Resource Management Method for Cooperative Web Computing on Computational Grid

Hye-Seon Maeng[1], Tack-Don Han[1], and Shin-Dug Kim[2]

[1] Media System Lab., Dept. of Computer Science,
Yonsei University, Seoul, Korea
{carchi, hantack}@kurene.yonsei.ac.kr
[2] Parallel Processing Lab., Dept. of Computer Science,
Yonsei University, Seoul, Korea
sdkim@kurene.yonsei.ac.kr

Abstract. Web computing only requires connection to a certain URL via Java applet supported web browser. Existing Web computing research has assumed that the application program can be partitioned into a lot of independent modules. In this paper a scalable communication method among distributed processes are proposed. And an analytic model is devised for Web computing communication time according to the characteristics of application programs and computing environment based on this method. It also provides a decision function to determine the degree of hierarchy for managing resources. With this function, the users can determine how the Web computing hierarchy architectures should be constructed to reduce the communication overhead and achieve the performance improvement.

1 Introduction

As the computing methods suggested in meta computing [10], clustering computing [11], and etc. require user account and installation of a certain program to use the other's computer as computing resources. Web computing only requires connection to a certain URL via Java applet supported web browser [1], [2], [3].

Existing Web computing research has assumed that the application program can be partitioned into a lot of independent modules [2], [3]. So eager scheduling method has been adapted as a robust scheduling method. Under this assumption and with this scheduling algorithm, communications among distributed modules cannot be performed. Though some approaches suggested the communication method among distributed modules in Web computing, these methods cannot be scalable to a huge computational environment [1].

In fact, any communication intensive applications cannot be executed with performance gain on the computational grid. But the applications that consist of much ratio of computation time to communication time can be effectively executed in parallel on these environments. So the support for communications among distributed processes can widen the applicable areas in Web computing.

In this paper a scalable communication method among distributed processes are proposed. And an analytic model is devised for Web computing communication time according to the characteristics of application programs and computing environment based on this method. It also provides a decision function to determine the degree of hierarchy for managing resources. With this function, the users can determine how the Web computing hierarchy architectures should be constructed to reduce the communication overhead and achieve the performance improvement.

In Section 2, Web computing hierarchy structure is proposed for cooperative Web computing environment. In Section 3, shared memory mechanism and operating methods for proposed hierarchy structure are explained. Section 4 provides an analytic model for the Web computing execution time and communication time. And also a decision function to determine the degree of hierarchy is introduced. Some real application is analyzed with the analytic communication model and experiment is performed with this application to show the usefulness of suggested method.

2 Web computing virtual environment

When application program can be partitioned into the independent jobs, a manager computer has only a role for job distribution and result gathering. When it should be partitioned into the cooperative jobs, the manager computer should have a responsibility of communication among distributed processes in addition to above roles.

In the latter case, one process assumes a certain process is working normally and tries to communicate with it. So worker computers cannot be added or deleted during execution time and Web computing environment should be managed with stationary method.

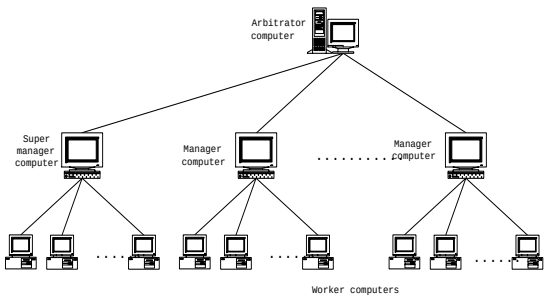

Fig. 1. Web computing hierarchy structure.

In the proposed method, worker computers should be registered to Web computing virtual environment via connecting to an arbitrator computer. When

a client computer requires a group of worker computers, an arbitrator selects worker computers in the pool according to network distance and availability. The numbers of worker computers and manager computers should be determined by a client computer. The decision function for the number of manager computers according to the application characteristics is suggested in next section.

When the needed numbers are fixed, the arbitrator computer constructs virtual computing environment with registered worker computers. As worker computer can open connection to the manager computer under security restriction, the communication among worker computers is performed through communication among manager computers. So the connection between worker-manager and manager-manager is established. The hierarchy structure for proposed Web computing environment is as Figure 1. The super-manager computer is the computer that is in charge of executing a sequential part for application.

