3-5 On Influence of Network Characteristics on Application Performance in a Grid Environment

KITATSUJI Yoshinori, TAGASHIRA Hideki, YAMAZAKI Katsuyuki, TSURU Masato, and OIE Yuji

In grid computing, a key issue is how limited network resources can be shared by communications by various applications more effectively in order to improve application-level performance, e.g., by reducing the completion time for individual applications and/or set of applications. Communication by an application changes the condition of the network resources, which may, in turn, affect communications by other applications, and thus may degrade their performance. In this paper, we examine the characteristics of traffic generated by typical grid applications, and the effect of the round-trip time and bottleneck bandwidth on the application-level performance (i.e., completion time) of these applications. Our experiments show that the impact of network conditions on the performance of various applications and the impact of application traffic on network conditions differ considerably depending on the application. These results suggest that effective allocation of network resources must take into account the network-related properties of individual applications.

Keywords
Grid, Network resource, Application-level performance, Distributed computing, Traffic engineering

1 Introduction

With the availability of high-performance off-the-shelf computers and high-speed wide-area networks, large-scale distributed computing environments are growing at an amazing speed. These environments enable massive computations to be performed using a large number of computers connected over WAN (Wide Area Networks). This form of distributed computing (grid computing) dynamically involves a number of heterogeneous computing resources (e.g., CPU, memory, storage, application program, data, and so on.) connected by heterogeneous network resources across geographically dispersed organizations[1][2]. The fundamental challenge for grid computing is to achieve a good performance from multiple distributed applications that share limited and heterogeneous computing and network resources (e.g., in terms of the completion time for each independent application and/or set of related applications). Large amounts of data are now being handled by distributed applications and the time required to transmit data is increasing, which has a significant effect on the performance of an application. Effective share of network resources is therefore critical.

There is considerable work on traffic engineering to improve the utilization of network resources. To name a few, Elwalid et al. proposed a decentralized method to balance flows over multiple paths based on the traffic load of the paths obtained by active end-to-end measurement along the paths[3]. Guven et al. proposed that routers re-balance flows over multiple paths with the information collected by a measurement device which passively measured loss and available bandwidth of connected links[4]. Although both suggested re-balancing traffic over multiple paths in terms of the network utilization, there is little insight on to improve the application performance. they neither considered the traffic patterns generated by various applications nor the application-level performance.

As focusing on improving the application performance, Kawahara proposed equalizing throughput of TCP connections established in a path in terms of improvement of the total performance of data transmitted among those connections, e.g., by reducing the time of file transmission[5]. However, the requirements of the network resources assigned to applications are diverse, e.g., some application may require a broadband path, and others may a low end-to-end delay one with narrow bandwidth instead.

Another direction of related work focuses on improving the application-level performance. Rao proposed a method based on estimating delay regression to achieve low end-to-end delays for message transmissions by using two paths for distributed computing applications[6]. This method does not allow for the fact that the application-level performance is influenced by both the delay and available bandwidth of the selected path. Thus, the method may unnecessarily assign a short-delay path to an application which rather requires a broadband path. Aida and Osumi illustrated the case study on improving the performance of a master-worker application, which was grouped into a work flow process (described in Section **2**), by localizing the communication frequently occurred, into a PC cluster or a single PC to reduce the communication influenced by a long delay[7]. Plaat et al. investigated the impact of a limited bandwidth and a long delay on the application-level performance for 6 applications, and suggested how to optimize the algorithms of them to weaken the sensitivity of application performance to a limited bandwidth and a latency and, thus, to improve their performance under the limited network resource conditions[8]. They mostly focus on optimizing the task allocation to reduce the effect of state of network resources on the distributed computing applications. However, in the case that multiple applications simultaneously run in a dedicated network, they neither consider influence of the traffic generated by each application on the other applications nor how to share the limited network resources among them to obtain the better performance in total.

