Resource Sharing in Distributed Environment using Multi-agent Technology

L. D. S. B. Weerasinghe
Department of Computer Science
General Sir John Kotelawala Defence University
Sri Lanka

R. P. S. Kathriarachchi
Department of Computer Science
General Sir John Kotelawala Defence University
Sri Lanka

B. Hettige
Department of Computer Science
General Sir John Kotelawala Defence University
Sri Lanka

A. S. Karunananda
Department of Computational Mathematics
University of Moratuwa, Moratuwa
Sri Lanka

ABSTRACT
Resource sharing is very important in the world Due to limited resources. People tend to use different applications for similar purposes within the network environment making the high traffic and duplicating resource. The complexity and the dynamic behaviour of computer network do not leave a clue to predict what happen next. The multi-agent technology has proven potential results in improving efficiency and accuracy in dynamic and distributed environments. Among other features, a multi-agent technology can produce solutions that are globally accepted to the agents through communication, negotiation, and coordination among the agents. This research presents the method for reducing resource wastage and sharing resources efficiently using multi-agent technology. Network users will download the same file again and again unintentionally, bringing network performance dramatically down. With the concept of dynamic scheduling and load balancing, implement a system to share resources within a network using Multi-agent technology. The solution is developed by the MaSMT, a Java-based framework, with one manager agent and four ordinary agents, namely, file send agent, file receives agent, download agent and load balancing and dynamic scheduling agent. Using this system task has been allocated to distributed agents within a dynamic network for sharing resources. The system is successfully tested in real environments and will help to reduce the resource wastage on network environments.

General Terms
Multi-agent Systems, File Sharing, JAVA, Socket

Keywords
Multi-Agent Systems, MaSMT, File-Sharing, Dynamic scheduling, Load balancing.

1. INTRODUCTION
Nowadays world increasing experiences of communication devices and therefore the growing bandwidth, file sharing applications have remarkably become of those are terribly celebrated for knowledge transfer over the Wide area Network (WAN) and local area Network (LAN). Communication and file sharing application's technology permit the sharing of computer resources such as files by a direct interchange between end-users computers or devices. Peer to Peer networking means files are not kept on a central server. Instead, client software (such as the popular Kazaa [1] , Limewire [2]) works as a server for transfer files on an individual's computer. This allows each computer with the software to performance as a mini-server from which other software holding users can download and share files. Peer to peer communication have several characteristics such as direct connections between network clients, each client is considered as an equal to all other clients, clients can share processing, applications, and content. There are several benefits of using peer to peer network in a real environment. They are reducing the cost of sharing, resource aggregation, and interoperability, autonomy, and privacy.

Nowadays networks resources are very importance in the world. Without user's knowledge, they waste these resources when it using. Most of the people use pen drives to share their data with others. One person downloads file from the internet and puts in into pen drive then share with other. It very risky because the virus can easily spread through the pen drivers. Otherwise, all the people try to download that file through the internet. Because of that network become slow due to the network traffic. It will affect all the users on the network and due to this reason wastage user's time. Finally, it will affect to reduce the total work time for the users. These two methods take so many time to get the file to your computer. Another most popular method is shared folders and this method is commonly used in industry and some organizations [14]. This method also has a lot of problems. People download a large number of files in one computer when all other computers are free on the same network. These people hold huge workload in their devices because of that it makes time wastage. The networking task becomes more complex due to dynamic behavior of the network. Network status will be change time to time and network application also changes frequently. Therefore another problem is difficult to schedule task in the network. Because of behavior is very dynamically changed.

This paper reports the design and implementation of the Multi-agent based system which is design for peer to peer network in local area network application, especially for resource sharing a purpose. The agent design architecture, class module, and agent communication diagrams are also given in the paper.

The rest of the paper is structured as follows. The section 2 gives some related works. Then section 3 gives a note on

multi agent systems technology including the MaSMT framework. Section 4 presents design and implementation of the system. The section 5 reports experimental result of the research and finally section 6 gives conclusions and further works of the research.