3 Shared Memory Mechanism

In this section, a shared memory system model for communication among worker computers and manager computers for cooperative work is proposed. The shared memory space is allocated within the manager computers. As there exist more than one manager computers, a portion of shared memory distributed to each manager computer should form a global shared memory, to be managed with distributed shared memory (DSM) mechanism.

The proposed shared memory mechanism provides two kinds of shared data types to reduce communication overhead and a shared datum can be declared as one of these types. The programmers can select a shared data type according to the access pattern of any data. One shared data type uses data replication algorithm and is called the multiple copied (MC) type. This type is managed with full-replication algorithm [7]. For this type of data, all manager computers have their own copies of data. A read request can be served at its parent manager computer of a worker computer that raises this read request but a write request requires data updates for all the manager computers. This writing policy is from the cache managing method, write-update.

Another data type doesn't replicate data and is called the single copied (SC) type. This type is managed with distributed-central algorithm [7]. When a datum is declared as the SC type, all manager computers create the object for the data but the latest value of that data is maintained in only one manager computer as an owner manager computer of that data. Read/write requests can be served locally in the parent manager computer when the owner manager computer and the parent manager computer are the same computer. This type of accesses is called as local accesses. But in another case, these requests should be served remotely from the owner manager computer and this type of accesses is called as remote accesses. Figure 2 shows the shared data access mechanism according to the data types and locations.

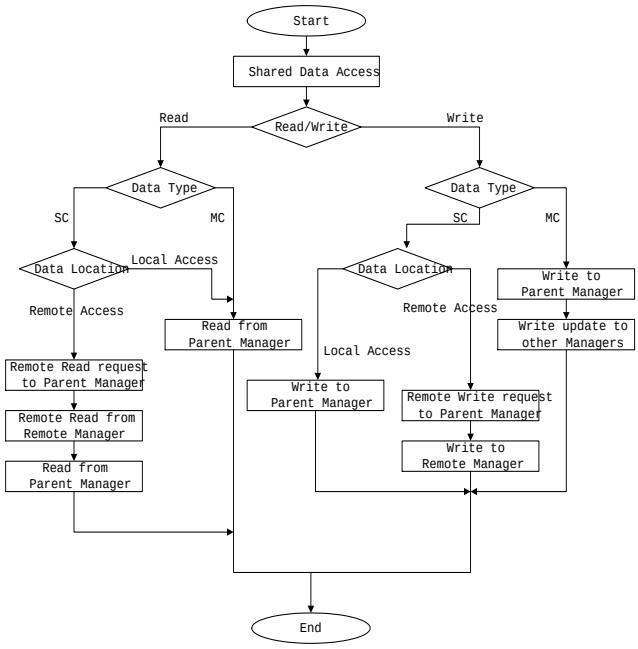

Fig. 2. Shared data access mechanism.

4 The analytic model for execution time and hierarchy level

Parallel execution time is composed of computation time and communication time. The computation time shows little variation when the computer and program are fixed, and it is not shared by any other program. But the communication time can cause much variation according to the circumference status. In this section, the application execution time is analyzed in terms of the computation time and communication time, and then the communication time is analyzed according to various factors. With the analyzed result, a decision function for the degree of hierarchy is introduced. This function can be used to determine the degree of hierarchy to reduce the communication overhead.

4.1 Application execution time

The execution time for parallel computing in the Web computing is divided into the execution time of sequential parts and that of parallel parts. In the proposed environment, the sequential part is executed on a super-manager computer and the parallel parts are executed on all the worker computers. The number of parallel parts can be determined according to any given application program. When the execution time of the i-th sequential part is denoted as TS_i and the

execution time of the i-th parallel part is denoted as TP_i, and the execution time of an application program A is defined as follow.

$$T(A) = TS_0 + \sum_{i=0}^{m} (Max_k(TP_i^k) + TS_i), \tag{1}$$

In this model,m is assumed to be one for simplicity. This model can be easily expanded by repetition of a code pair of one parallel part and one sequential part. When the jobs are well load balanced, $Max_k(TP_i^k)$ can be written as TP_i of any worker computer. Now the execution time of A is redefined as follow;

$$T(A) = TS_0 + TP + TS_r, \tag{2}$$

where TS_0 is the execution time for the front sequential part and is the same as in previous definition. TS_r is that for the rear sequential part and then can be represented as follow;

$$TP = T_{CP} + T_{CM}. \tag{3}$$

As the number of communications is varied according to the application, the computation time and the communication time are denoted as follow;

$$T_{CP} = \sum_{j=1}^{t} T_{CP}^j \quad \text{and} \quad T_{CM} = \sum_{j=1}^{t} T_{CM}^j, \tag{4}$$

where T_{CM}^j is the j-th communication time for a given parallel part and T_{CP}^j is the j-th computation time which happens just before T_{CM}^j. The number of communications, t, can be more than or equal to one. The communication time is heavily influenced by a specific computing environment and the communication overhead can vary depending upon the factors to configure the environment.

