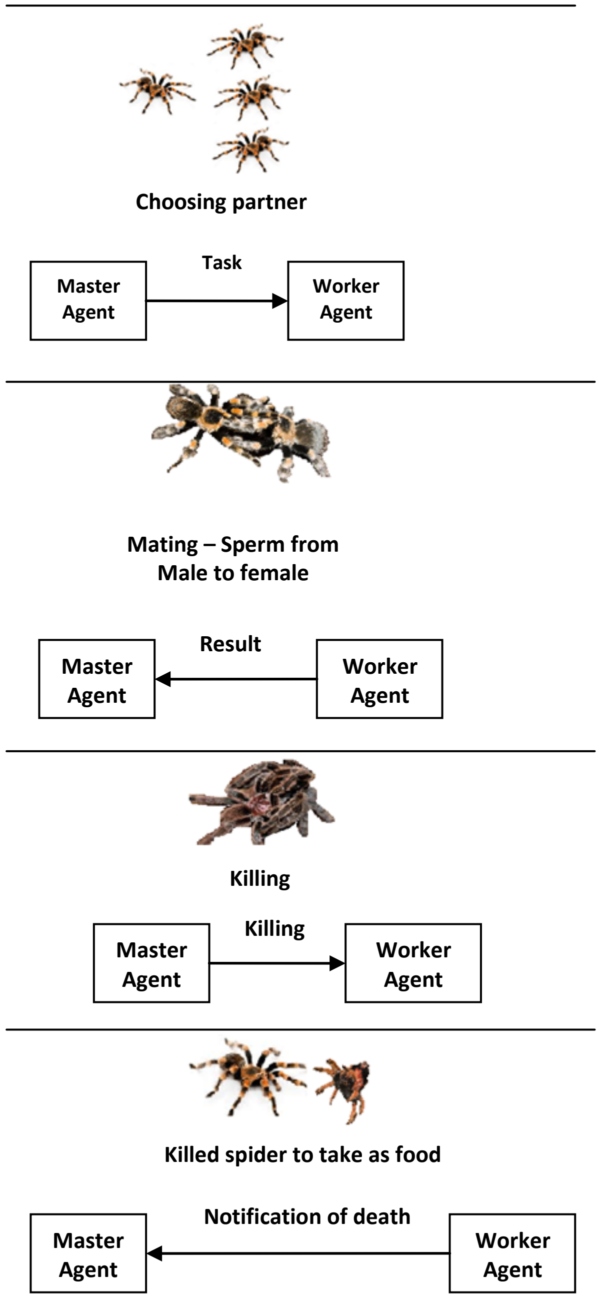

Fig 2: Analogy with Tarantula Mating Behavior

2.2 Congestion Agent

The congestion agent checks for the congestion of the edges from the current node to each of the neighboring nodes and from each of the neighboring nodes to their immediate neighbors on their shortest paths towards destination (Figure 4). Although the congestion can be represented by more than one factor, but in this research study, congestion is represented by the number of jobs travelling on a particular edge.

2.3 Deadlock Agent

The deadlock agent (Figure 5) checks whether the neighbors of the current node faces any immediate cyclic path. Let the current node is *c* and the neighbors of c are *x, y* and *z*. If the immediate neighbors of two or more neighbor of the neighbors *x, y, z* be same, then the algorithm marks those neighbors (*x* and/or *y* and/or *z*) as unsafe, otherwise they are safe. It can easily be realized that, in dynamic environment where the number of jobs on each path vary continuously, it will be less significant to find a cyclic deadlock throughout the entire network. Thus instead of finding the cyclic deadlock in the entire network, it will better to find such an immediate cycle. The algorithm endeavors to avoid cyclic path since in such path, there is more chance of facing a collision.

Input: Set of edges (E), Set of vertices (V), d[], source node (s), destination node (d)
Output: Shortest Path from s to d of length contained in v[d]

1. Initialize:
   For each edge $e \in E$
      Set Fuzzy Number $W_e = (a_e + 4 * b_e + c_e) / 6$
   End For
   Set d[s] ← 0 and d[i] ← ∞, $\forall i$, $i \neq s$
   Set permanent node p ← s
2. For each neighbor $j \in neighbor(p)$
      Set *v[j]* ← *minimum {d[j], d[p]+$C_{p,j}$}*
      Set pred[j] ← p
   End For
3. Find minimum among non-permanent nodes
   Set min ← ∞
   For each vertex $v \in V$
      If *v* is not permanent node Then
         If d[v] < min Then
            Set min ← d[v]
            Set *j'* ← *v*
         End If
      End If
   End For
4. Set next permanent node p ← j'
5. Repeat steps 2-4 until all the nodes become permanent or destination is reached