2. RELATED WORKS

Koen and others implemented a home automation software that dynamically schedules network bandwidth for various types of multimedia streams with various constraints toward quality and latency [8]. The dynamic scheduling problems can be overcome using agent systems by using communications. Nowadays these systems are configured using statically methods and this method has so many limitations. The video stream is configured to a specific format therefor network no longer overloaded. When so many application comes to the environment that configuration is no longer valid. Therefore network streams must be configured dynamically when environment change. This tool is using multi-agent technology with Contract Net Protocol (CNP) [9]. Agents can communicate their objectives and try to collaborate with each other in order to accomplish a better usage of shared resources. Dynamically allocation bandwidth to network streams is a very big problem in normal applications. The various types of streams have various constraints including latency and quality. Streams which need high quality can use buffering, whereas low latency streams have to be scheduled immediately. For the home automation Researchers identified a number of network stream types such as multimedia streams, videophone streams, command signals, logging services and alarm signals. Researchers found that these streams need a different amount of quality and latency for their streams. As an example multimedia streams need high quality and high latency, videophone streams need low latency, and low quality and all other streams need ultra-low latency. In researcher's tool, each agent works according to a utility function that denotes the satisfaction level of the stream application. The tool has a video agent, it will be answerable for displaying real-time video streams and uses a Fermi-utility function to get judgments on whether or not the agent can free some bandwidth when the system network agent makes a request. The network agents wait for the suggestions of the active video agents about how much bandwidth agent are willing to turn on. Based on potential and priority losses the network agent will accept suggestions till the new service can be provided and a reallocation is prepared. Researchers simulate a home situation were concurrently numerous activities can be active that need bandwidth. Primarily bandwidth is appropriately present and then progressively the network will get more and more overloaded. The tool shows all competing agents and the progress of the bandwidth that is available to that agents. So finally Multi-agent systems can successful handle dynamic distributed scheduling and have been used for developing a wide range of dynamic scheduling applications.

"Agent Technology in Load Balancing for Network Applications" paper presents by M. Grivas and S. J. Turner [9]. This paper includes classification which is used to review the models and implementations that using software agents to accomplish load balancing in the Web applications. Distributed computing presented new problems and a new field of research for load balancing. Many types of research proposed different techniques to load balancing in the distributed system but research use agent base solution for load balancing because of agent have new characteristics for solving the problem in this environment. Using tree of processors as the central balancer and using a single processor are the simple methods of load balancing in small simple systems. Clustering is the new trend in distributed computing. Main disadvantages of these techniques are the base of algorithms and all depend on that algorithm. These researchers classify distributed workload tasks into communicative vs non-communicative, mobile vs static, centralized vs distributed, sender-initiated vs receiver-initiated, global techniques vs grouping, predefined groups vs reformatting groups, approximate vs heuristic and instant information vs buffered information. Under non-communicative architectures, researchers describe Ant-like agents and Adaptive load balancing methods. Under communicative architecture, they give points to NetSolve [11] system architecture load balancing and how PageSpace [12] system balance their load in the network. In Challenger multi-agent system researcher how to balance their workload with client and server architecture using agent communication. In web applications links are changed, servers change and improve or deteriorate applications interfaces so the non-adaptive scheduling cannot be effective. The load balancing scheme shouldn't exclusively consider the data received, since it may not receive it. This paper gives research information about the existing agent solutions offers novel abilities. Agents can alleviate or even eliminate most of the problems very quickly and easily.

Mobile environments pose additional challenges on P2P networks due to the heterogeneity of nodes, inherently limited resources, dynamic context and wireless network characteristics. RobP2P is a robust architecture use to construct mobile Peer to peer networks and resourcefully preserve the network state [13]. This paper proposes RobP2P with lay the foundation for efficient and scalable P2P overlay networks and develop a reliable P2P infrastructure that enables efficient resource sharing and service provisioning in mobile environments without infrastructure support. Much of the study which has explored the challenge of super-peer selection and donated with many algorithms and techniques were considered either for static networks. The strategy of RobP2P contains three objectives such as advance a robust and effective super-peer selection protocol, decrease the overhead traffic of network topology maintenance and finally growth the consistency and immovability of the network structure over the allowing peers to flexible alternating their role. All peers estimate their profile index by utility function and join to the super-peer Selection Algorithm. When super-peers are selected, all announcements and queries inside groups are sent to corresponding super-peers. That peer gathers index group information together with active peers, presented resources, and obtain services with the aim of managing the group communications and resolve queries addressed to the group. A super-peer should be skillful to increase the general performance of P2P networks, else it might convert to a bottleneck. Researchers conducted numerous experimentations to evaluate the performance of RobP2P, directing on its characteristics. They are overhead of super-peer choice, how to maintains a steady state while decreasing the amount of needless super-peer choice, which will test the quality of our utility functions and how to handles the churn of mobile networks while maintaining the system reliability. Finally, research showed that RobP2P is robust mobile peer to peer architecture that allows efficient resource sharing.