4.2 Shared memory access time

In the proposed Web computing environment, communication time is the time to access any shared data. With one manager computer,i.e. under single managing, MC type and SC type are managed with the same mechanism and may involve the same amount of access time. But with more than one manager computers in hierachy,i.e. with multi-managing, the shared data access time is varied according to the type of shared data and their locations as indicated in Figure 2. Cost factors involved in accessing the shared data are defined in Table 1.

With single-managing, the time to access a shared datum from a worker computer to its manager computer, T^{W-M} can be calculated as follow;

$$T^{W-M} = |D| \cdot U_t \tag{5}$$

. To read a shared datum, it is required to send a read request signal. After sending this signal, the shared datum read is transferred. To write a shared datum,

Factor	Definition	Factor	Definition
U_t	Unit data transfer time	P	Number of worker computers
$\|D\|$	Size of transferred data in unit words	M	Number of manager computers

Table 1. The cost factors for accessing shared data.

it is required to send the datum to be written first and then an acknowledgement signal is transferred. The shared data access time should include both the signal and the datum transfer times but the amount of signal transfer time is relatively small when the size of the actual data is big. So the signal transfer time is omitted in Eq. (5).

The condition that there is no communication request simultaneously caused in this application is assumed. When assuming the existence of each communication request from n worker computers, the shared data access time should be redefined as n times of T^{W-M} [4] to represent the worst case. The redefinition of T^{W-M} with an argument as the number of the worker computers that can simultaneously cause any communication request is as follow;

$$T^{W-M}(n) = n \cdot |D| \cdot U_t. \tag{6}$$

When all the worker computers send the requests to access the shared data simultaneously, the time to complete all these requests can be $T^{W-M}(P)$. With Eq.(6), the shared data access time with single-managing can be calculated as Table 2.

Sync. purpose	Access time	Ratio
Yes	$T^{W-M}(P)$	ρ
No	$T^{W-M}(1 + \lfloor (P-1) \cdot c \rfloor)$	μ

Table 2. Shared data access time with single-managing.

A shared data access can be performed for the purpose of synchronization, where simultaneous communication requests from all other worker computers should be reflected in the shared data access time. A shared data access without the purpose of synchronization is not interfered by all other worker computers but by some other worker computers. If a communication request ratio,c, is defined as the ratio of communication time to computation time for any given application, the value of $\lfloor (p-1) \cdot c \rfloor$ means the number of worker computers that can request their respective shared data accesses simlutaneously. The values of ρ and μ are the ratios of data accesses with synchronization purpose and without synchronization purpose respectively. The sum of them becomes always one.

Though an application program may have a large number of shared data accesses, the average access time for any single access can be calculated. The

time to access a single shared datum with single-managing,T_CM^1M is obtained as follow;

$$T_{CM}^{1M} = T^{W-M}(P) \cdot \rho + T^{W-M}(1 + \lfloor (P-1) \cdot c) \rfloor \cdot \mu. \quad (7)$$

To calculate the shared data accesss time with multi-managing, a new cost factor should be introduced. With Eq.(6), the shared data access time to the parent manager computer from a worker computer can be claculated. With multi-managing, the owner manager computer and the parent manager computer can be different. In that case, any handling time occurred from a particular parent manager computer to its owner manager computer should be added to the shared data access time. When there exist n communication requests caused simultaneously from a parent manager computer to some owner manager computers, the completion time of these requests is as follow;

$$T^{M-M}(n) = 2n \cdot |D| \cdot U_t. \quad (8)$$

$T^{M-M}(n)$ shows twice as much as $T^{W-M}(n)$ because a parent manager computer and an owner manager computer are located at different subnets when requesting any communication. When transfering a datum between two different manager computers, the datum should be passed from the sourcing subnet to the target subnet via main network. The transfer time in a subnet is usually longer than ten times of the transfer time in the main network. So the transfer time in the main network is omitted in this equation. With multi-managing, the average number of worker computers is P/M in one subnet. Thus the shared data access time can be classified according to the data types and the locations as obtained as Table 3.