Let us suppose that we could determine the network-related properties of various applications in advance based on either a test run or the first run in a series of repeated runs. There are two aspects to the network-related properties of an application i.e., how the performance of individual applications is affected by the state of the network resources and, conversely, how traffic generated by individual applications affects the state of the network resources. A scenario in which the network-related properties of applications are taken into account might enable these distributed applications to be scheduled to share the network resources on an internal-network more effectively as well as the computing resources at the end-nodes. To address this issue, therefore, we investigated whether some applications show specific performance characteristics in relation to the network. We focus on the sensitivity of some typical distributed applications in terms of completion time under different network resource conditions. The results show a non-trivial tension between the performance of an application and the network resource conditions, which can be utilized to achieve more effective sharing of the network resources, allowing multiple applications to be executed in parallel.

2 Distribute computing applications

In this paper, we use four distributed computing applications which display distinct traffic features, as follows.

- N Queen
  solves the placement of *N* Queens on an *N* by *N* grid such that none of them shares a common row, column or diagonal.
- Jigsaw Puzzle
  solves a jigsaw puzzle problem for computers, which comes originally from Problem C in the 2001 ACM International College Programming Contest, Asia Preliminaries in Hakodate[9]. The problem is expanded to take the rotation and size of a piece into account.
- LU Decomposition
  is one of the programs in the NAS Parallel Benchmarks[10] for solving a lower triangular matrix, *L*, and an upper triangular matrix, *U*, composing an *N* by *N* matrix, *A* (= *LU*).
- Task Scheduling
  is a task scheduling program that deals with standard task graph archives[11]. The algorithm assigns a task to a computer that has sufficient available memory in the order of a priority given to a task belonging to a task flow that takes longer to be processed.

N Queen, Jigsaw Puzzle, and LU Decomposition are classified as task-framing programs in terms of type of distributed computing. Task Scheduling is classified as a work flow program[12]. In the task-framing process, a master distributes the tasks that make up the target problem among a farm of multiple slave processors and gathers the partial results to produce the final result of the computation. Generally, communication between the master and slaves involves huge transfers of data. In work flow processing, the target problem is divided up into multiple pipelined stages and/or steps and various amounts of data are asynchronously exchanged between pairs of processors.

3 Experimental environment

We used the network configuration shown in Figure 1 for the experiments described in this paper.

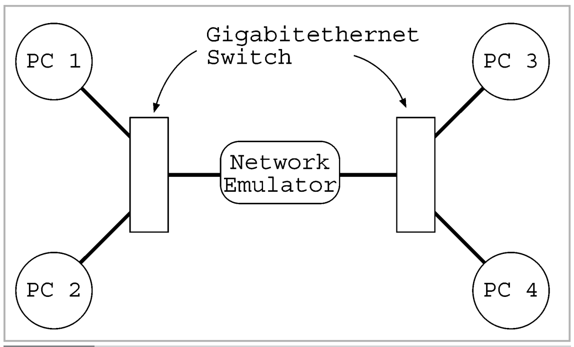


Fig. 1 Network Configuration.

The distributed processing for each application described in Section **2** was performed on computers PC1 through PC4 starting from PC1. All the computers had the following basic features: Xeon 3.06 GHz CPU, 2 GBytes memory, Intel (R) PRO/1000 NIC and PCI-X bus. The speed of all links was 1 Gbit/s.

For the task-framing applications, the master process ran on PC1, and the slave processes ran on all the computers including PC1. A network emulator[13] was used to insert latency and to shape a bottleneck link in packet forwarding between two switches. Measurements were performed by capturing all the packets sent to or received from each of the computers. The average round-trip times (RTTs) were 0.141 ms and 0.331 ms between PC1 and PC2, and between PC1 and PC3, respectively, without any latency inserted.

4 Analysis of communication features

We investigated the features of the application traffic, e.g., the amount of transferred data, fluctuations in throughput and the flow composing the application traffic.

Table 1 shows the total amount of transmitted data, average throughput, and completion time for the individual applications. The transmitted data and throughput were measured at PC3 when an application was run with the sufficient network resources, where the average RTTs were 0.114 and 0.331 ms and the bandwidth of links was 1 Gbit/s. The values in the table are the average of 20 exper-

