DISTRIBUTED CLOUD COMPUTING

Ben Choi and Surya Bingi
Computer Science, Louisiana Tech University, USA

ABSTRACT

This paper describes a framework for distributed cloud computing, where networked computers share computing resources among group members. The framework extends the concept of cloud computing to form parallel and distributed cloud computing. It allows cloud computing systems to use person computers to provide services. People use web services, such as Google Docs, and on the other hand, they can contribute to the community by allowing their personal computers to help others. The framework combines many key technologies, including intelligent agents, multi-agent system, object space, code mobility, and parallel and distributed computing, into a new computing platform, which has been successfully implemented and tested. It uses intelligent agents and concepts of multi-agent system to allow personal computers to function autonomously to help each other in groups, object space to provide a share place for communication and a place for share knowledgebase, and code mobility to expand the functionality of personal computers. The implementation and test results show that the platform supports distributed computing and high scalability, which allows Internet-scale of people and computers working together in communities.

KEYWORDS

Cloud computing, intelligent agents, parallel distributed processing, multi-agent system, social computing.

1. INTRODUCTION

Current trends to cloud computing will result on overloading the servers. Current researches on cloud computing (Rittinghouse and Ransome 2009) focus on delivering web services. Their proposed method is to move computation away from personal computers into servers, which perform all the needed computation and send the results to personal computers. Powerful personal computers only serve as input/output devices and as displays. Most of the computing power is wasted even when the computer is being used to access web services.

Current researches on parallel and distributed computing and grid computing attempt to employ a very large number of computers to solve very large computing problems. These researches focus solely on computing speed. They partition a very large computing problem into small pieces, send each pierce to be computed by a computer, and then wait for all the results. This centralized control method of computing simply ignores the problem of collaboration between computers. On the other hand, current researches on distributed file sharing based on peer-to-peer networks attempt to allow every person to share his/her files and storage spaces through a decentralized network. This distributed file sharing method facilitates sharing of storage spaces but ignores the needs to share computing power.

Nowadays most computers are networked and can communicate with each other. Communicate is a key requirement for collaboration. The networked computers have provided a much faster and more effective media for people to communicate and to collaborate. The next stage is to create a platform for computers themselves to collaborate with each other.

Our projects attempt to create a platform for computers themselves to collaborate with each other to share computing power (Choi 2010, Choi & Dhawan 2003, 2004). In this platform, computers can help each other both in term of running applications and providing computing power. If a person needs to complete some tasks that are not capable on his own personal computer, his computer will ask other computers for help. His computer makes requests to other helping computers which complete the required computations and returns the results back to his computer. If a person working on certain job needs more computing power, her computer will ask other idle computers for help. Any person using a computer will have access to not just the computing power of his/her own personal computer but the vast computing power of a community of computers.

Our proposed framework will allow cloud computing systems to use person computers to provide services. The computers of cloud computing systems are also belonging to computing communities. They can request other members of the communities for help on performing computations. This extends the concept of cloud computing to form parallel and distributed cloud computing. People use the services provided by the cloud computing systems, and on the other hand, they contribute to the communities by allowing their personal computers to help others. This forms an Internet-scale of machines and people working together in communities.

Our projects combine many key technologies, including parallel and distributed computing, intelligent agents, multi-agent system, object space, and code mobility, to form a unified computing platform. The platform should require minimal user involvement and system administration. To achieve this, our projects extend the notions of intelligent agents (Plekhanova 2002) and multi-agent system (Shamma 2008, Dignum 2009) to conceive of a computer as a whole including its software and hardware as an active agent. A computer acts autonomously like a person in a community. Computers, having various abilities and workloads, join together to form groups where they can help each other both in terms of sharing abilities and workloads. This in turn requires a share place for the computers to communicate with each other. To achieve this, our projects extend the concept of Object Space to become an Active Space, which can function as a rendezvous, a repository, and a manager of its own resources. Code mobility is used to expand the abilities of a computer, allowing any computer to perform any tasks by providing both the program code and the data.

The remaining of this paper is organized as follows. Section 2 outlines the related researches. Section 3 defines the framework of Distributed Cloud Computing. Based on the proposed framework, Section 4 describes an implementation and testing of a platform for distributed computing. And, Section 5 gives the conclusion and outlines the future research.

