New Generation Computing, 28(2010)129-136
Ohmsha, Ltd. and Springer

Multi-path based Algorithms for Data Transfer in the Grid Environment

Muzhou XIONG[1,2], Dan CHEN[2,3*],
Hai JIN[1] and Song WU[1]
[1]*School of Computer Science and Technology,*
Huazhong University of Science and Technology,
Wuhan, Hubei, CHINA, 430074
[2]*School of Computer Science,*
China University of Geosciences,
Wuhan, Hubei, CHINA, 430074
[3]*School of Computer Science,*
University of Birmingham, Birmingham,
B15 2TT, UNITED KINGDOM
{mzxiong, hjin, wusong}@mail.hust.edu.cn,
d.chen@cs.bham.ac.uk

Received 28 November 2008
Revised manuscript received 14 April 2009

Abstract This paper introduces two alternative algorithms for efficient data transfer in the Grid environment. For data transfer from a source node to the destination node, the algorithms can construct multiple dynamic paths by selecting some other nodes as data relays. The bandwidth available in different paths can be aggregated thus to significantly speed up the data transfer process. The proposed algorithms differ from each other in whether the global networking information should be considered. Experimental results indicate that both algorithms can provide efficient data transfer under various circumstances.

Keywords: Data Transfer, Multi-path Approach, Grid Environment.

* Corresponding Author (Dan Chen). This research was supported by the National Natural Science Foundation of China under the project grant No. 60804036. It is a collaborative work among the UoB, CUG and HUST under the Birmingham-Warwick Science City Research Interdisciplinary Alliance.

§1 Introduction

Data transfer is one of the key issues in Grid computing. The efficiency of data transfer directly affects the performance of the whole Grid environment.[1] Nowadays, GridFTP [2] is the *de facto* standard method for data transfer in Grid computing. However, the performance of GridFTP is mainly limited by the low bandwidth available in the data channel from the server to the client.

Multi-path approaches have been adopted to make use of the bandwidth of multiple alternative paths between two nodes over Internet. The approach may work at different networking levels. Chen et al [3] present an example of multi-path routing protocol, which exhibits a better performance than that of single-path routing protocols. Existing work operating at routing protocol level needs to rebuild existing network protocols. Cheng et al [4] introduces an instance of multi-path approach working at the application level. Similar approach has also been aimed at P2P networks, such as SplitStream.[5] These demand that the whole copies or parts of the original data file should be distributed in the LAN or Internet.

Significantly different from the above-mentioned work, this study has the following aims: (1) the candidate approach should work at the application level to facilitate easy deployment over Internet, which need no change to the routing protocols; (2) the approach needs not to duplicate data, thus to avoid the cost of maintaining consistency and storing redundant data.

§2 Multi-path based Algorithms

The discussions about the algorithms in this paper assume an overlay network in the Grid environment. The overlay network consists of three types of nodes in terms of data transfer, i.e., source nodes, destination nodes, and multiple intermediate nodes (relays) which can be explicitly exploited for relaying data between the source and the destination nodes. A given pair of nodes can be linked together directly or through one intermediate node (one relay) which forms an exclusive *path*. Traditional data transfer techniques (e.g., GridFTP) only support a single path between the source and the destination conforming to some routing protocol (e.g., BGP over TCP). In contrast, our approach enables connecting two nodes via multiple paths. The notation "*channel*" is referred to as a set of paths via any of which data can be transferred between the two nodes. A channel can have a *default path* determined by the routing protocol. Another type of paths is named *Path via Relay* (*PvR*) as each PvR contains a relay selected by the proposed algorithms. In the context of our approach, data transfer between the source node and the destination node can be performed over the multiple paths simultaneously.

Prior to sending data out, two important issues should be addressed: (1) choosing relays; and (2) controlling data sending rate on each candidate path. In fact, the problem of how to select relays and how to control sending rate is the core problem in optimizing data transfer over multiple paths. The problem of relay selection has been proved NP-hard.[6] More details about the two issues are available in sections 2.1 and 2.2.

2.1 Relay Selection

The primary objective of relay selection is to select appropriate relays to achieve a high data transfer rate. Meanwhile, it is also essential to avoid any congested relay which has an ultra high computational or communicational load. The congestion could be incurred when the relay has been competed by a number of source nodes.