Chongjie Zhang, Victor Lesser and Prashant Shenoy propose a multi-agent learning algorithm and use it for improving online resource allocation in cluster networks [14]. To

develop a shared computing structures, the traditional model is to establish a set of shared clusters into a network and allow resource sharing through shared clusters. The resource sharing consequence is now spread to each shared cluster. Individual cluster still uses a cluster-wide technique for managing its local resources. Task allocation requirements vary across clusters, a cluster may need to dynamically choice what tasks is allocated locally and wherever to direct unallocated tasks to compassionately improve the over-all effectiveness of the whole system. To simplify the learning algorithm in proposed system, researchers divide each agent's decisions into 2 connected learning problems namely task routing problem and local allocation problem. This approach works effectively and simulation shows that are not totally realistic from the perspective of actual networks of shared clusters, researchers have confidence in this research has shown multi-agent corroboration learning is a favorable approach to resource controlling across shared clusters.

3. MULTI-AGENT TECHNOLOGY

Multi-agent technology becomes a new trend in artificial intelligence because of the power of communicating for solving problems like humans. A single agent cannot provide best solutions by using individual capacity, but by communicating with each agent can provide more accurate solutions for the complex problem. The problem becomes more complex because of dynamic behavior, interconnected with each other, distributed and uncertain. Nowadays there is so many well-established frameworks and toolkit for developing multi-agent systems such as JADE [3] (Java Agent Development Framework), JaCaMo [4], SeSam [5] (Shell for Simulated Agent Systems), Madkit [6](Multi-Agent Development Kit), MaSMT [7] and etc. Among these frameworks, JADE is the well famous framework for agent development and it's developed for the Java language. MaSMT is lite weight framework for the agent development and its provide GUI interface for debugging agent simulation and it's totally developed for Java based agent applications. JaCaMo framework supports agent development for the android platform to mobile agents. There are so many advantages of using agent framework for the agent development such as save developer's time and developer's work capacity. Those all frameworks are developed for general purposes, not for special purposes like file sharing, load balancing, and dynamic scheduling.

4. DESIGN AND IMPLEMENTATION

This section presents design and implementation of the proposed solution in terms of MaSMT framework, Agent architecture, Task scheduling, Load balancing and Task allocation.

4.1 MaSMT Framework

Multi Agent System for Machine Translation (MaSMT) is a java based agent development framework with two kinds of agents namely manager agent and ordinary agent [7]. Manager agents have responsible for controlling ordinary agents. The framework developed under Fundamentals of Intelligent Physical Agent (FIPA) stands and message passing is done by according to the Agent Communication Language (ACL). MaSMT agents follow an role-group model of the MaDKit [19] . According to the abstract agent model, every agent has agent id, agent group and agent rule. MaSMT agent has a life cycle containing main parts of active, live and end life events. In active session, all the agents become activated and initializing whatever they want to do in other sessions. In live session, agent does the all the activities and the move to end session to quit. This framework originally implemented for English to Sinhala Machine translation [16][17] and it has been improved as a general framework [18].