2. RELATED RESEARCHES

Although currently most computers are networked and can communicate with each other, they cannot yet fully work together and help each other. The ability of a personal computer depends on the installed software and the processing power of its CPU. If a person needs some new applications and more computing power, he/she needs to buy new software and new computer.

Current researches on collaboration focus on allowing people to work together. For instance, Microsoft NetMeeting provides a complete Internet conferencing solution. These researches do not intend to address the problem for computers themselves to collaborate. Current researches on parallel and distributed computing and grid computing attempt to employ a very large number of computers to solve very large computing problems (Berman 2003). For instance, Folding@home uses a very large number of personal computer and PlayStations to tackle previously intractable problems in computational biology. SETI@home (Anderson et al 2002) uses millions of personal computers worldwide to search for extraterrestrial intelligence. These researches focus solely on computing speed and solving very large problems. Current researches on cloud computing (Rittinghouse and Ransome 2009) focus on delivering Web services. On the other hand, current researches on distributed file sharing based on peer-to-peer networks (Androutsellis-Theotokis and Spinellis 2004) attempt to allow every person to share his/her files and storage spaces through a decentralized network but ignore the needs to share computing power.

3. FRAMEWORK FOR DISTRIBUTED CLOUD COMPUTING

In this section, the framework for Distributed Cloud Computing is defined. This computing framework is formulated such that any computer on the Internet can join groups. A group can also link to other groups forming a whole community of computers. The organization of the computers can mimic the organization of a community. A computer can belongs to several groups and benefit from them. Figure 1 depicts a simple organization of twelve computers to form two groups. The center of a group is the group manager that is depicted by a ring. The group manager is a computer serving to provide and maintain a share space for communication. Each of the computers is depicted by a circle. Two of the computers join both groups and one of the computers joins another group not shown in the figure. The two groups are linked together and one of the groups is linked to another group not shown in the figure. A computer has the freedom to join or leave any group at any time. It is a free community and the computing community evolves over time like human community.

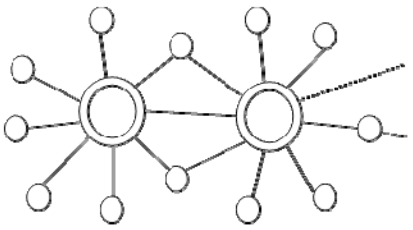
Figure 1. Computers Joining to Form Groups.

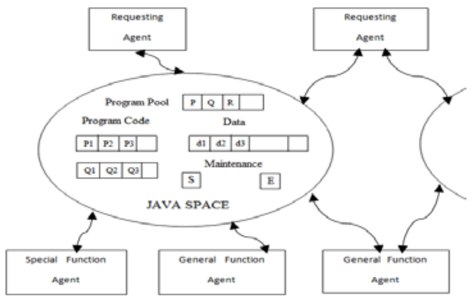

Figure 2. Agents Joining Space to Form Groups.

The ability of a group is more than the sum of ability of its individuals. The function of a group can be considered to be similar to the function of a discussion group. A computer needs a task to be done and posts the request on the group. Another member in the group reads the request, finishes the task, and posts the result on the group. In this analogy, the group serves two purposes: (1) it is a shared place for communication and (2) it is a depository of shared knowledge. Also similar to hosting a discussion group on a server, a computer can be used as a group manager. The group manager helps maintain the shared place for communication and organize the shared knowledge base.

The group provides a shared knowledge base for the computers. This function of the group is also similar to the function of a discussion group. If we have a question, we may find the answer on the prior postings of our discussion group. In this case, we do not need to ask anyone to help. Similarly, if a computer needs a task to be done and can just find the results on the shared knowledge base, then there is no need to repeat the computation.

The group provides a mechanism for parallel and distributed computing. If a computer needs two tasks to be done, it can post both of them on a group. Two other computers can concurrently work on the two independent tasks. This analogy can be extended to multiple tasks and multiple computers working in parallel.

Distributed Cloud Computing provides a general framework for multiple computers to work together in groups to share computing power and knowledgebase. To show the capabilities, this computing framework has been implemented and tested by research projects, which are described in more detail in the following sections.

