World Academy of Science, Engineering and Technology
International Journal of Computer and Information Engineering
Vol:1, No:9, 2007

Optimizing Mobile Agents Migration Based on Decision Tree Learning

Yasser k. Ali, Hesham N. Elmahdy, Sanaa El Olla Hanfy Ahmed

Open Science Index, Computer and Information Engineering Vol:1, No:9, 2007 publications.waset.org/14719.pdf

***Abstract*—**Mobile agents are a powerful approach to develop distributed systems since they migrate to hosts on which they have the resources to execute individual tasks. In a dynamic environment like a peer-to-peer network, Agents have to be generated frequently and dispatched to the network. Thus they will certainly consume a certain amount of bandwidth of each link in the network if there are too many agents migration through one or several links at the same time, they will introduce too much transferring overhead to the links eventually, these links will be busy and indirectly block the network traffic, therefore, there is a need of developing routing algorithms that consider about traffic load. In this paper we seek to create cooperation between a probabilistic manner according to the quality measure of the network traffic situation and the agent's migration decision making to the next hop based on decision tree learning algorithms.

***Keywords*—**Agent Migration, Decision Tree learning, ID3 algorithm, Naive Bayes Classifier

I. Introduction

MOBILE agents are groups of executing objects that can migrate as a whole from node to node in a heterogeneous network. Mobile agent systems offer advantages such as better performance, lower usage of network bandwidth and asynchronous processing. The migration of agents comprises the transport of data, code and execution state from one node to another. The transportation of data values is common practice in traditional distributed systems. In large scale communication agent have to be generated frequently and dispatched to the network, thus they will certainly consume a certain amount of bandwidth of each link in the network. If there are too many agents' migrations through one or several links at the same time, they will introduce too much transferring overhead to the links. Eventually, these links will be busy and indirectly block the network traffic. Therefore, there is a need of developing routing algorithms that consider about traffic load. Mobile agents are autonomous, because the mobile agent is learning about the network as it progresses through it [1]. The mobile agent will visit nodes that were unknown when it was originally dispatched. At each node agent can make decisions such as invoke service, leave peer, or destroy itself based on its history of visited nodes and the current node. Information is being disseminated at every node that the mobile agent visits. Every node benefits from accepting a visiting mobile agent, because the mobile agent will have either new or more recent information about resources. Also, every mobile agent benefits from visiting a node because it will learn of either new or updated resources. If the mobile agents do not contain any new information they may be destroyed. In this paper, we propose an agent-based routing algorithm (ABRA), we investigate how agents should behave if they have to base their decisions on possibly known traffic information. In our model, we present agents migration processes based on decision tree on possibly extremist of interest of known traffic information**,** by analyzing, classifying network traffic attributes, and applying decision tree learning algorithm classified network flow data to get the best path that agent should migrate from node to node . The rest of this paper is organized as follows: We will first present background of network traffic analysis and classifying using naïve bayse technique, and overview of decision tree learning using ID3 algorithm in Section 2. In Section 3 related work will introduced. The method is elaborated in Sections 4; section 5 will deal with simulation methods and experimental results. Conclusion and future work are presented in Section 6.

Manuscript received May 9, 2007.

Yasser K. Ali is a PhD candidate, information technology department, Faculty of computers and Information, Cairo University (phone: 202-0101932430-202-24096617; e-mail: yaser_kamal@hotmail.com).

Hesham N. Elmahdy, Assoc.Prof, information technology department, Faculty of computers and Information, Cairo University (e-mail: ehesham@mailer.eun.eg).

Sanaa El Olla Hanfy Ahmed , Vice Dean for Educational and Student Affairs, Faculty of computers and Information, Cairo University (e-mail: s.hanafy@fci-cu.edu.eg).

II. Background

A. Network traffic analysis and classifying

Large quantities of network traffic flow data are generated on Peer-to-peer networks. These data contain information on the sources and destinations of individual flows encoded as IP addresses. The analysis of such data can reveal useful knowledge for traffic load, traffic cost, network situation, and network resource management. In order to perform analysis of network traffic flow data, *naïve bayes technique* [2] can be used, appropriate input data is needed , such as TCP flow , the classification process of input data requires parameterization of each object (dataset) to be classified . Using these parameters, classifier allocates an object to a class.