4.2 Agent Architecture

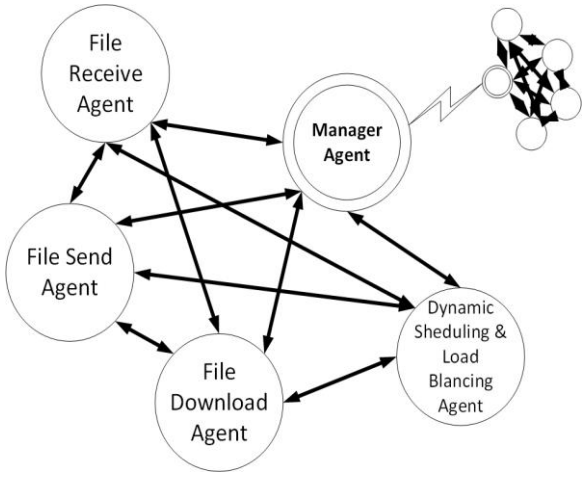

Fig 1: ITray Agent Diagram

This system is containing one manager agent and four ordinary agents namely file send agent, download agent, file receives agent and load balancing and dynamic scheduling agent figure 1 shows the agent architecture. Manager agent communicates with other manager agents which are on other computers by using java socket mechanism. Sockets are responsible for an interface for programming networks at the transport layer. The transport layer consists of two types of protocols, TCP (Transport Control Protocol) and UDP (User Datagram Protocol)

Network communication using sockets is very good communication methods in Java.message agent does their tasks. The agent can work with parallelism, can send a large number of huge file simultaneity without any interference. File send agent send the file via java socket with a specific port.

Manager invokes file send agent by passing the message to that agent for the purpose of sending files to the other peer or peers in the same networks. Figure 2 shows that how communicate agents. That message contains IP address of the peer and the location of the file which want to send to the other machine which is feed by the user. Agent can send multiple files at the same time with the implementation of parallelism in java language without any interference. Get the file location by send agent and send the file via java socket with a specific port.

Manager agent receives a message from another manager agent who is on the same network about receiving the file. Then manager agent invokes file receive agent by passing received a message to the file receive agent. First, catch the file name and after that get the file. This agent can get all kind of file types such as PDF, JPGE, EXE, RAR and etc. and can receive any size of files. Receive agent uses the java server socket technology and agent listen to a specific port to catch the file and related information.

Manager agent invokes download agent by passing message when user feed a URL (Uniform Resource Identifier) to the system. The message contains URL and download agent

waiting for a message to activate. If you allocate a number of URL to the download agent, this agent can download the file in concurrently with downgrading any performance of the system. Download agent makes parallelism to the system by using java threads mechanism and can download any types of files. Download agent file download speed depends on the network speed which is connected to the computer device.

4.3 Task Scheduling

The system maintains an XML-based ontology to as its knowledge base to store peer's IP (Internet Protocol) address. When new users come to the LAN (Local Area Network) others peers share their ontology with new peer and then new peer can build his own ontology. The system maintains list who are the available peer agents in the network by periodically make a communication with other peers by taking IP address which stores in XML ontology. This available peer list is automatically updated time to time by background process and system also give a feature for the user to update the peer list manually. This peer list update is very importance because of, due to many reason peers disconnect their network. As an example peer shutdown, his computer or peer disconnects his network connection. Task scheduling to peers is based on this available peer list. The network is not a static one its change time to time and for task, scheduling in distributed dynamic environment is very challenging for that this communication is very important to get maximum performance and accuracy of the system.

4.4 Load Balancing

The user feeds a lot of URLs to the system to download the system degrades the performance, because of that user want to stay a lot of time to get downloaded files To overcome that problem system maintain a load balancing mechanism to get shared resource with other peers. First, it filters the name of the file using the URL and sends to all other available agents to check that the file is available with them. If available it will send to the requested agent using send agent. Otherwise, when user feeds URL to the system, the system maintains a queue to store URL in a first come first out based mechanism. If system downloads a large, then system going to download two or three files at the same time the download speed becomes slowly and it effects to every downloading files. For overcome that issues system cluster the download URL queue. The number of clusters depends on available peers of the network at that moment. This number is taken from available peer list of the system. After clustering the URL queue then send that URL to other agents according to the IP address of available peers. This all action is done by load balancing and dynamic scheduling agent of the system.

4.5 Task Allocation

Manager agent gets the URL of the download file and sends a message to the download agent for the download file and load balancing and dynamic scheduling agent to check the workload of the system. According to the URL, the file is downloaded to other machine and pass the message to manager agent to send that file to the request agent. Then manager agent invokes file send agent and send that file to the request agent. After downloaded the file or completely allocation task to the other agent, URL is automatically removed from download queue. Using this kind of mechanism download task can be allocated to others in a distributed environment within the same LAN.

