A Comparative Study of Bidirectional Ring and Crossbar Interconnection Networks*

Hitoshi Oi and N. Ranganathan†
Department of Computer Science and Engineering,
University of South Florida, Tampa, FL 33620

Abstract

For distributed shared memory multiprocessors, the choice and the design of interconnection networks have a significant impact on their performance. Bidirectional ring networks are considered to be physically fast due to their simple structure, but topologically slow since their communication latency grows at $O(N)$. In this paper, we will present a quantitative measure to the question of how physically fast a bidirectional ring has to be to overcome its topologically slow communication latency by comparing it to the crossbar, an interconnection network that has opposite characteristics to the ring network. A hybrid method, in which workload parameters extracted from memory traces are given to an analytical model is used for performance evaluation. Our study shows that for a configuration of 32 nodes, the bidirectional ring outperforms the crossbar by 37% on the average of four parallel applications.

Keywords: *Interconnection networks, distributed shared memory multiprocessor, performance evaluation, slotted ring, crossbar.*

1 Introduction

A distributed shared memory (DSM) multiprocessor provides a view of a globally shared address space by sending coherence protocol messages between processing elements. Therefore, the interconnection network affects the performance of a DSM multiprocessor significantly.

Ring networks have the advantages of (1) fixed node degree (modular expandability), (2) simple network interface structure (fast operation speed) and (3) low wiring complexity (fast transmission speed). On the other hand, the main disadvantage of the ring network is its communication latency growth rate of $O(N)$, where N is the number of nodes. However, when non-local misses are likely to be destined to neighboring PE's, this disadvantage can be alleviated by using a bidirectional ring [1, 2]. Crossbar switches have opposite characteristics: they can connect any pair of nodes by one hop, but the transmission speed is expected to be much slower due to wiring and circuit complexity [3]. In this paper, we will present a quantitative measure to the question of how physically fast a bidirectional ring has to be to overcome its topologically slow communication latency, by comparing its performance to that of the crossbar switch.

This paper is organized as follows. The rest of this section introduces the past studies on the comparisons of the interconnection networks for multiprocessors. The architectures of multiprocessors that will be assumed in our study are presented in Section 2. Our methodology for evaluating the performance and the profile of the applications used in the performance evaluation are described in Section 3. The comparison of the bidirectional ring and the crossbar by estimated execution times is presented in Section 4. Some conclusions are provided in Section 5.

1.1 Related Work

Ravindran and Stumm compared the performance of multiprocessors using hierarchical ring and mesh networks [4]. The miss latency was used for performance comparison.

*This research is supported in part by a National Science Foundation Grant No. CDA-9522265.
†Email: {oi, ranganat}@csee.usf.edu

Their study mainly showed maximum number of nodes at which the hierarchical ring outperformed the mesh network. Since they assumed a unidirectional ring, nearest-neighbor communication pattern was not taken into account.

Barroso and Dubois evaluated the performance of the slotted ring multiprocessor [5]. They investigated the effect of the design choices such as coherence protocols (snoopy, linked list and full-map directory) and processor speed, and compared their unidirectional ring architecture with the split-transaction bus.

Lang et al studied the effective bandwidth of the crossbar switches, and compared it to that of the multiple-bus interconnection network using parametric simulations [3]. They assumed the dance-hall UMA architecture (processors and memories are different side of interconnection network).

2 Architecture Model

In this section, we describe the architecture of the DSM multiprocessors using the bidirectional ring and the crossbar that will be assumed in this paper. Each processing element (PE) consists of a processor, a memory unit with directory entries, a cache, and a network interface (Figure 1). PE's are connected by either a bidirectional ring or a crossbar switch network.

It is assumed that both architectures are based on the CC-NUMA model with directory based cache coherency protocol. The memory unit within a PE is a part of the globally shared memory and is associated with directory entries corresponding to the part of global address space assigned to the PE. A cache miss is responded either by the home node (the PE that is assigned the portion of global address of the accessed data) or by the owner node (the PE that owns a modified copy of the requested data in the cache).

For a write access, invalidation scheme is used. On the bidirectional ring, a single invalidation message is broadcast by passing it all the way through the ring. On the other hand, the crossbar network is assumed to have the multicast functionality to send invalidate messages simultaneously to all the PE's that have copies of the block.

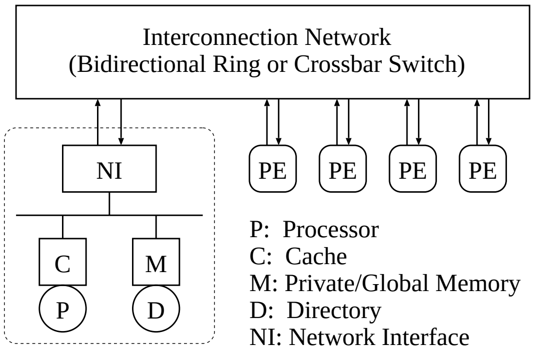

Figure 1: DSM Multiprocessor Architecture

The parameters that define the configuration of the system are listed in Table 1. All the timing parameters are represented in terms of processor clock cycles. In this paper, we consider small to medium scale multiprocessors, and hence we chose $N = 16$, 32 and 64. Both bidirectional ring and crossbar are assumed to be flat (one level). On the bidirectional ring, a header message is divided into two packets, and it takes tr to transfer a packet between adjacent PE's. This assumption of tr is to compensate that a node on the bidirectional ring has two links for each direction. On the bidirectional ring, each link connects a pair of adjacent PE's with a minimum wire length. Therefore, it is not difficult to keep tr constant for larger N. Thus, we use the same $tr = 1, \ldots, 5$ for all N.