Open Science Index, Computer and Information Engineering Vol:1, No:9, 2007 publications.waset.org/14719.pdf

B. Naive Bayes Classifier Overview

The Naive Bayes Classifier technique is based on the so-called Bayesian theorem [3] and is particularly suited when the dimensionality of the inputs is high. Despite its simplicity, Naive Bayes can often outperform more sophisticated classification methods.

To demonstrate the concept of Naïve Bayes Classification for network traffic attributes, consider the example displayed in the figure [1]. As indicated, the instance raw data of response time can be classified as either normal response time represented as a green color or low response time represented as red color. Our task is to classify new cases of response time value as they arrive, i.e., decide to which class label they belong, based on the currently exiting data. Since there are twice as many green color values as red color, it is reasonable to believe that a new case (which hasn't been observed yet) is twice as likely to have membership green color rather than red color. In the Bayesian analysis, this belief is known as the prior probability. Prior probabilities are based on previous experience, in this case the percentage of green color and red color objects, and often used to predict outcomes before they actually happen.

Thus, we can write:

Prior Probability for Green Color (Normal Response Time)

$$\propto \frac{NumberofGreenColor}{TotalNumberofObjects}$$

Prior Probability for Red Color (Low Response Time)

$$\propto \frac{NumberofRedColor}{TotalNumberofObjects}$$

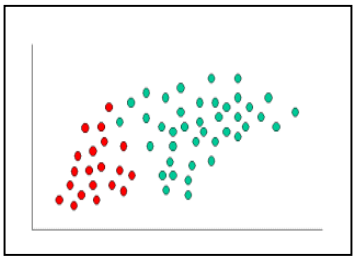


Fig.1 Response Time Instance Objects

C. Decision tree learning using ID3

In decision theory and decision analysis, a decision tree is a graph or model of decisions and their possible consequences, including chance event outcomes, resource costs, and utility. It can be used to create a plan to reach a goal. Decision trees are constructed in order to help with making decisions [4]. A decision tree is a special form of tree structure. Another use of trees is as a descriptive means for calculating conditional probabilities.

In agent migration, on the other hand, a decision tree is a predictive model; that is, a mapping of observations about a network situations to conclusions about the agent's migration path. More descriptive names for such tree models are classification tree or reduction tree. In these tree structures, leaves represent classifications and branches represent classes of network traffic attributes that lead to those classifications.

ID3

An early technique by the influential Ross Quinlan [5] that influenced a large part of the research on Decision Trees is useful to look at in order to understand basic decision tree construction.

Entropy

A measure used from Information Theory [6] in the ID3 algorithm and many others used in decision tree construction is that of Entropy. Informally, the entropy of a dataset can be considered to be how disordered it is. It has been shown that entropy is related to information, in the sense that the higher the entropy, or uncertainty, of some data, then the more information is required in order to completely describe that data. In building a decision tree, we aim to decrease the entropy of the dataset until we reach leaf nodes at which point the subset that we are left with is pure, or has zero entropy and represents instances all of one class (all instances have the same value for the target attribute).

We measure the entropy of a dataset, ***H***, with respect to one attribute, in this case the target attribute, with the following calculation:

$$H = \sum_{i=1}^{c} p_i \log_2 p_i$$

where P_i is the proportion of instances in the dataset that take the i^{th} value of the target attribute .This probability measures give us an indication of how uncertain we are about the data. And we use a ***log2*** measure as this represents how many bits we would need to use in order to specify what the class (value of the target attribute) is of a random instance.

III. RELATED WORK

Ichiro Satoh [7] presents a framework for building network protocols for migrating mobile agents over the Internet. The framework allows network protocols for agent migration to be naturally implemented within mobile agents and then dynamically deployed at network hosts by migrating the agents that perform the protocols. It is built on a hierarchical mobile agent system, called MobileSpaces, and several protocols for agent migration are designed and implemented based on standard protocols in the Internet, for example agent migration protocols through plain TCP, HTTP, SMTP or SSL, and application-specific routing protocols for efficiently migrating agents among multiple hosts.