Table 1 *Feature of traffic to/from PC3.*

Application	Incoming traffic to PC3		Outgoing traffic from PC3		Completion Time (s)
	Total amount of traffic (MB)	Ave. throughput (Mbit/s)	Total amount of traffic (MB)	Ave. throughput (Mbit/s)	
N Queen (15 ×15 grids)	0.926	0.878	39.2	37.2	40.78
N Queen (16 ×16 grids)	4.00	0.167	242	10.1	191.33
Jigsaw Puzzle (4 ×4 pieces)	0.0780	0.0443	0.117	0.0666	13.70
Jigsaw Puzzle (36 ×36 pieces)	101	4.56	194	8.79	176.48
LU Decomposition (36 ×36 matrix)	67.5	10.4	67.2	10.3	50.18
LU Decomposition (64 ×64 matrix)	179	5.24	179	5.23	258.64
Task Scheduling (300 tasks)	116	7.68	132	8.78	131.83
Task Scheduling (500 tasks)	163	10.1	188	11.6	132.52

iments.

All applications take longer to complete their processing and transmit larger amounts of data as the scale of the problems becomes larger. The relationship between the amount of transmitted data and the completion time greatly differ depending on the application. For N Queen, the large scale of problem increases both the amount of data and the completion time by 6 times. Jigsaw Puzzle shows a 10-fold increase in completion time while the data increases significantly by more than 1000 times. The completion time for Task Scheduling scarcely changes while the amount of data is increased by 1.5 times. For LU Decomposition, in contrast to the other applications, the average throughput decreases by half when the completion time and amount of transmitted data are increased by 5 and 3 times, respectively. The completion time for Jigsaw Puzzle and Task Scheduling are less affected by an increased amount of data, probably because the average throughput increases to convey larger amounts of data.

Note that in the following experiments, we present the results of using the 16×16 grids for N Queen, 36 x 36 pieces for Jigsaw Puzzle, 102×102 matrix for LU Decomposition, and 500 tasks for Task Scheduling because the alternative cases tended to produce quite similar results to those.

Figure 2 shows the fluctuations in the 10-ms-average throughput of data transmitted to/from PC3 for one instance of the 20 experiments described in Table 1. Note that both PC2 and PC4 show a similar traffic pattern to PC3 for all the applications. However, the patterns between applications were completely different. N Queen sends a huge amount of data from the slave to the master (outgoing from PC3) near to the end of the process. Jigsaw Puzzle continuously exchanges data at a stable rate, 5 and 10 Mbit/s throughout processing. LU Decomposition also exchanges data continuously, but its throughput behaves in an on-off manner, and varies within a short period. Task Scheduling intermittently exchanges large amounts of data throughout processing.

Figure 3 shows the cumulative probability of the throughput of each application for one instance of the 20 experiments. For each application, the distributions of incoming and outgoing traffic are similar. For N Queen, there is no traffic or less than 1 Kbit/s for more than 95 % of the processing time. And the second most frequent rate of throughput is near maximum. For Jigsaw Puzzle, the peak