The whole data can be transferred in a number of rounds, and in each round a portion of the data will be sent. In each round, the source node selects a relay (or no relay for default path), which is associated with a dynamic probability. Figure 1 illustrates two candidate algorithms for relay selection, namely the Specification-Unaware (SU) algorithm and the Specification-Aware (SA) algorithm. The two algorithms assign the initial probability of each relay (or no relay for default path) in different ways. When the SU algorithm is adopted, the initial probabilities for all relays are the same, with a value of $1/Number\_of\_Nodes\_in\_Network$ (see the function *Init-SU()*). After initialization, either algorithm will be used to select relays based on their associated probabilities. If congestion is detected from a relay in some round, the relay's probability to be selected in the next round will be reduced to half of the current probability; otherwise, the probability will be doubled. After that, the probabilities of all relays will be normalized between 0 and 1 such that their sum is always equal to 1 (from a particular source node's perspective). We anticipate that appropriate relay selection procedure will be stabilized to maximize the bandwidth exploitable in the data channel linking the source and the destination after a number of rounds.

The SU algorithm does not consider the network topology and connecting conditions at the initial stage. It therefore will take a significantly long time for the algorithm to select the appropriate relays after blindly selecting relays on initialization. Nevertheless, in the Grid environment, knowledge of the topology and status of the *underlying physical network* can be sensed by source nodes. The network can always be classified into several Autonomous Systems. Typically a pair of nodes in an Autonomous System holds a networking condition (e.g., bandwidth) similar to that between any other two nodes in the same AS. In

```
/*parameters in the algorithm
 SRC: the set of sources containing the
  requested data;
 RLY: set of relays;
 P_sr^n: The selection probability of PvR from
  source s to the destination via relay r at
  the nth round;
RL_n: current selected relay;
sr_ir:Run-time data sending rate of source i
  through the PvR containing the relay r;
D_i: maximum receiving rate of destination;
S_i: maximum sending rate of source;
TSR_r: the real sending rate of relay r;
MSR_r: the maximum sending rate of relay r;
threshold: the probability assigned to the
relay if the source, relay and destination
belong to the same node set
*/
Begin
  If(SU_Algorithm)
    Init-SU()
Else If
  Init-SA(set)
End If
 While(transfer task is not complete from
  any s in SRC?)
  For all the relay node r in RLY
    P_sr^n = P_sr^n-1
    if(path through r congested)
         P_sr^n = 0.5*P_sr^n-1
     end if
     else
         P_sr^n =min(2* P_sr^n-1 ,1)
     end else
    end for
    Normalize P_sr^n for all r in RLY
   RL_i<-null
    While(∑sr_ir<min(S_i, D_i) or TSR_r>MSR_r)
       Randomly select relay
        according to P_sr^n
     RL_i<-r
    End While
       waiting for the next round
  End While
End

Funtion Init-SU()
  For all the relays
    P_sr^0= 1/Number\_of\_Nodes\_in\_Network
  End For

Function Init-SA ()
  For all the relay node r in RLY
    if r, source and destination belong to
      the same node set
       P_sr^0 = threshold;
    else
       P_sr^0= 1/Number\_of\_Nodes\_in\_Network
    End if
  End For
  Normalizing P_sr^0 to Normal_P_sr^0 such that
   ∑Normal_P_sr^0 for all the relays r
```

Fig. 1 Algorithms of Specification-unaware and Specification-aware Relay Selection

this study, the nodes in an Autonomous System are called a node set. Some special nodes may exist in multiple sets. This fact implies that the data transfer rate from node in one set to the destination in another set may be accelerated by selecting such special nodes as relays. The SU algorithm by its nature will not be aware of this information at the beginning, although such configuration can be detected through relay selections in the long run. Assume that all nodes in the Grid environment are aware of this configuration, it is then possible to shorten the time for convergence. The SA algorithm has been designed based on this assumption. The process of relay selection in the SA algorithm is similar to that in the SU algorithm. The source still selects relays randomly as in the SA algorithm. At the stage of initializing the selection probability for each relay (see function *Init-SA()*), if the source and the destination belong to different sets, the nodes belonging to these two sets will always be allocated with a high priority. If the source and the destination are in the same set, nodes in this set will be selected with a higher probability than those in other sets. A threshold will be defined in this case. The probabilities assigned to these nodes will not become less than the threshold.