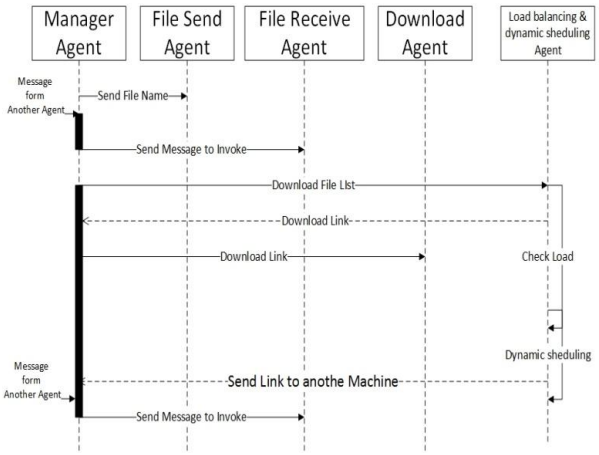

Fig 2: Agent Communication Diagram

5. EXPERIMENTAL RESULTS

The system put into the real environment and collect quantitative and qualitative data for the evaluation. First collect quantitative data without connecting peers, as a standalone download manager. For evaluation system put into the SLT(Sri Lanka Telecom) network and download various types of files such as PDF, EXE, MP3 and etc with different file size. Also used the different web browsers and analyse the time taken to download. Data is collected different blocks of time within a 24 hour day: morning, afternoon and evening in random three days. Finally get the average download time for each file in different browsers and system.

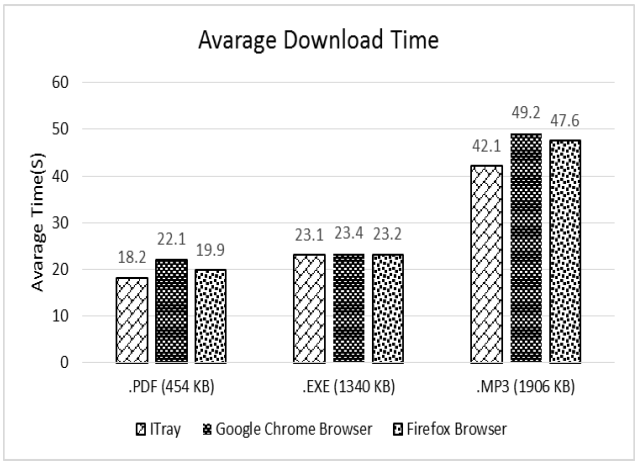

Fig 3: Average Download Time for File

According to the results of figure 3 graph, developed system is very fast download feature. Developed system can download any types of file with high speedily.

The system put into the real environment [20] with a total number of 3 peers in that environment. Connect all peers and let start download files using SLT network. Download list contains the different size of the different file. Above graphs 4 show that the system is working very accurately in a real environment with several peers by sharing a resource with each other.

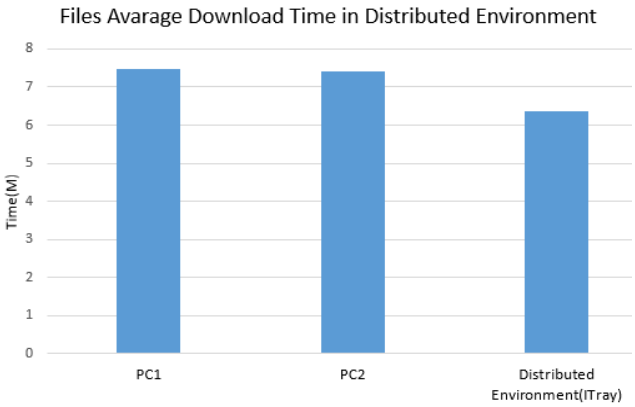

Fig 4: Average Download Time in Different Environments

6. CONCLUSIONS AND FURTHER WORKS

This paper presented the research of resource sharing using multi-agent technology. Due to the assigned tasks, the computer network acts dynamic roles and it always becomes a challenge to each other tasks. Using load balancing mechanism agent reduce the own load and use tasks scheduling to allocate the task to available resources in the network. Using these resources can be shared more effetely in the dynamic and distributed environment. Simulate the MaSMT framework contained with one manager agent and four ordinary agents namely file send agent, file receives agent, download agent and load balancing and dynamic scheduling agent. The core system consists of an XML-based ontology. The system has a very complex communication mechanism to do several tasks over the network. The MaSMT is fully tested under actual environment and it shows that resources can share perfectly within the network. Furthermore, research can be improved by sharing processing power within the network as per the requirements of the network.