Sync. purpose	Type	Access	Access time	Ratio
Yes	SC	Read/write	$T^{W-M}(\frac{P}{M}) + q_S \cdot T^{M-M}(\frac{P}{M})$	ρ
No	MC	Read	$T^{W-M}(1 + \lfloor (\frac{P}{M} - 1) \cdot c \rfloor)$	μ_{MR}
		Write	$T^{W-M}(1 + \lfloor (\frac{P}{M} - 1) \cdot c \rfloor)$ $+T^{M-M}(M - 1 + \lfloor (\frac{P}{M} - 1) \cdot c \rfloor)$	μ_{MW}
	SC	Read/Write	$T^{W-M}(1 + \lfloor (\frac{P}{M} - 1) \cdot c \rfloor)$ $+q_N \cdot T^{M-M}(1 + \lfloor (\frac{P}{M} - 1) \cdot c \rfloor)$	μ_S

Table 3. Shared data access time with multi-managing.

A shared datum in SC type can be accessed by a local access or a remote access depending on its location and the computer requesting that datum. The values of q_S and q_N are the ratios of remote accesses over the overall shared data accesses with synchronization purpose and without synchronization purpose respectively.

The values of ρ, μ_{MR}, μ_{MW}, and mu_S are the ratios of data accesses for each type and the sum of them is always equal to one. The time to access a single

shared datum under multi-managing with M manager computers, $T_{CM}^{MM}(M)$ is represented as follow;

$$\begin{aligned} T_{CM}^{MM}(M) = \{T^{W-M}(\frac{P}{M}) + q_S \cdot T^{M-M}(\frac{P}{M})\} \cdot \rho\mu \\ +T^{W-M}(1 + \lfloor(\frac{P}{M} - 1) \cdot c\rfloor) \cdot \mu \\ +T^{W-M}(M - 1 + \lfloor(\frac{P}{M} - 1) \cdot c\rfloor) \cdot (\mu + q_N \cdot \mu_S). \end{aligned} \tag{9}$$

From Eq.(7) and Eq.(9), the average reduced overhead for a single shared data access time with M manager computers, $R(M)$ can be calculated as $T_{CM}^{1M} - T_{CM}^{MM}(M)$. From the factors of $R(M)$, is always positive and does not influence any change of the direction curve of $R(M)$, so we can omit this term from the equation. By this, a decision function for the degree of multi-managing, $L(M)$ is defined as follow;

$$\begin{aligned} L(M) = \frac{R(M)}{U_t \cdot |D|} = \rho \cdot P(1 - \frac{1 + 2q_S}{M}) + \mu \cdot \lfloor(P - 1) \cdot c\rfloor \\ -(\mu_{MR} + 3\mu_{MW} + (1 + 2q_N) \cdot \mu_S) \cdot \lfloor(\frac{P}{M} - 1) \cdot c\rfloor \\ -2((M - 1)\mu_{MW} + q_M \cdot \mu_S), \end{aligned} \tag{10}$$

assuming that M is more than or equal to two.

To obtain the most reduced communication overhead, the degree of multi-managing can be determined when $L(M)$ has a maximum value. But the more manager computers mean the high costs so the degree of multi-managing can be obtained according to a trade-off between the reduced communication overhead and the cost of multi-managing.

5 Application program analysis and experiment

An application is analyzed in terms of parameters by using the analytic communication model and the degree of multi-managing is determined. And then experiments are performed with these applications to show the effectiveness of multi-managing. The experiment is performed in the Ethernet simulation environment with SMPL [Mac87]'s Ether model.

The Jacobi algorithm [8] is an iterative method for solving the linear system $Ax = B$ for the unknown vector x. As this program involves a lot of data parallelism, both the computation and communication loads can be well balanced among the worker computers.

Let an application program A be the probem on which Jacobi algorithm is applied to $K \cdot K$ grid to determine the next temperation. Figure 3 shows the shared data portion among the manager computers when using P worker

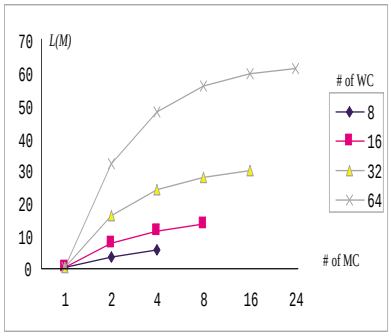

Fig. 3. Value of L(M) function.