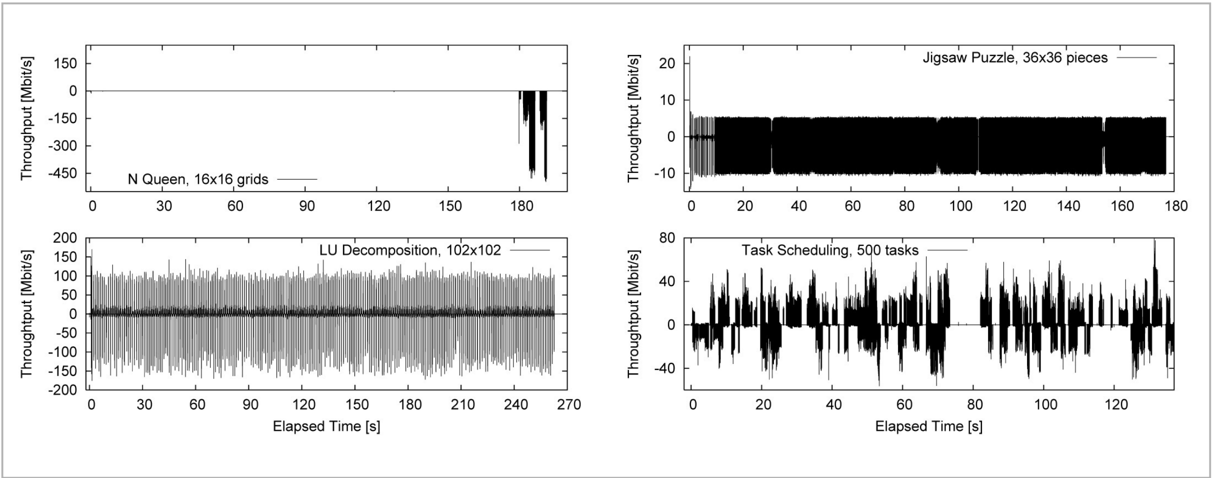


Fig.2 *Fluctuation of throughput in each of applications. X-axis is the elapsed time in seconds after that the application starts. Throughput of 10-ms average is on y-axis. Positive values on y-axis indicate traffic incoming to PC3 while negative values indicate outgoing traffic.*

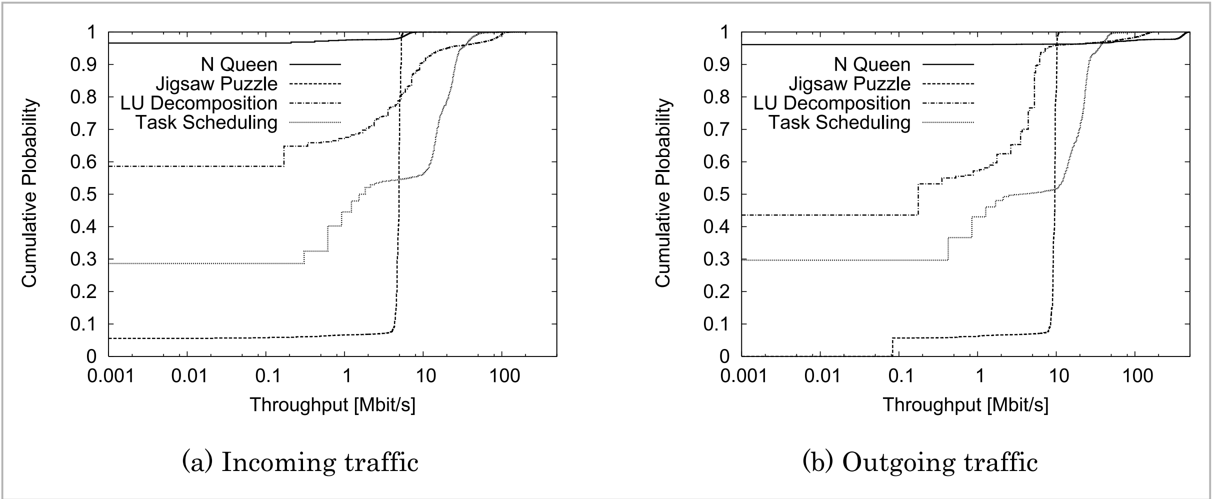


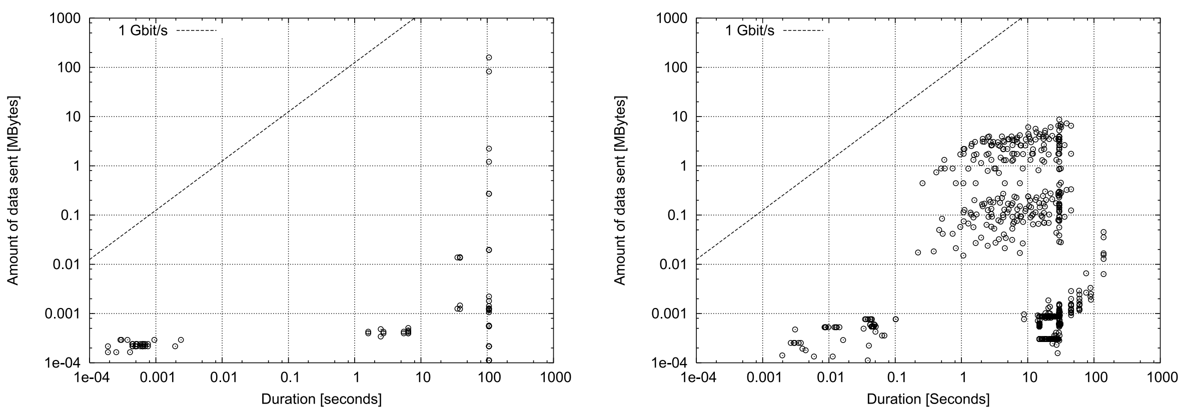
Fig.3 *Cumulative probability of throughput for each application.*

values are about 5 Mbit/s for incoming traffic and about 10 Mbit/s for outgoing. For LU Decomposition, for both incoming and outgoing traffic, about half of the throughput is less than 1 Kbit/s and the rest varies. Task Scheduling produces a variety of levels of throughput, with more than 40% of it being between 20 and 50 Mbit/s.