Torsten Illmann, Tilman Krueger, Frank Kargl, Michael Weber [8] examined a migration technique of mobile agents in Java technology; they classified different migration styles and presented possible solutions and related work. They proposed classification distinguishes between code migration, execution

Open Science Index, Computer and Information Engineering Vol:1, No:9, 2007 publications.waset.org/14719.pdf

migration and data migration. The classification defines a partial order to compare different migration approaches. For realizing strong migration in Java, two solutions are proposed. On the one hand, a pre-processor adds all the necessary information for migration to the source code before compilation time. On the other hand, A JNI-based plug-in for any virtual machine provides mechanisms to captures the agent's execution state. The restoration of the execution state is done by the plug-in in combination with a byte code modifier which slightly changes the byte code of the agent.

Xiliang Zhong, Cheng-Zhong Xu, and Haiying Shen [9] presented a reliable connection migration mechanism that provides exactly-once delivery for all transmitted data during agent migration. It integrates with an agent-based access control mechanism that controls the access to network ports. To avoid frequent agent authentication and permission checking due to agent migration, a secret session key is associated with each connection. They present the design and implementation of the mechanism, named NapletSocket in Naplet [10] mobile agent system. It is a pure middleware implementation, requiring no modification of Java virtual machines.

Yingyue Xu, Hairong Qi [11] present two itinerary planning algorithms with the goal of maximising the information extracted while keeping resource usage to a minimum. The ISMAP determines the itinerary before dispatching the mobile agent while the IDMAP algorithm selects the route on the fly. They design three metrics (energy consumption, network lifetime, and the number of hops) and they used simulation tools to quantitatively measure the performance of different itinerary planning algorithms. Simulation results show considerable improvement over the ISMAP using the IDMAP itinerary algorithm.

Tino Schlegel, Peter Braun, Ryszard Kowalczyk [12], they suggested to apply the programming paradigm of mobile agents to reduce the network overhead by allowing agents to meet at the same network node before commencing communication. A remote communication between two agents could then be replaced by one or two agent migrations, followed by local communication. Since only in trivial cases it is possible to decide at design time whether remote communication or agent migration with subsequent local communication would perform better, this decision has to be made at run-time based on environmental parameters and agents' past experience. Also they presented an adaptive approach, which is inspired by a solution of the El Farol problem [13]. Every agent forecasts the network load of the next communication step and applies a simple mathematical model to decide between the two alternatives at run-time.

IV. METHOD

A peer-to-peer system is a distributed system whose component nodes participate in similar roles, and are therefore peers to each other. Peer-to-peer can be viewed as decentralized network architecture.

Our algorithm focuses on how the mobile agents will behave while they roam around the network and can be sketched as follows:

1. A user accesses the system and places service requests specifying some criteria.
2. Mobile agents are periodically generated by its peer group; they room inside its peer group, collect and disseminate information
3. The data state contains the information carried by the MA during migrations obtained when each peer periodically sends a mobile agent, by broadcasting replicas of it to each neighbor node. and giving a path and an extremist of interest, the path identifies a subtree, e.g. path "***abcd***" , path "***abcf***" , or path "***abce***" as seen in Figure 2. The extremity of interest obtained by analysis , classify the network flow data attribute for each subtree .

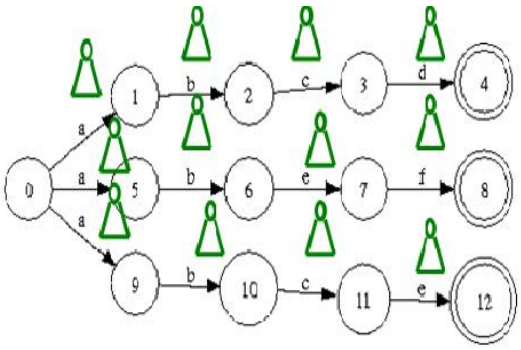


Fig. 2 Mobile agent created by its peer

4. The main approach that is used to classify network traffic flow flow data is the Naïve Bayesian technique. Consider a collection of flows x = (x1,……, xn), where each flow xi is described by m discriminators{ d1(i),…….., dm(i)}that can take either numeric or discrete values. In the context of the network traffic c, dj(i) is a discriminate of flow xi, for example it may represent the mean interval time of packets in the flow xi. In this paper, flows xi belongs to exactly one of the mutually-exclusive classes described in Section 2. The supervised Bayesian classification problem deals with building a statistical model that describes each class based on some training data, and where each new flow y receives a probability of getting classified into a particular class according to the Bayes rule below,