4. IMPLEMENTING A DISTRIBUTED COMPUTING PLATFORM

Based on the framework for Distributed Cloud Computing, a computing platform for general computing has been developed, implemented, and tested. For the implementation of this platform, several key technologies have been used, including multi-agents, Javaspace (Freeman et al 1999), code mobility (Carzaniga et al 2007), caching, and multicast network protocol (Wen 2001). Figure 2 shows an overview of varies agents joining Space to form groups. In general, a group may consist of a large number of agents and the agents may join several spaces (only five agents and two spaces are show in the figure). Agents and spaces are implemented by extending JINI and Javaspace API’s developed by Sun Microsystems (JINI 2010).

A computer can run many agents and assume many roles. In this platform, we defined three types of agents (as shown in Figure 3). A computer requests other computers for help by using Requesting Agents. A computer serves other computers by using Special Function Agents and General Function Agents. A Special Function Agent can only perform a specific predefined task. A General Function Agent can perform any task that is specified by a Requesting Agent.

Requesting agent	General function Agent	Special function agent
-Name	-Name	-Name
-startComputing() -readFile() -add to pool() -cachesearch() -cacheCollectResults() -generateTasks() -collectResults() -computeRem() -deleteJob()	-getJob() -takeFile() -compileFile() -compteTasks()	-getJob() -computeTasks()

Figure 3. Three Types of Agents.

Data	Program Code	Program Pool
-Name -Data -Result -Calculate flag -Complete flag -Index -No. of agents	-Name -Code -Index	-Pool Name -Program Name -Index

Figure 4. Three Types of Messages.

Code mobility technique is used in this project to enable the platform for general computing. When a computer needs more processing power, it can send both the program code and the data through a Requesting Agent to a Space. The request, in this case, consists of both the program code and the data is stored on the Space. Another computer running a General Function Agent will monitor the Space and retrieve the pending request. The General Function Agent retrieves both the program code and the data. It uses the program code to process the data and generates the required results, which is then send back to the Space. The results are stored in the Space. The requesting computer, through the Requesting Agent, will then retrieve the results from the Space. In general, a large number of requests can be send to a Space and a large number of computers will concurrently work on the requests, creating a general purpose, parallel and distributed computing platform.

A Special Function Agent, unlike a General Function Agent, can only perform a specific predefined task. In this case, the Requesting Agent does not need to send the program code, but only the name of the specific function and the data. A Special Function Agent retrieves and processes the request that matches its specialty.

This processing method is similar to remote execution. However, the communications, in this case, are all through Space. A Requesting Agent does not need to know the destination IP address of a Special Function Agent.

A Space is used in this project not only as a share place for communications but also as a repository of knowledgebase. A sever computer can run the service of a Space, which is implemented in this project by extending the JavaSpace service (JINI 2010). Unlike other servers, this server does not process requests. Its main purpose is to serve as a share place for Agents to meet. In this project, we use multicast network protocol for an Agent to discover a Space to join. Thus, an Agent does not need to know the IP address of a Space. Through the multicast network protocol, it broadcasts its wish to join a Space. A Space responds to the request, and then establishes direct communication with the Agent. An Agent communicates with a Space, by placing requests on the Space and by retrieving results from the Space. An agent communicates with a Space through messages. Three types of messages are defined as shown in Figure 4, which implement the interface net.jini.core.entry.Entry present in the Javaspace API. Messages in a Space are located via associative look up rather than memory location or by identifier. Both the request and the result messages are cached on the Space, which now serves as a repository of knowledgebase. The requests program codes are cashed on the Space, thus that the Requesting Agent does not need to resend the same program codes for used with different sets of data. The computed results are also cashed on the Space, thus that when another Requesting Agent needs the same request, it simply retrieves the results from the Space.

This computing platform has been implemented in our lab using several computers each of which has a network card that supports multicast protocol. Many test cases have been successfully executed to verify the various functionalities of this platform (Bingi 2010). Some test cases, for example, test the fault tolerance of this computing platform, in which a General Agent died (maybe due to computer malfunction) before completing a task. In this case, another General Agent was able to pick up and finish the task. Some test cases test the ability for a Requesting Agent to send both the data and the program code as a request and then a General Function Agent picked up the request, completed the task, and returned the results. The tests results showed that the platform is applicable for general purpose computing.