Fig 3: Fuzzy Dijkstra's Shortest Path Algorithm

```
For each neighbor i
     Record number of jobs on arc (src, i)
End For
For each neighbor i
     Find the next node n of i on SP(i)
     Record number of jobs on arc (i, n)
End For
```

Fig 4: Congestion Finding Algorithm

```
For each neighbor n1
    For each neighbor n2
        If n1 != n2 Then
            If neighbor[n1] != neighbor[n2] Then
                Safe[n1] = 1
                Safe[n2] = 1
            End If
        End If
    End For
End For
```

Fig 5: Deadlock Finding Algorithm

```
For each neighbor i
    Find shortest path SP(i) to destination from i
    Set count[i] ← 0
    For each node from n to SP(i)
        Set count[i] ← count[i] + 1
    End For
End For
```

2.4 Hops Agent

The hops agent finds the number hops or intermediate nodes on a path towards destination (Figure 6). Thus for each of the shortest path from the neighboring nodes towards destination, there is a particular number representing the number of intermediate nodes on the way. The target is to choose that particular neighbor as the better node which will have least number of hops since, the greater the number of hops, greater is the chance of facing more congestion, more deadlock, more blockage at the nodes due to loaded buffers.

2.5 Capacity Handling Agent

Each of the nodes in a network is assumed to have fixed capacity to station the number of vehicles. The capacity handling agent provides two types of data on each of the immediate neighbors – 1) the total capacity of each of the immediate neighbors and 2) the capacity spent at each of the immediate neighboring node.

2.6 Master Agent

The master agent takes the final decision based on the information provided by the worker agents (Figure 7). The shortest path agent provides the alternate path through neighbor of the current node. Thus the number of alternate paths equals the number of immediate neighbors of the current node. The congestion agent provides the congestion data in terms of the number of jobs travelling on the edge between the current node and each of the neighboring nodes and the number of jobs travelling on the edge between the immediate neighbors of the current node and the neighbors of the immediate neighbors. The deadlock agent provides the boolean values indicating whether the immediate neighbors of the current node are safe. The hops agent delivers the number of hops or intermediate nodes between the current node and the destination node on each of the alternate paths through the neighbors. Based on the above data and information, the master agent takes decision using a multi-criteria decision analysis technique known as PROMETHEE by selecting best neigbor to which the job may be routed next.

3. RESULTS AND DISCUSSION

The multi-agent and multi-criteria approach as proposed in this research study has been applied on a network example (Figure 8) with edge lengths represented by triangular fuzzy numbers. The experimentation has been performed in C# of Visual Studio .Net 2008 in a dual core PC with 4 GB memory. The worker agents have been implemented by using threads which run in parallel through thread synchronization. The relevant details are shown in Figure 9. Next, a total of 4 decision makers are assumed and they all assign their own preferences to the seven criteria. The seven criteria are – 1) Path length; 2) Number of jobs travelling on the edge from current node (node 4) to immediate neighbors (nodes 2, 6, 8). The respective edges in this example are: 4-2, 4-6, and 4-8; 3) Number of jobs travelling on the edge between the immediate neighbors (nodes 2, 6, 8) and their neighbors on their respective shortest paths; 4) deadlock status of each of the neighbors; 5) Number of intermediate nodes or hops; 6) the total capacity of buffers at each of the immediate neighbors; 7) the capacity spent for the buffers at each of the immediate neighbors.

1. Input a network with fuzzy edge values
2. Input source and destination nodes s and d respectively
3. Find neighbor N of s
4. For each neighbor $n \in N$
  Find shortest path p from n to d
  Store p in PATH
End For
5. For each neighbor $n \in N$
  Find path $p \in PATH$ from N
  Find number of jobs on edge (s, n)
  Find number of jobs on edge (n, i), i: neighbor of n on p
  Check whether there is any deadlock cycle with n
End For
6. Apply PROMETHEE to find the best neighbor n