We also analyzed the feature of flows consisting of traffic generated by each of the applications. All the applications use only TCP for task communication. We therefore defined a flow as a set of packets transmitted via a TCP connection by direction, beginning with a SYN flag and terminating with a FIN flag. Multiple TCP connections were often established in parallel as a result of the processing taking place in each application.

Figure 4 shows the amount of transferred data and duration of each of the flows for N Queen and Task Scheduling for one instance of the 20 experiments, respectively. Jigsaw Puzzle and LU Decomposition show similar features to those of N Queen, as described in Figure 4 (a) and which presumably relates to the fact that those applications are all categorized as task-framing types.

There is a much smaller number of flows for N Queen than Task Scheduling. Some flows last for less than 10 ms, some for about 10 s, and others from start to finish. The long-lived flows transmit various amounts of data



(a) N Queen (b) Task Scheduling

Fig.4 *The duration and amount of transmitted data in each flow generated in N Queen and Task Scheduling. Line is boundary of plots and is equivalent to 1 Gbit/s.*

from 100 bytes to more than 100 Mbytes. The huge amount of data transferred by N Queen must be carried by these long-lived flows.

As shown in Figure 4 (b), there are a large number of flows of varying duration for Task Scheduling, which transmits various amounts of data.

From the analyses of application traffic features, we found the follows that:

- applications show distinct traffic patterns, and,
- although individual applications respond to increase their completion time in the case of large scale of problems to be solved, their traffic fluctuations are similar among the simple and complex problems.

In addition, from the traffic pattern of each application, we expect that N Queen achieves the good application-level performance when its data is transmitted through a sufficiently large bandwidth path even if there is a large RTT with the path, that Jigsaw Puzzle slightly affects the other application traffic when its traffic is multiplexed with other applications because of its low throughput, that, conversely, LU Decomposition significantly effects on the other application traffic in case of multiplexing, because its on-off traffic pattern, and that Task Scheduling shows the effectiveness of traffic multiplexing because of its intermittent traffic pattern.

5 Influences of network properties on application-level performance

To clarify the effect of the network properties on the application-level performance, we examined the characteristics of the completion time for each application by imposing either a long RTT, or a bottleneck link with a narrow bandwidth.

5.1 Impact of large round-trip time

We investigated the influence of a long RTT on the completion time for each application running on PC1 through PC4, as described in Figure 1. In our experiments, 1 to 32 ms latencies were inserted into traffic passing the bottleneck link both ways using the network emulator shown in Figure 1. The link bandwidth was configured to a value of 1 Gbit/s so that the focus was on the impact of the RTT on the performance of the applications. TCP socket buffers of 16 and 128 KB in length were used in N Queen and Jigsaw Puzzle, 128 and 1024 KB in LU Decomposition, and 64 and 256 KB in Task Scheduling. Figure 5 shows the completion time for each application influenced by the RTT. The completion times are normalized by the times obtained when these is no latency. Each completion time is the average of the results of 20 experiments.

We found that the characteristics of the completion time for both N Queen and Jigsaw Puzzle were almost the same in the case of both the 16 and 128 KB TCP socket buffers. The results for using a 128 KB socket buffer are therefore omitted from Figure 5.