$$p(cj \mid y) = \frac{p(cj)\, f(y \mid cj)}{\sum_{cj} p(cj)\, f(y \mid cj)}$$

where ***p(cj)*** denotes the probability of obtaining class cj independently of the observed data, ***f (y | cj)*** is the distribution function (or the probability of y given ***cj***) and the denominator acts as a normalising constant ,

Open Science Index, Computer and Information Engineering Vol:1, No:9, 2007 publications.waset.org/14719.pdf

Table 1 demonstrates the classification accuracy of sample

TABLE I
NETWORK TRAFFIC ATTRIBUTE CLASSES

Path	Round Trip Times (Millisecond)	Throughput	Packet Loss
A-B-C-D	Minimum = 0ms	Normal	One way loss
A-B-C-D	Average = 2ms	Low	One way loss
A-B-C-D	Average = 2ms	Low	One way loss
A-B-C-D	Maximum = 10ms	Low	RT loss
A-B-C-F	Average = 2ms	Normal	One way loss
A-B-C-F	Average = 2ms	Normal	One way loss

data represent a classes of network attributes such as; ***Round Trip Times, Throughput*** and ***Packet Loss***, they are very well classified.

5. After classifying network traffic flow data, decision tree learning using ID3 algorithm, will applied to define the interest of extremist, that illustrate the best path, which agent should migrate from node to node.

First, we calculate the entropy for each class:

Round Trip Time:

$$H=\frac{1}{6}(-\frac{1}{1}\log\frac{1}{1})+\frac{4}{6}(-\frac{2}{4}\log\frac{2}{4}-\frac{2}{4}\log\frac{2}{4})+\frac{1}{6}(-\frac{1}{1}\log\frac{1}{1})=0.462$$

Throughput:

$$H=\frac{3}{6}((-\frac{2}{3}\log\frac{2}{3})-(\frac{1}{3}\log\frac{1}{3})+\frac{3}{6}(-\frac{3}{3}\log\frac{3}{3}))=0.318$$

Packet Loss:

$$H=\frac{5}{6}((-\frac{3}{5}\log\frac{3}{5})-(\frac{2}{5}\log\frac{2}{5})+\frac{1}{6}(-\frac{1}{1}\log\frac{1}{1})=0.56$$

Thus we select the class throughput as the first decision node since it is associated with the minimum entropy. This node has two branches normal and low. Under the branch low, only path **A-B-C-D** fall, and hence no further discrimination is needed. Under the branch normal, we need another attribute to make further distinctions. So we calculate the entropy for the other two attributes under this branch:

Round Trip Time

$$H=\frac{1}{3}(-\frac{1}{1}\log\frac{1}{1}+\frac{2}{3}(-\frac{2}{2}\log\frac{2}{2}))=0$$

Throughput:

$$H=\frac{3}{3}(-\frac{2}{3}\log\frac{2}{3}-\frac{1}{3}\log\frac{1}{3})=0.636$$

Thus the decision tree path that explains the extremist of interest of network traffic flow data class is configured as illustrated in figure [3]. Figure [4] illustrate the best path that agent should migrate.

6. Each mobile agent while traveling collects, and carry path information, once an agent reaches a peer , the peer tells it whether the request can fulfilled in its peer group, if so the mobile agent goes back to the source peer along the same route it has explored , updates the routing table along its route. Thus each peer in the network maintains current routing information for reaching nodes outside its neighborhood.