Fig 7: Algorithm for Master Agent

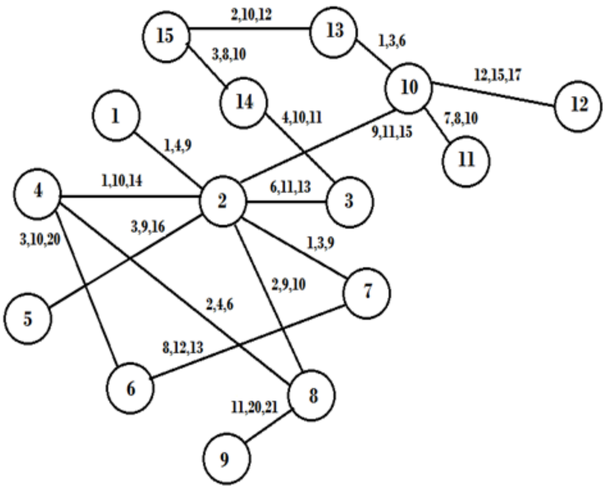

Fig 8: Network Diagram

Figure 10 shows the preferences as provided by the 4 decision makers (DM), and the fuzzy weights as calculated for the above seven criteria. For calculating the fuzzy weights, first the preferences for each DM are converted to probability values. Thus there will be 4 probability values from each DM under each criterion. Then the minimum, intermediate and maximum values are found out from each of the 4-valued set for each criterion. These three calculated numbers form the fuzzy number for each criterion and then the fuzzified value of the criterion is calculated following expression (2) shown below.

$$W_j = (\min_j + avg_j + \max_j)/3 \qquad (2)$$

Next, the values as obtained from the seven agents are shown in Figure 11. The preference index and the outranking flows are calculated following expressions (3), (4), (5) and (6). The value of Φ is calculated by expression (7) (Table 1). Since higher the value of Φ, higher is the preference of the alternative, thus the ranked alternatives in the descending order are: A1 → A3 → A2, A1 being the highest ranked alternative and thus the next best neighbor is 2.

Source node: 4;
Destination node: 7

Alternate Paths through Neighbors

Alternatives	Paths
A1	4→2→7
A2	4→6→7

Fig 9: Relevant Data on Network

Preference for 7 Criteria from 4 Decision Makers (DM)

DM	C1	C2	C3	C4	C5	C6	C7
1	3	6	4	5	7	2	1
2	5	2	6	1	4	3	7
3	1	3	6	7	2	4	5
4	7	2	6	4	1	5	3

Weights of 7 Criteria

C1	0.1547619
C2	0.1309524
C3	0.1190476
C4	0.1428571
C5	0.1190476
C6	0.1309524

Fig 10: Preference Function Values and Weights

Path length	Number of jobs from source to immediate neighbors	Number of jobs from neighbors to neighbors of immediate neighbors	Deadlock status	Number of hops	Buffer capacity
4	7	9	1	2	26
12	1	9	1	2	26
12	8	9	0	3	25

Fig 11: Data Values from Agents

Table 1. Outranking Flows

Alternatives	ϕ^+	ϕ^-	ϕ
A1	1.75	-1.75	3.5
A2	-1.285714	1.285714	-2.571428
A3	-0.4642857	0.4642857	-0.9285715

$$\pi(a,b)=\sum_{j=1}^{C}W_j P_j(a,b) \tag{3}$$

$$\pi(b,a)=\sum_{j=1}^{C}W_j P_j(b,a) \tag{4}$$

$$\phi^{+}(a)=\frac{1}{(n-1)}\sum_{x\in A}\pi(a,x) \tag{5}$$

$$\phi^{-}(a)=\frac{1}{(n-1)}\sum_{x\in A}\pi(x,a) \tag{6}$$

$$\phi(a)=\phi^{+}(a)-\phi^{-}(a) \tag{7}$$

4. CONCLUSION

This paper proposes the application of a routing strategy for manufacturing networks where, instead of establishing a point-to-point connection between source and destination nodes, a job or message is routed to the next optimal neighboring node. A hierarchical multi-criteria multi-agent based system is considered with a master agent and several worker agents for the proposed routing strategy. The number of worker agents is same as the number of criteria considered for decision making of the master agent. In this paper, a total of seven criteria have been considered to determine the next optimum node to route a particular job. These criteria are – 1) shortest path length between the each of the immediate neighbors and the final destination; 2) the number of jobs on route between the current node and each of the immediate neighbors; 3) the number of jobs between each of the immediate neighbors and the neighbors of the immediate neighbors; 4) the deadlock status involving the current node; 5) the number of intermediate nodes (hops); 6) the total capacity of buffers at each of the immediate neighbors; 7) the capacity spent at each of the immediate neighbors. The consideration for capacity is required for proper management of traffic at the junctions, especially during peak hours of a day. The master agent takes all these inputs from the worker agents and selects the best immediate neighbor using a multi-criteria outranking method known as PROMETHEE. The entire idea is based on the mating behavior of a species of spider known as Tarantula. The female Tarantula sometimes eats the male Tarantula just after mating to satisfy the intermediate need for food or for any genetic reason. Specific example has been considered to implement the proposed strategy.

5. REFERENCES

[1] Bandyopadhyay, S., Bhattacharya, R. 2015 Finding optimum neighbor for routing based on multi-criteria, multi-agent and fuzzy approach. Journal of Intelligent Manufacturing 26(1), 25-42.