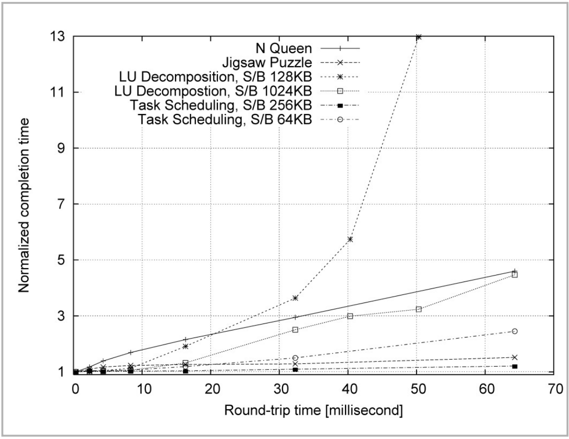


Fig.5 *Completion time influenced by RTT. Application-level performance deteriorates as RTT increases*

For all the applications, the application-level performance deteriorates as the RTT increases. For LU Decomposition and Task Scheduling, the large socket buffer is very effective in reducing the completion time, probably because the average size of the TCP sending window is equal to the product of the average throughput and the RTT in successive sending of large amounts of data. In the following experiments we therefore used 1024 KB and 256 KB socket buffer for LU Decomposition and Task Scheduling, respectively. N Queen and LU Decomposition are still influenced by the RTTs despite the use of large socket buffers. And Jigsaw Puzzle is not influenced by the RTTs even though it does not use large socket buffer. To try to determine the cause of these differences, we analyzed the interval between consecutive packets and the total amounts of transmitted data.

Figure 6 (a) and (b) show the distribution of intervals between consecutive packets in flows generated between PC1 and PC3 for N Queen, Jigsaw Puzzle, respectively. The results for LU Decomposition are omitted because LU Decomposition showed the change of the distribution of packet intervals, similar to that of N Queen.

For N Queen (a), the distribution of packet intervals in the case of a short RTT (0.331 ms) shows that almost all the intervals are less than the RTT. This implies that almost all the packets are sent in succession. The packet intervals show two peak values for, one at about 0.012 ms corresponding to back-to-back data packets of 1500 bytes (more than 70 % of intervals), and the other at about 0.1 ms. In the case of a large RTT (32.3 ms), the number of intervals in the most bursty case (i.e., back-to-back packets) decreases to 60 % and the intervals near the RTT increase instead. This shift

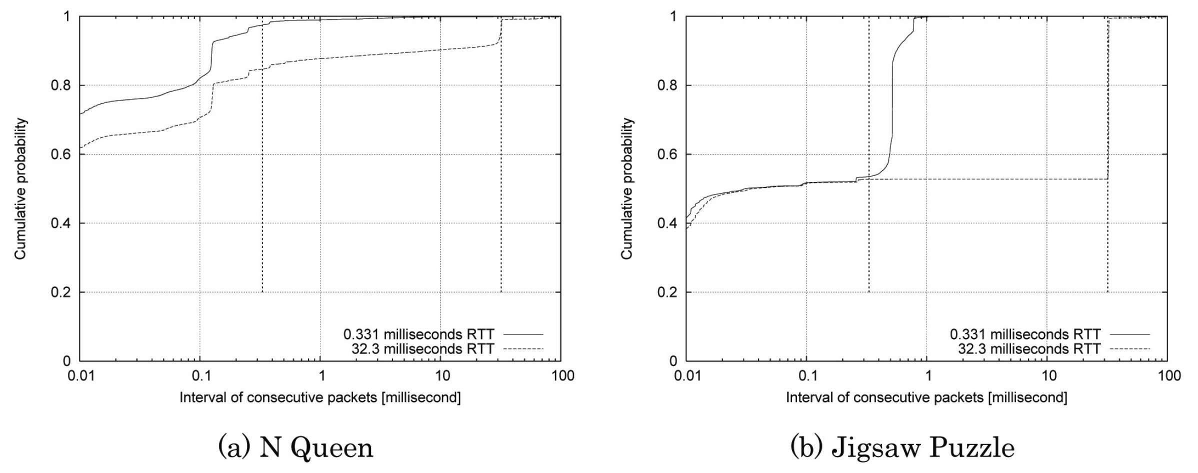


Fig.6 *Distribution of inter-packet gaps in flows generated by N Queen and Jigsaw Puzzle.*

Table 2 Amount of transmitted data in applications influenced by various round-trip times.