2.2 Sending Rate Control

The method of sending rate control aims to avoid the potential congestion problem in case multiple sources competing for a single relay. In this method, each PvR holds a maximum data sending rate from the source i to the destination via the relay r, notated as C_{ir}. The maximum data sending rate actually represents the *capacity* for data transfer of this path. If there is only one source node that selects relay r, the relay will forward data from the source node to the destination node with its maximum capability, i.e., C_{ir}. When multiple sources select r at the same time, the rate at which each source may send data will be determined as follows: (1) getting the current probability that r may be selected by a source i, i.e., P^n_{ir}; (2) normalizing all probabilities (from the relay r's perspective) to $n\_P^n_{ir}$, such that $\sum_i n\_P^n_{ir} = 1$; and (3) the sending rate of each source i via relay r is calculated as $n\_P^n_{ir} \times C_{ir}$.

§3 Performance Evaluation

This section presents a performance evaluation of the SA algorithm and the SU algorithm. The two algorithms have been simulated upon Network Simulation 2 (NS2, more details see http://www.isi.edu/nsnam/ns/). The simulation was aimed to emulate a Grid environment of a medium scale with a common networking condition of Grid environments in practice. In the experiments, we assume that there are 200 nodes in the Grid environment and the maximum sending rate of each source is 1 Mbps. For the experiments with multiple source nodes, the number of source nodes has been fixed to 20, which represents the situation that the whole Grid environment is neither too busy nor idle. All candidate relays of a source are indexed with an integer (initialized as 1). The bandwidth via the jth relay from the source to the destination is proportional

to $1/j^{\beta}$, where β is the skew factor and is subject to $0 \leq \beta \leq 1$. When $\beta = 0$, all the relays have the same bandwidth. As β increases, the bandwidth distribution among the relays becomes even more asymmetric. This represents the situation in which although some links have a high bandwidth, the overall networking performance is still poor. Here we use a *normalized sending rate* as the criteria to evaluate the performance of data transfer under different conditions. The maximum rate at which a source node/destination node sending/receiving data to/from the *underlying physical network* is exactly determined by the bandwidth available to the direct link via which the node connects to the Internet. From the perspective of a source node i, the *utilization rate* of its direct link to the Internet can be denoted by the ratio of it current sending rate csr_i against the direct link's bandwidth B_i. When multiple sources send data simultaneously, their *normalized sending rate* are written as $\sum_i (csr_i / \sum_i B_i)$, which reflects the average utilization rate of all source nodes' directs links to the Internet.

These simulation experiments have been carried out to compare the normalized sending rate of the SA algorithm and the SU algorithm when there exist a single data source or multiple data sources. Another objective is to study how fast the two candidate algorithms can identify the appropriate relays to optimize the data transfer procedure eventually (the convergence rate). Different values of β (0.25, 0.5, 0.75 and 1) have been used in the simulation experiments. The experimental results are presented in Fig. 2 in the case of a single data source. The results indicate that both algorithms can approach the maximum sending rate in a short time (5 and 8 rounds for the SU algorithm and the SA algorithm respectively) when β has a small value (0.25). In contrast, when β increases, the convergence rates of both algorithms decrease (27 and 35 rounds for the SU algorithm and the SA algorithm respectively when β is 1). This is because when the network becomes more asymmetric, both algorithms need more time to detect the most appropriate relays. Besides the above, the normalized sending rates of both algorithms decrease when β increases. The network becoming more asymmetric (i.e., the value of β increases) means there will be less relays offering high bandwidths. The results indicate that the performance of the algorithms is directly affected by the network condition.

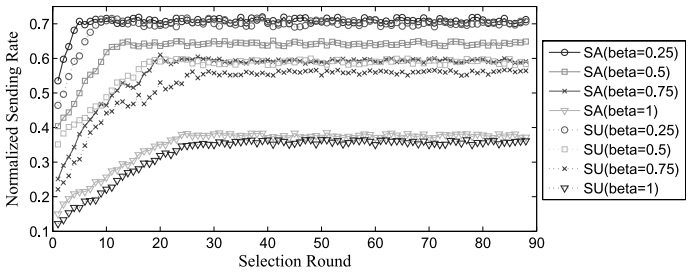


Fig. 2 Normalized Sending Rate with Single Sending Source and Different Values of β