7. REFERENCES

[1] N. Leibowitz, M. Ripeanu, and A. Wierzbicki, "Deconstructing the Kazaa network," 2003, pp. 112–120.

[2] J. Lewthwaite and V. Smith, "Limewire examinations," Digit. Investig., vol. 5, pp. S96–S104, Sep. 2008.

[3] F. Bellifemine, G. Caire, A. Poggi, and G. Rimassa, "JADE: A software framework for developing multi-agent applications. Lessons learned," Inf. Softw. Technol., vol. 50, no. 1–2, pp. 10–21, Jan. 2008.

[4] O. Boissier, R. H. Bordini, J. F. Hübner, A. Ricci, and A. Santi, "Multi-agent Oriented Programming with JaCaMo," Sci Comput Program, vol. 78, no. 6, pp. 747–761, Jun. 2013.

[5] F. Klgl, "The Multi-Agent Simulation Environment SeSAm," 2003.

[6] O. Gutknecht and J. Ferber, "The MADKIT Agent Platform Architecture."

[7] B. Hettige, A. S. Karunananda, and G. Rzevski, "MaSMT: A Multi-agent System Development Framework for English-Sinhala Machine Translation," Int. J. Comput. Linguist. Nat. Lang. Process. IJCLNLP, vol. 2, no. 7, pp. 411–416, 2013.

[8] K. Vangheluwe, W. Souffriau, K. Verbeeck, and P. De Causmaecker, "Dynamic scheduling of multi-media streams in home automation systems," in Proceedings of the 7th international joint conference on Autonomous agents and multiagent systems: demo papers, 2008, pp. 1683–1684.

[9] FIPA. Contract net interaction protocol speci_cation. http://www._pa.org/specs/_pa00029/SC00029H.pdf.

[10] M. Grivas and S. J. Turner, "Agent Technology in Load Balancing for Network Applications," in International Workshop on Intelligent Agents on the Internet and Web, Mexico, 1998.

[11] "NetSolve." [Online]. Available: http://icl.cs.utk.edu/netsolve/.

[12] "Defining a Page Space," 27-Mar-2014. [Online]. Available: https://www.ibm.com/support/knowledgecenter/api/content/nl/en-us/SSLTBW_2.1.0/com.ibm.zos.v2r1.idad400/pgsp.htm..

[13] K. Elgazzar, W. Ibrahim, S. Oteafy, and H. S. Hassanein, "RobP2P: A Robust Architecture for Resource Sharing in Mobile Peer-to-Peer Networks," Procedia Comput. Sci., vol. 19, pp. 356–363, 2013.

[14] C. J. Zhang, V. Lesser, and P. Shenoy, "A multi-agent learning approach to resource sharing across computing clusters," UMass Comput. Sci. Tech. Rep. UM-CS-2008-035, 2008.

[15] "The Problem with Shared Network Folders," The Information Management Pulse, 24-Apr-2011. .

[16] B. Hettige, A. S. Karunananda, G. Rzevski, Multi-agent solution for managing complexity in English to Sinhala Machine Translation, International Journal of Design & Nature and Ecodynamics, Volume 11, Issue 2, 2016, 88 – 96.

[17] B. Hettige, A. S. Karunananda, G. Rzevski, "Multi-agent System Technology for Morphological Analysis", Proceedings of the 9th Annual Sessions of Sri Lanka Association for Artificial Intelligence (SLAAI), Colombo, 2012.

[18] HMHR Jayarathna, B Hettige, AgriCom: A communication platform for agriculture sector, Proceedings of the 8th IEEE International Conference Industrial and Information Systems (ICIIS), 2013.

[19] J. Ferber and O. Gutknecht, "A meta-model for the analysis and design of organizations in multi-agent systems," presented at the Multi Agent Systems, 1998. Proceedings. International Conference on, 1998, pp. 128–135.

[20] "ITray." [Online]. Available: https://sourceforge.net/projects/itray/?source=directory