Application	RTT: 0.331 ms		RTT: 4.3 ms		RTT: 16.3 ms		RTT: 64.3 ms	
	packet	byte	packet	byte	packet	byte	packet	byte
N Queen	233K	242M	234K	242K	227K	238M	225K	237M
Jigsaw Puzzle	2372K	297M	992K	118M	293K	35.1M	84K	9.99M
LU Decomposition	674K	475M	716K	513M	739K	537M	748K	542M
Task Scheduling	823K	338M	898K	362M	720K	313M	690K	352M

indicates that a long RTT prevents rapid expansion of the sending window, resulting in a decrease in the throughput.

For Jigsaw Puzzle (b), the distribution of intervals also shows two peak values. One, which accounts for more than 40 %, is roughly close to the interval between back-to-back packets of 1500 bytes (0.012 ms), and the other is roughly close to the RTT (0.331 ms) for sending packets interactively. In the case of a 32.3-ms RTT, the data is still transmitted in a bursty manner, while the peak value corresponding to an RTT of 0.331 ms moves to a new RTT of 32.3 ms. The reason that Jigsaw Puzzle does not take so long to complete its processing when the RTT increases although this interactive communication is influenced by a large RTT may be because the total amount of data transmitted decreases as the RTT increases, as shown in Table 2.

The information on the sensitivity of application performance to a long RTT will be useful in selecting the appropriate path for different applications in multi-path environments.

5.2 Impact of limiting the bandwidth of the bottleneck link

We investigated the influence of limiting the bandwidth of the bottleneck link on the completion time of each application running on PC1 through PC4, as described in Figure 1. In the experiments, the bandwidth of the bottleneck link was varied from 80 Kbit/s to 1 Gbit/s using the network emulator in Figure 1. The RTT was the original short value without any additional latency so that the focus was on the impact of bandwidth restriction on the application performance.

Figure 7 shows that the completion time remained roughly the same (less than 1.1), even if the bandwidth is reduced before reaching a threshold (a gradual deterioration threshold) for the individual applications. Furthermore, the completion time increases abruptly, exceeding 1.5 if the bandwidth is reduced after reaching a threshold (a severe deterioration threshold) for each application. Table 3 shows the rangs of both gradual and severe deterioration threshold.

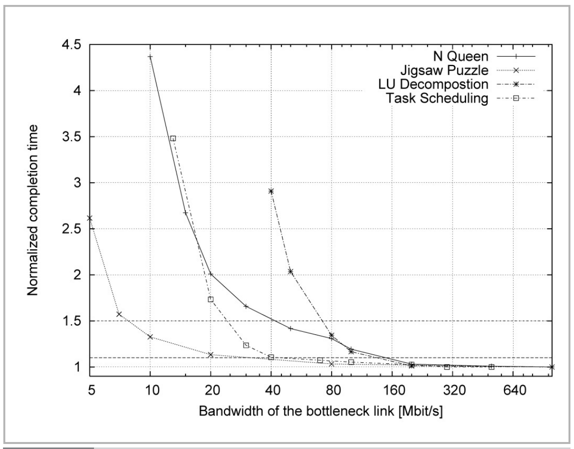


Fig.7 Completion time influenced by narrow bottleneck link.

Table 3 Bottleneck bandwidths at which completion time is deteriorated gradually or severely.

Application	Gradual deterioration	Severe deterioration
N Queen	140〜500 Mbit/s	< 40 Mbit/s
Jigsaw Puzzle	30〜200 Mbit/s	< 7 Mbit/s
LU Decomposition	140~200 Mbit/s	< 70 Mbit/s
Task Scheduling	40〜200 Mbit/s	< 25 Mbit/s

The information on the thresholds for limiting bandwidths will be useful in determining how limited amounts of network bandwidths should be allocated to various applications.

6 Conclusion

We investigated the network-related properties of some typical distributed applications, focusing on the influence of application traffic on the condition of network resources and, conversely, that of the condition of network resources on application-level performance.

We first analyzed the characteristics of traffic generated by applications classified as either task-framing or the work-flow types of distributed processing. We found that all the applications increased their completion time and the amounts of transmitted data as the scale of problems became larger while the relationship between the amount of transmitted data and the completion time depended heavily on the application.

We next analyzed how the performance of each application was affected by the condition of the network resources. It was found that the sensitivities of the completion time to the RTT and the bottleneck link bandwidth differed strongly depending on the application. Note that the availability of a large window in a TCP connection could mitigate the performance degradation caused by a long RTT in an application sending large amounts of data in succession.