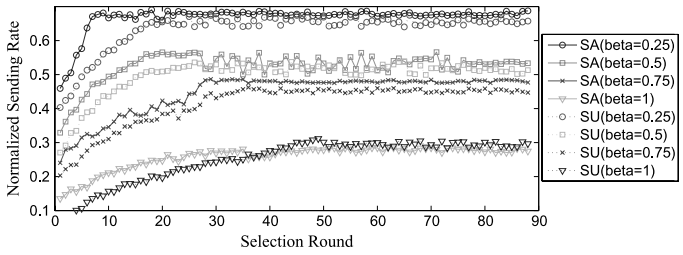


Fig. 3 Normalized Sending Rate with Multiple Sending Sources and Different Value of β

The experimental results for multiple source nodes are presented in Fig. 3. With the same values of β specified, the normalized sending rate obtained from the experiments with multiple concurrent data transfers is less than that with only one source node. This is because multiple sending sources may compete for the same relay and cause congestion. It can also be observed that the difference between the results for the two algorithms becomes less when the value of β increases. As the value of β increases, congestion will occur as sources are likely to compete for a single relay. As a result, the relay cannot provide enough bandwidth as expected.

§4 Conclusions

In this paper, two data transfer algorithms in the Grid environment have been proposed, namely the SU algorithm and the SA algorithm. Both algorithms select some nodes as the relays to form multiple paths for data transfer between a pair of nodes. The SA algorithm requires each node being aware of the status that it connects to other nodes. In contrast, this is not required by the SU algorithm, and it can still adapt to dynamic network conditions. Experiments have been carried out to evaluate the performance of the algorithms in a Grid environment of medium scale. The results indicate that the SA algorithm holds a higher convergence rate than that of the SU algorithm.

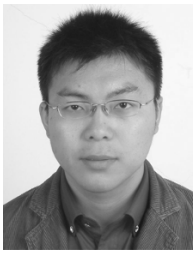
For the future work, more experiments about extra-cost evaluation and performance comparison with GridFTP will be conducted to clearly quantify the potential benefits of the proposed algorithms. Secondly, we need to address how to apply the algorithms to the real systems in practice, in which a variety of sizes of source data need to be handled properly.

References

1) Wang, L. and Jie, W., "Towards supporting multiple virtual private computing environments on computational Grids," *Advances in Engineering Software, 40, 4*, pp. 239-245, 2009.

2) Allcock, W. E., Bester, J., Bresnahan, J., Chervenak, A.L., Foster I., Kesselman, C., Meder, S., Nefedova, V., Quesnel, D., and Tuecke, S., "Data management and transfer in high-performance computational grid environments," *Parallel Computing, 28, 5*, pp. 749-771, 2002.

3) Chen, J., Druschel, P., and Subramanian, D., "An efficient multipath forwarding method," in *Proc. the Seventeenth Annual Joint Conference of the IEEE Computer and Communications Societies 1998*, IEEE Computer Society, pp. 1418-1425, 1998.
4) Cheng, B., Chou, C., Golubchik, L., Khuller, S. and Wan, Y.-C., "Large scale data collection: a coordinated approach," in *Proc. of the 22nd Annual Joint Conference of the IEEE Computer and Communications Societies 2003*, IEEE Communications Society, pp. 218-228, 2003.
5) Castro, M., Druschel, P., Kermarrec, A-M., Nandi, A., Rowstron, A. and Singh, A., "SplitStream: High-bandwidth multicast in a cooperative environment," in *Proc. of Symposium on Operating Systems Principles 2003* (Michael, L. S. and Larry, L. P. ed.), ACM Press, pp. 298-313, 2003.
6) Xiong, M. Z. and Jin, H., "Optimization Algorithms for Data Transfer in the Grid Environment," *Quantitative Quality of Service for Grid Computing: Applications for Heterogeneity, Large-Scale Distribution and Dynamic Environments* (Wang, L., Chen, J. and Jie, W. ed.), IGI Global, Chapter XXII, 2009.



Muzhou Xiong: He is a Ph.D. candidate at Huazhong University of Science and technology (China). His research interests include grid computing, network storage, and crowd modelling and simulation.

Dan Chen: He is a professor with the School of Computer Science, China University of Geosciences. He is also a joint HEFCE Fellow with the University of Birmingham and the University of Warwick (United Kingdom). His research interests include high performance computing and neuroinformatics.

Hai Jin: He is a professor of computer science and engineering at the Huazhong University of Science and Technology (HUST) in China. His research interests include computer architecture, virtualization technology, cluster and grid computing, peer-to-peer computing, network storage, and network security.

Song Wu: He is a professor of Computer Science and Engineering at the Huazhong University of Science and Technology (HUST) in China. His current research interests include Grid/Cloud computing and virtualization technology.