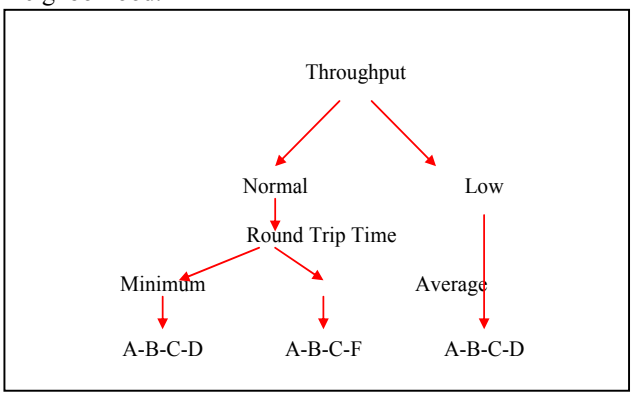


Fig. 3 Extremist of Interest

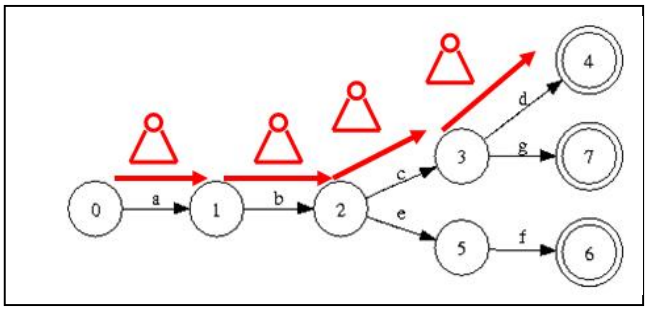


Fig. 4 Agent Migration Path

7. Algorithm 1 presents the agent based routing algorithm **(ABRA)** in detail. First, it checks if the mobile agent was correctly routed (line 3 to 7). If not, a bad request is sent. Lines 10 to 23 route the mobile agent to a random peer having path information. An index parameter is used to indicate the number of common data items. If the mobile agent can not be forwarded, a peer not found message is sent and the algorithm stops. Lines 25 to 41 route the agent to most extremity of a subtree. When a peer inside a subtree receives a mobile agent and can not forward it, it replies to the initiator peer (line 45) as the peer it was looking for and the algorithm stops.

Table 2 gives important notations of the agent-based routing algorithm, and it can be sketched as seen in algorithm [1].

TABLE II
INTRODUCES COMMON NOTATIONS OF
THE AGENT BASED MIGRATION ALGORITHM.

Notation	Description
P_c	A current peer receiving a mobile agent
P_i	The initial peer generates a mobile agent
$H(p)$	Expected information content of probability distribution (entropy) used to define the extremist of interest.
AVG_ENTROPY	Return $-\sum_v p(v)\log_2(p(v))$, where $0.\log_2(0)=0$
B	Set containing all, paths between the higher and the lower bound of information contentment
CommonPerfix	Returning the number of common
$\pi(p)$	Returns the path of peer P
ξ *(path)*	Returns a set of all classes managed by a peer with path ***path***
Index	Number of common data

```
//Part 1 , input data , classification attribute
Return decision tree
For each attribute compute AVG_ENTROPY is minimal
Part 2
1 compath =commonPrefix(localPath, path);
//3 Check if this Mobile Agent was correctly routed
4 if compath < mA.getIndex() then
5       sendBadRequest ;
7        Return ;
8  endif
// part 3
10 if compath < localPath.length AND compath < path.length then
12       setIndex(mAgent, compath);
13       peers =randomize(getPeersAtLevel( compath) );
14       for peer = peers.getF irst(); peer ≠ ⊥ ; peer = peers.getNext() do
15       if NOT online( peer) then Continue ;
16       correct = Send (peer, mAgent);
17       if correct then return ;
18       else Continue ;
19      endfor
20     reply = new PeerNotFound();
21     Send(mAgen.getInitialPeer(), reply);
23     Return ;
25 else if compath < localPath.length AND compath == path.length then
// Part 3
26       for level = mAgent.getIndex(); level < maxLevel; level++ do
27       peers =randomize(getPeersAtLevel( compath) );
28      for peer = peers.getFirst();peer ≠ ⊥; peer = peers.getNext() do
29       if NOT online( peer) then Continue ;
30       else if (isOnTheLeftOf( peer, localpeer) AND mode == LEFT)
OR (isOnTheRightOf ( peer, localpeer) AND mode == RIGHT) then
32      setIndex(mAgen, level);
33       correct = Send (peer, mAgen);
34      if correct then return ;
35      else Continue ;
36     endif
37     Break ;
38    endfor
39 endfor
40 endif
// The local peer should be the one we are looking for
41 reply = new Reply(localPeer);
42 Send(mAgen.getInitialPeer(), reply);
```