Our goal is to develop an application-aware method of allocating network resources using the information on the network-related properties of different applications. Through the analyses, we conclude that

- the traffic engineering based on application-level performance is efficient, and
- the information on the impact of network conditions on application-level performance and the impact of application traffic on network conditions is vital for efficient traffic engineering with different network properties.

References

1 I. Foster and C. Kesselman, "The GRID Blueprint for a New Computing Infrastructure", Morgan Kaufmann Publishers, 1998.

2 I. Foster and C. Kesselman, and S. Tuecke, "The Anatomy of the Grid: Enabling Scalable Virtual Organizations", International Journal of Supercomputer Applications, Vol. 15, No. 3, pp. 200–222, 2001.

3 A. Elwalid, C. Jin, S. Low, and I. Widjaja, "MATE: MPLS adaptive traffic engineering", Proc. of the Infocom, pp. 1300–1309, 2001.

4 T. Guven, C. Kommareddy, R. La, M. Shayman, and S. Bhattacharjee, "Measurement Based Optimal Multi-path Routing", Proc. of the Infocom, 2004.

5 Kawahara, "An Adaptive Load Balancing Method for Multiple Paths Using Flow Statistics and Its Performance Analysis", IEICE Transactions on Communications, Vol. E87-B, No. 7, pp. 1993–2003, 2004.

6 N. S. V. Rao, "NetLets: End-To-End QoS Mechanisms for Distributed Computing in Wide-Area Networks Using Two-Paths", Proc. of the first International Conference on Internet Computing, pp. 475–478, 2001.

7 K. Aida and T. Osumi, "A Case Study in Running a Parallel Branch and Bound Applications", Proc. of the 2005 International Symposium on Application and the Internet (SAINT2005), pp. 164–173, 2005.

8 A. Plaat, H. E. Bal, and R. F. H. Hofman, "Sensitivity of Parallel Applications to Large Differences in Bandwidth and Latency in Two-Layer Interconnects", Proc. of the 5th High Performance Computer Architecture, pp. 244–253, 1999.

9 The ACM International Collegiate Programming Contest Japan Domestics, Problem C, Jigsaw Puzzle for Computers, http://www.fun.ac.jp/icpc/domestic_problems.html, 2001.

10 D. H. Bailey, E. Barszcz, J. T. Barton, D. S. Browning, R. L. Carter, L. Dagurm, R. A. Fatoohi, P. O. Federickson, T. A. lasinski, R. S. Schreiber, H. D. Simon, V. Venkatakrishnan, and S. K. Weeratunga, “The NAS Parallel Benchmarks”, International Journal of Supercomputer Applications, Vol. 5, No 3, pp. 66–73, 1991.

11 T. Tobita and H. Kasahara., “A standard task graph set for fair evaluation of multiprocessor scheduling algorithms”, Journal of Scheduling, Vol. 5, issue 5, pp. 379–394, 2002.

12 R. Buyya, “High Performance Cluster Computing: Programming and Applications”, Vol. 2., Prentice Hall PTR, 1999.

13 Empirix Packet Sphere, http://www.empirix.com/

KITATSUJI Yoshinori

Expert Researcher, Kitakyusyu JGNⅡ Research Center, Collaborative Research Management Department

Routing of Computer Communication Networks

***YAMAZAKI Katsuyuki**, Ph.D.*

Guest Researcher, Kitakyusyu JGNⅡ Research Center, Collaborative Research Management Department (KDDI R&D Laboratories, Inc.)

Internet QoS, Networking

***OIE Yuji**, Dr. Eng.*

JGNⅡ Project Leader, Collaborative Research Management Department

Information Network Engineering

TAGASHIRA Hideki

Guest Researcher, Kitakyusyu JGNⅡ Research Center, Collaborative Research Management Department (Kyushu Electric Power Co., Inc.)

Performance Measurement of Computer Communication Networks

***TSURU Masato**, Dr. Eng.*

Expert Researcher, Kitakyusyu JGNⅡ Research Center, Collaborative Research Management Department

Performance Measurement, Modeling and Analysis of Computer