ts is the time to transfer a header message between any pair of PE's on the crossbar network. We have chosen ts as follows: First, we take two instances of shared memory multiprocessors using crossbar switches, and look at their relative speed between the processor and the crossbar, the dimensional size and the data path width of the crossbar, and the word length of the processor. Exemplar S-Class of Hewlett Packard uses 180MHz 64-bits processors and 8×8 crossbars operating at 120MHz with 64-bits datapath [6]. CP-PACS developed at University of Tsukuba uses 180MHz 64-bits processors and 17×17 crossbar switches with data

Symbol & Description		Value
N	Number of PE	16, 32, 64
ts	Packet Transfer Time (Crossbar Switch)	4, 8, 16, 32, 64
tr	Packet Transfer Time (Bidirectional Ring)	1 - 5
tm	Memory Access Time	50
tc	Protocol Handling Time	10
dp	Data Mesg. Packet Length	4

Table 1: System Parameters (Timing parameters are in processor clock cycles)

bandwidth of 300MB/sec per node [7]. Using the above definition, Exemplar and CP-PACS have $ts = 1.5$ and $ts = 8$ respectively. Taking these values into consideration, we have chosen $ts = 4$ and 8 for $N = 16$. Next, we consider ts for larger N (32 and 64). The delay of crossbar switch grows at $O(N^2)$ due to the length of wire and number of nodes [3]. Thus, we use $ts = 16$ and $ts = 64$ for $N = 32$ and 64, respectively. Unlike UMA (including Exemplar) in which the delay of crossbar is dominant for the network latency, both the wire connection between PE's and crossbar, which is considered to be $O(N)$, and the delay of crossbar affect the latency of the network in NUMA architecture. Hence, we also use lower values of $ts = 8$ and $ts = 16$ and 32 for $N = 32$ and 64, respectively.

A data message is assumed to be four times longer than a header message ($dp = 4$). This value is chosen from the cache block size (16Bytes for 32bit processors) assumed when the trace files were collected. On the crossbar, a data message is transmitted in a contiguous $4ts$ time period, while on the bidirectional ring a data message is divided into multiple packets and they are sent in (possibly) uncontiguous slots. We assume the slotted ring configuration for the bidirectional ring.

3 Performance Model

The methods that have been used for performance evaluation of computer systems include analytical models ([8]), parametric simulations ([4]), trace-driven simulations ([9]) and execution-driven simulations ([10]). In this paper, we use the hybrid approach by Barroso and Dubois [5] by extending it to the bidirectional ring and the crossbar. Below we briefly describe the derivation of execution time using the hybrid approach.

The execution time (ET) in clock cycles is given by

$$ET = Inst + DataAccess \qquad (1)$$

where $Inst$ is instruction fetches. With assumption of sequential consistency and both an instruction fetch and a data access that is a cache hit take one clock cycle, (1) becomes

$$ET = Refs + Miss * MissLatency \qquad (2)$$

where $Refs$, $Miss$ and $MissLatency$ are the total number of references (both instructions and data), the total number of misses and the average miss latency, respectively. $MissLatency$ is defined to consist of T, the time to transmit a message (either request or data) to the network, P, the propagation delay of a message to reach the destination on the network, M, the time to access the memory, and C, the time of coherence protocol handling. These components of miss latency appear different number of times per miss depending on the access mode and the state of the memory block.

P is constant ts on the crossbar while it is the average message traversal length extracted from the traces on the bidirectional ring. C is assumed to be constant tc. The values of parameters T and M are the functions of the utilizations of the resources (network and memory). The utilization of the bidirectional ring R_u is

$$R_u = \frac{tr * (\sum HdTrv + dp * \sum DtTrv)}{ET}$$

where $\sum HdTrv$ and $\sum DtTrv$ are the sums of header and data message traversals, respectively. On the other hand, the utilization of the crossbar switch is

$$SW_u = \frac{ts * (\sum HdTx + dp * \sum HdTx)}{ET}$$

where $\sum HdTx$ and $\sum DtTx$ are the the numbers of header and data message transmissions. This is because on the crossbar switch network, a message only takes one hop to reach any destination PE. These utilization parameters are combined with the M/G/1 queue model to estimate the execution time of applications on both architectures (see [11] for the details of T on ring networks).

Thus, ET itself is a function of ET.

$$ET = Refs + Miss * MissLatency(ET)$$

The derivation of ET is iterated until it converges to a tolerant level ($< 0.1\%$). We use two metrics to evaluate the performance of two architectures. One is the ratio of execution times, namely, the execution time on the crossbar divided by the execution time on the bidirectional ring. Another metric is the scalability (speed up), which is obtained by dividing the execution time of the uniprocessor by that of the multiprocessor.