Algorithm 1. Agent Based Routing Algorithm

V. Simulation Structure

In order to validate our algorithm, we present some simulation results from omnet++ discrete event simulation systems [14], we compared it with di caro and dorigo's antnet [15] that use similar forward agents but with more complex back-propagating agents and a different routing table updates procedure. Their antnet system has been shown to outperform OSPF [16]. in this section we use a simulated network that is composed of two cases , the first one is when no of nodes are greater then no of agents , and the case two is when no of agents are greater than no of nodes . Table [3] summaries network parameters used to evaluate a simulation method

TABLE III
NETWORK PARAMETERS

Network Parameters	Values of case 1	Values of case 2
No of nodes	3-20 nodes	14 nodes
No of Agents	5 agents	5-20 agents
No of links	8-32	21
Link bandwidth	2 Mbit/sec	2 Mbit/sec
Propagation Delay	From 2 to 20 milliseconds	From 2 to 20 milliseconds
Packet data size	512 byte	512 byte
Agent Size	32 byte	32 byte

To evaluate the result of simulation we used two measurements: ***average delay*** on the network and ***throughput***. The average delay experienced by all the packets that arrived during a given interval of time gave another view of the general status of the network. The throughput is the number of packets that went through the network during one unit of time. Figure [4] compares the average packet delay and throughput for agent based routing algorithm (ABRA) and OSFP. In figure [4.a] it is clearly show that ABRA algorithm is much better than OSFP when the number of nodes is increased the time delay in ABRA is lower than OSFP. Also In figure [4.b] throughput in ABRA consumed little time than OSFP. Case two represented in figure [4.c,d] time delay and throughput in ABRA consume much time than OPSF when numbers of agent s and nodes are small , when numbers of agents and nodes increased ABRA gives better performance than OSFP.

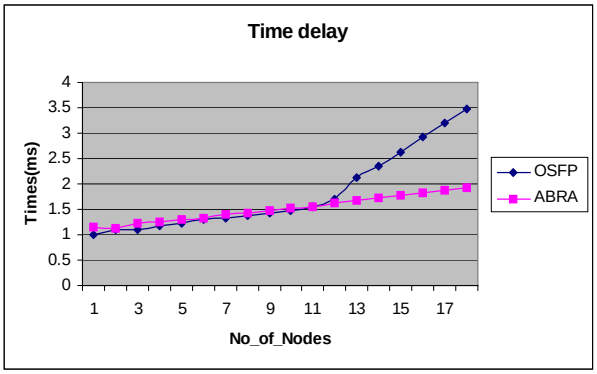


Fig. 4 a time delay at no of nodes from 3 to 20 and 5 agents

Open Science Index, Computer and Information Engineering Vol:1, No:9, 2007 publications.waset.org/14719.pdf

Open Science Index, Computer and Information Engineering Vol:1, No:9, 2007 publications.waset.org/14719.pdf