[2] Wooldridge, M., Jennings, N.R., and Kinny, D. 2000 The Gaia methodology for agent-oriented analysis and design. Autonomous Agents and Multi-Agent Systems 3(3), 285–312.

[3] Coleman, D., Arnold, P., Bodoff, S. C., Dollin Gilchrist, H., Hayes, F., and Jeremaes, P. 1994 Object-Oriented Development: The FUSION Method. Prentice Hall International, Hemel Hempstead, England.

[4] Jennings, N. R., Faratin, P., Johnson, M. J., Norman, T. J., O’Brien, P., and Wiegand, M. E Wiegand 1996 Agent-based business process management. International Journal of Cooperative Information Systems 5(2-3), 105–130.

[5] Jennings, N. R. 1997 Using ARCHON to develop real-world DAI applications. IEEE Expert 11(6), 64–70.

[6] Juan, T., Pearce, A., and Sterling, L. 2002 ROADMAP: extending the Gaia methodology for complex open systems. In M. Gini, T. Ishida, C. Castelfranchi, W.L. Johnson (Eds.), Proceedings of the First International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS’02*)*, ACM Press, pp. 3–10.

[7] Padgham, L., and Winikoff, M. 2004 Developing Intelligent Agent Systems - A Practical Guide. John Wiley & Sons, ISBN 0-470-86120-7.

[8] Bresciani, P., Giorgin,i P., Giunchiglia, F., Mylopoulos, J., and Perini, A. 2004 Tropos: An agent-oriented software development methodology. Journal of Autonomous Agents and Software Development Methodologies 8, 203–236.

[9] Burrafato, P., and Cossentino, M. 2002 Designing a multi-agent solution for a bookstore with the PASSI methodology. In Proceedings of the Fourth International Bi-Conference Workshop on Agent-Oriented Information Systems (AOIS-2002), Toronto.

[10] Holmgren, J., Davidsson, P., Persson, J. A., and Ramstedt, L. 2012 TAPAS: A multi-agent-based model for simulation of transport chains. Simulation Modelling Practice and Theory 23, 1–18.

[11] Brussel, H. V., Wyns, J., Valckenaers, P., and Bongaerts, L. 1998 Reference architecture for holonic manufacturing systems: PROSA. Computers in Industry 37(3), 255-274.

[12] Leita˜o, P., Colombo, A., and Restivo, F. 2005 ADACOR: a collaborative production automation and control architecture. IEEE Intelligent Systems 20(1), 58–66.

[13] Chirn, J., and McFarlane, D. 2000 A component-based approach to the holonic control of a robot assembly cell. In Proceedings of the IEEE 17th International Conference on Robotics and Automation, ICRA, 2000.

[14] Saaty, T.L. 1980 The Analytic Hierarchy Process: Planning, Priority Setting, Resource Allocation. McGraw-Hill, New York, USA.

[15] Saaty, T.L. 1994 Fundamentals of Decision Making and Priority Theory: With the Analytic Hierarchy Process. RWS Publications, Pittsburgh, USA.

[16] Edwards, W., and Barron, F.H. 1994 SMARTS and SMARTER: improved simple methods for multiattribute utility measurement. Organizational Behavior and Human Decision Processes 3, 306–325.

[17] Roy, B. 1990 The outranking approach and the foundations of ELECTRE methods. In Bana e Costa, C.A. (Ed.), Readings in Multiple Criteria Decision Aid. Springer-Verlag, Berlin, German, pp. 155–183.

[18] Roy, B. 1991 The outranking approach and the foundation of ELECTRE methods. Theory and Decision 31, 49–73.

[19] Roy, B. 1993 Decision science or decision-aid science?. European Journal of Operational Research 2, 184–203.

[20] Roy, B. 1996 Multicriteria Methodology for Decision Aiding, Kluwer Academic Publishers, Dordrecht, Holland.

[21] Brans, J.P., and Vincke, Ph. 1985 PROMETHEE. A new family of outranking methods in MCDM. Management Science 6, 647–656.

[22] Brans, J.P., Vincke, Ph., and Mareschal, B. 1986 How to select and how to rank projects: the Promethee method. European Journal of Operational Research 2, 228–238.

[23] Munda, G. 1995 Multicriteria Evaluation in a Fuzzy Environment—Theory and Applications in Ecological Economics, Physica-Verlag, Heidelberg, Germany.

[24] Munda, G., Nijkamp, P., and Rietvald, P. 1994 Qualitative multicriteria evaluation for environmental management. Ecological Economics 10, 97–112.

[25] Deng, Y., Chen, Y., Zhang, Y., & Mahadevan, S. 2012 Fuzzy Dijkstra algorithm for shortest path problem under uncertain environment. Applied Soft Computing 12(3), 1231–1237.