5. CONCLUSION AND FUTURE RESEARCH

This paper describes a framework for networked computers to work in groups where computers can help each other perform various tasks. A computer can run many agents and assume many roles. Agents join a group through a share place of communication called Space. A sever computer can run the service of a Space. A Space is used in the framework not only as a share place for communications but also as a share knowledgebase. This framework is applicable for current trend of cloud computing by allowing the servers of cloud computing systems to use personal computers to aid providing services, hereby forming distributed cloud computing systems.

Based on the framework, a computing platform has been implemented and tested. The platform is designed for general computing, which uses code mobility to allow a computer to specify a request by sending both the program code and the data. Our experiences with the projects (including projects for creating a distributed computing system for search engine servers (Choi & Dhawan 2003, 2004)) indicate that the proposed framework for distributed cloud computing has many advantages, including high performance, scalability, and availability, requiring less human intervention, and providing natural fault tolerance.

The framework allows computation to be highly parallel and distributed all over the networked computers, and allows personal computers to join together to form large computing communities. The future of computing is moving from personal computers to communities of computers. Any person using a computer will have access to not only the computing power of his/her own personal computer but also the vast computing power of communities of computers.

REFERENCES

Anderson, David P.; Cobb, Jeff; Korpela, Eric; Lebofsky, Matt & Werthimer, Dan, "SETI@home: an experiment in public-resource computing," *Communications of the ACM*, Volume 45, Issue 11, Pages: 56 - 61, November 2002.

Androutsellis-Theotokis, Stephanos & Spinellis, Diomidis, "A survey of peer-to-peer content distribution technologies," *ACM Computing Surveys*, Vol. 36, Issue 4, pp. 335-371, December 2004.

Berman, Fran (Editor); Fox, Geoffrey (Editor); Hey, & Anthony J.G. (Editor), *Grid Computing: Making The Global Infrastructure a Reality*, ISBN-10: 0470853190 Wiley, 2003.

Bingi, S.C. "*Code Mobility with Cache in Distributed Systems using Javaspaces*", Louisiana Tech University practicum report (supervised by Ben Choi), March 2010.

Carzaniga, Antonio; Picco, Gian Pietro; and Vigna, Giovanni, "Is Code Still Moving Around? Looking Back at a Decade of Code Mobility", *Proceedings of the 29th International Conference on Software Engineering*, 2007.

Choi, Ben. "Multiagent Workgroup Computing," *The IADIS International Conference on Internet Technologies & Society (ITS 2010)*, pp.75-82, November 2010.

Choi, Ben and Dhawan, Rohit, "Agent Space Architecture for Search Engines," *IEEE/WIC/ACM International Conference on Intelligent Agent Technology*, pp. 521-525, 2004.

Choi, Ben and Dhawan, Rohit, "Distributed Object Space Cluster Architecture for Search Engines," *High Availability and Performance Computing Workshop*, 2003.

Dignum, Virginia (Editor), *Handbook of Research on Multi-agent Systems: Semantics and Dynamics of Organizational Models*, ISBN-10: 1605662569, Information Science Reference, 2009.

Freeman, Eric; Hupfer, Susanne; and Arnold, Ken, *JavaSpaces: Principles, Patterns, and Practice*, Addison-Wesley, Reading, Massachusetts, 1999.

JINI, "Jini Specifications and API Archive", http://java.sun.com/products/jini/, 2010

Kilduff, Martin and Tsai, Wenpin, *Social Networks and Organizations*, ISBN-10: 0761969578, Sage Publications Ltd, 2003.

Plekhanova, Valentina, *Intelligent Agent Software Engineering*, ISBN-10: 1591400465, IGI Global, 2002.

Rittinghouse, John; Ransome, James, *Cloud Computing: Implementation, Management, and Security*, ISBN-10: 1439806802, CRC, 2009.

Shamma, Jeff (Editor), *Cooperative Control of Distributed Multi-Agent Systems*, Wiley-Interscience, 2008.

Wen, Su; Griffioen, James; and Calvert, Kenneth, "Building multicast services from unicast forwarding and ephemeral state," *OPENARCH 01*, March 2001.