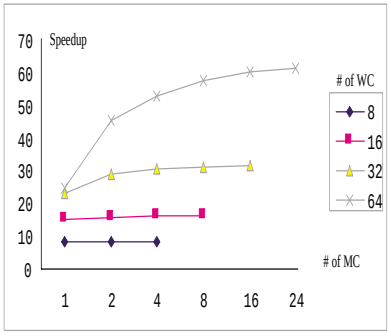

Fig. 4. Execution speedup.

computers ($WC_1 - WC_P$) and Mmanager computers ($MC_1 - MC_M$). All the worker computers except the both side worker computers have two $1 \cdot K$ data in their shared memory. Each of worker computers located at both sides has only one $1 \cdot K$ data. The whole size of shared data is $1(P-1) \cdot K$. All data accesses are performed for synchronization purpose and so all shared data are declared as SC type. When using M manager computers, $a(M-1) \cdot K$ elements are shared by each of two manager computers. As all SC type data should be located within one of two manager computers, the ratio of remote accesses to the whole SC type data,q_s can be calculated as follow;

$$q_s = 0.5 \cdot \frac{2(M-1)K}{2(P-1)K} = \frac{M-1}{2(P-1)}$$

As all shared data are declared as SC type, the value of ρ is one and the ratios of other factors are all zero. With these factors, $L(M)$ function shows the the result as in Figure 3. It shows that the value of $L(M)$ function increases much more when the number of worker computers increases because in that case the overhead of communication becomes much significant. The value of $L(M)$ increases when the number of manager computers increases because multi-managing can reduce the communication overhead. The value of $L(M)$ will converge when the number of manager computers increases much more and the user should determine the number of manager computers according to a trade-off between the reduced communication overhead and the cost of multi-managing.

The experiment with $10^5 \cdot 10^5$ area and 500 iterations is executed. Figure 4 shows the variation of speedup when the number of manager computers varies. The result of the speedup graph looks similar to the result of $L(M)$. When the number of worker computers is quite big, the speedup is far from the linear speedup with single-managing but it becomes near the linear speedup with multi-managing. High degree of multi-managing can help avoiding this network contention much more in such an application with high data parallelism especially when there are a large number of worker computers.

6 Conclusion

In this paper a scalable communication method among distributed processes are proposed. And an analytic model is devised for Web computing communication time according to the characteristics of application programs and computing environment based on this method. It also provides a decision function to determine the degree of hierarchy for managing resources. With this function, the users can determine how the Web computing hierarchy architectures should be constructed to reduce the communication overhead and achieve the performance improvement.

References

1. Arash B., Mehmet K., Zvi K., and Peter W.: Charlotte: Metacomputing on the Web, International Conference on Parallel and Distributed Computing Systems, (1996) 151-159.
2. Bernd O. C., Peter C., Mihai F. Ionescu,: Javelin: Internet-Based Parallel Computing Using Java, ACM Workshop on Java for Science and Engineering Computation, (1997) 30-40.
3. Sarmenta, L.,: Bayanihan: Web-Based Volunteer Computing Using Java, Int'l. Conference on World-Wide Computing and its Applications (WWCA'98), Springer-Verlag Lecture Notes in Computer Science, (1368), (1998) 444-461..
4. M. J. Clement, M. R. Steed, P. E. Crandall: Network Performance Modeling for PVM Clusters, Supercomputing, (1996).
5. M. H. MacDougall: Simulating Computer Systems: Techniques and Tools, The MIT Press, (1987).
6. Holger Karl: Bridging the Gap between Distributed Shared Memory and Message Passing, Concurrency: Practice and Experience (10), (1998) 1-14.
7. Michael S. and Songnian Z.: Algorithms Implementing Distributed Shared Memory, IEEE Computer, (1990) 54-64..
8. M.J. Quinn: Parallel Computing:Theory and Practice, McGraw-Hill Book Company, (1994).
9. D. Zwillinger: Handbook of Integration, Jones and Bartlett, (1992).
10. A. S. Grimshaw and W. A. Wulf: The legion vision of a worldwide virtual computer, Communication of the ACM, 40(1), (1997).
11. Chungmin C., Kenneth S., Miron L.: The DBC: Processing Scientific Data Over the Internet, International Conference on Distributed Computing Systems, (1996).