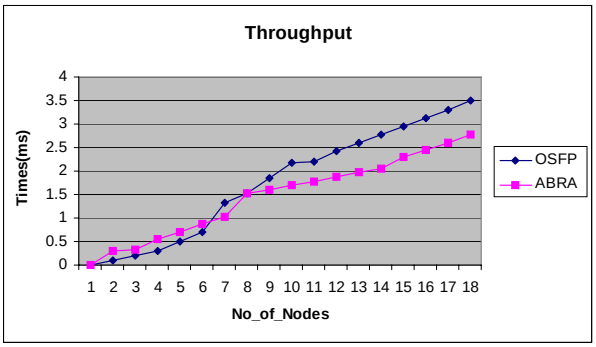
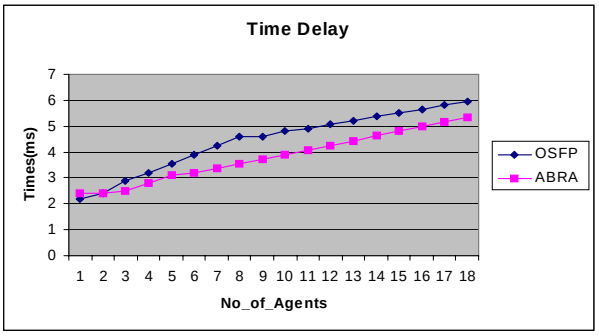


Fig. 4 b throughput at no of nodes from 3 to 20 and 5 agents



ig. 4 c time delay at no of nodes = 5 and no of agents from 3
o 20

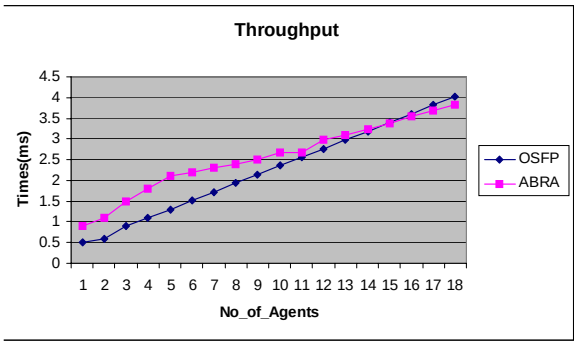


Fig 4.d throughput at no of nodes = 5 and no of agents from 3 to 20

VI. Conclusion and Future Work

In this paper we have introduced a mobile agent migration technique based on decision tree learning for P2P networks. The proposed solution in this work has the objective of an agent based routing algorithm (ABRA). We defined a most extremist path (subtree) for each link based decision tree learning algorithm, it was applied after analyze and classify network traffic flow data for each peer by using naïve baye technique, and this approach gives the best path , that agent should migrate from node to node . Figure 5 is a block diagram which illustrates the main processes of our technique.

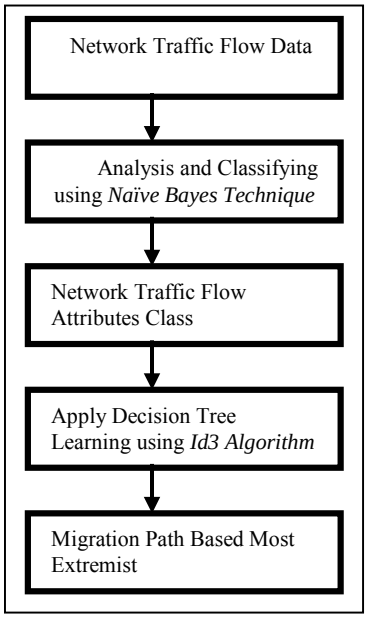


Fig. 5 Main Process of ABRA

Future work will focuses on enhanced mobile agent performance in distributed system environment, while web services and peer-to-peer are considered the ideal distributed system technology [17]; we will study interaction of mobile agent with web services in semantic approaches [18]. Based on information theory in capacity channel we propose a layered communication protocol in which the agents gradually build towards a semantically integrated system by establishing minimal and effective shared ontologies[19].

References

[1] Wenyu Qu; Kitsuregawa, M.; Keqiu Li; Hong Shen, "An Execution Prototype of Mobile Agent-Based Peer-to-Peer Systems" presented at Computer and Computational Sciences, 2006. IMSCCS , 20-24 April 2006 Page(s): 330 - 338

[2] Liu, J.N.K.; Li, B.N.L.; Dillon, T.S.," An improved naive Bayesian classifier technique coupled with a novel input solution method" presented at Applications and Reviews, IEEE Transactions on Volume 31, Issue 2, May 2001 Page(s):249 - 256

[3] José M. Bernardo, Adrian F. M. Smith "Bayesian Theory" presented by (Wiley Series in Probability and Statistics) copyright 2000, chapter 5 page(s) 241-263.

[4] Sven Ove Hansson," Decision Theory A Brief Introduction" presented at Department of Philosophy and the History of Technology Royal Institute of Technology (KTH) Stockholm, 23 August 2005

[5] Padraic G. Neville " Decision Trees for Predictive Modeling" presented by SAS Institute Inc. 4 August 1999, http://bus.utk.edu/stat/datamining/Decision%20Trees%20for%20Predictive%20Modeling%20(Neville).pdf

[6] G J Chaitin," Algorithmic Information Theory" presented at IBM, P O Box 218 Yorktown Heights, NY 10598, Third Printing, April 2, 2003

[7] Ichiro Satoh "Network Processing of Mobile Agents, by Mobile Agents, for Mobile Agents" presented at Mobile agents for telecommunication

applications. International workshop No3, Montreal PQ, Canada, 14 august 2001.

[8] Torsten Illmann, Tilman Krueger, Frank Kargl, Michael Weber " Transparent Migration of Mobile Agents Using the Java Platform Debugger Architecture, Mobile Agents" presented at 5th International Conference, MA 2001 Atlanta, GA, USA, December 2-4, 2001. Proceedings

[9] Xiliang Zhong, Cheng-Zhong Xu, and Haiying Shen," A Reliable and Secure Connection Migration Mechanism for Mobile Agents" presented at the 24th International Conference on Distributed Computing Systems WorkshopsW7: EC (ICDCSW'04) - Volume 7 Pages: 548 - 553 , 2004, ISBN:0-7695-2087-1 .

[10] ttp://www.ece.eng.wayne.edu/~czxu/software/tutorial/napletExamples/napletSocket.html

[11] Yingyue Xu, Hairong Qi " Dynamic mobile agent migration in Wireless Sensor Networks" presented at International Journal of Ad Hoc and Ubiquitous Computing 2007 - Vol. 2, No.1/2 pp. 73 - 82

[12] Tino Schlegel, Peter Braun, Ryszard Kowalczyk,(2006) Towards Autonomous Mobile Agents with Emergent Migration Behaviour, AAMAS'06 May 8–12 2006, Hakodate, Hokkaido, Japan.

[13] Ann M. Bell, William A. Sethares" The El Farol Problem and the Internet: Congestion and Coordination Failure" http://fmwww.bc.edu/cef99/papers/Bell.Sethares.pdf

[14] http://www.omnetpp.org/

[15] Yong Lu; Guangzhou Zhao; Fanjun Su "Adaptive ant-based dynamic routing algorithm" presented at Intelligent Control and Automation, 2004. WCICA 2004. Volume 3, 15-19 June 2004 Page(s): 2694 - 2697.

[16] Uwe R¨ottgermann "Decentralized Throughput Optimization in Industrial" presented at Institute of Information and technology, Munchen University, Germany. http://deposit.ddb.de/cgibin/dokserv?idn=978930878&dok_var=d1&dok_ext=pdf&filename=978930878.pdf

[17] Gehlen, G. Pham, L. "Mobile Web services for peer-to-peer applications" presented at Consumer Communications and Networking Conference, 2005. CCNC. 2005 Second IEEE Publication Date: 3-6 Jan. 2005 On page(s): 427- 433

[18] Jiangang Ma, Jinli Cao, Yanchun Zhang ," A Probabilistic Semantic Approach for Discovering Web Services" Presented at the international world wide web conference, Banff, Alberta, Canada. May 8–12, 2007.ACM 978-1-59593-654-7/07/0005.

[19] Jurriaan van Diggelen, RobbertJan,Beun, Frank Dignum, Rogier M. van Eijk, JohnJules Meyer " ANEMONE: An Effective Minimal Ontology Negotiation Environment" presented at International Conference on Autonomous Agents Proceedings of the fifth international joint conference on Autonomous agents and multiagent systems, Hakodate, Japan May 15-16- 2006,Pages: 899 - 906 ,2006

Open Science Index, Computer and Information Engineering Vol:1, No:9, 2007 publications.waset.org/14719.pdf