3.1 Trace Files

The profile of the memory traces we use in our experiments are listed in Table 2. These trace files are from the MIT trace set [12], and are obtained from the ftp server of the TraceBase project at the New Mexico State University [13]. FFT, Simple and Weather were written in fortran and traces of these applications were derived using the postmortem method, a technique that generates a parallel trace from a uniprocessor execution. Speech was written in the programming language Mul-T, and its trace file was collected by inserting instrumentation codes into the program by the compiler.

The workload parameters of each application were extracted from the trace file as follows. Each trace file is fed to a cache/directory simulator developed at the Laboratory for Computer Science of MIT[1] (the cache size of 256KB and the block size of 16B were used), and statistical information of the applications such as hit/miss, read/write, clean/dirty was collected. These trace files were collected on a

[1]This simulator was also provided by the TraceBase ftp server.

Application	Refs (Instr)	N	Miss Rate	Write Prob	Local Miss
FFT	7.44M (3.11M)	16	0.030	0.354	0.525
		32	0.048	0.226	0.509
		64	0.065	0.166	0.501
Simple	27.03M (11.59M)	16	0.043	0.114	0.240
		32	0.050	0.104	0.206
		64	0.072	0.089	0.114
Weather	31.76M (13.64M)	16	0.018	0.207	0.425
		32	0.020	0.196	0.397
		64	0.035	0.160	0.239
Speech	22.55M (11.77M)	16	0.048	0.374	0.672
		32	0.051	0.362	0.661
		64	0.053	0.348	0.656

Table 2: Profile of Trace Files

64 processor system. To extract data for the case of 16 (32) processors, access traces from $4i$ to $4i+3$ (from $2i$ to $2i+1$) were given to i-th cache, where $i = 0, \ldots, 15$ ($i = 0, \ldots, 31$). It is assumed that all the instruction fetches are cache hits.

In addition to the original functionality, the simulator was modified to collect performance measures regarding the communication of coherence protocol. One such measure is the ratio of whether the home node of a miss to a shared block is local or non-local (shown in Table 2). This ratio can be an index to the amount of communication since if the home node is local, no read request message will be transmitted on the network (when the block is clean). We assume that the first PE that accesses a block is the home node of the block. On FFT and Speech, more than half of accesses to shared data are to local addresses, while on Simple and Weather the locality of shared access is quite low.

Another performance measure collected by the modified simulator was the communication distance, the number of links a message has to traverse on the bidirectional ring. This communication distance affects the performance of the bidirectional ring significantly in two ways: First the longer the distance the longer the propagation delay (higher P). Second the longer the distance the higher utilization of the ring (higher T). Figures 2 show the distribution of request - home node distance on the bidirectional ring for $N = 64$, where the difference of communication pattern among applications is observed most clearly. FFT and Speech exhibit all-to-all communication pattern while Weather and Simple have a certain level of nearest neighbor communication patterns. Especially for $N = 64$ on Weather, 30% of non-local misses are accesses to adjacent nodes. For $N = 16$ and 32, the degree of nearest neighbor communication pattern of weather and simple are weaker than the case of $N = 64$. Although the number of applications is not large, these applications exhibit variations in their workload parameters.

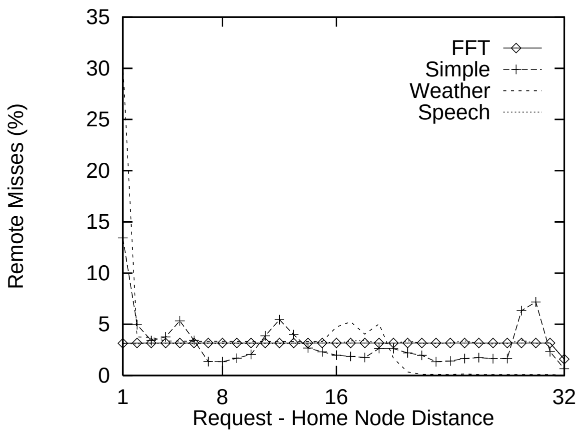

Figure 2: Home Node Distribution ($N = 64$)

4 Bidirectional Ring versus Crossbar Evaluation

In this section, we present comparisons of the bidirectional ring and the crossbar using the model and the traces presented in the previous section (Figures in this section are placed after the references).

FFT (Figure 3) has all-to-all communication pattern (Figures 2), which is an advantage to the crossbar over the bidirectional ring. The speed up for larger N is relatively small among four applications (Figure 4). The reasons for this small speed up are, that the execution of FFT on the uniprocessor is very fast (processor busy rate is 92%) and that shared miss rates increase and write hit rate decreases for larger N. Even with very fast link, we cannot expect much performance improvement on the bidirectional ring beyond $N = 16$. On the crossbar, situation is much worse: the performance degrades for $N > 16$ even with very fast switch devices.

Simple (Figure 5) has relatively high miss rate, especially in its read accesses, and the local miss rate is quite low. However, it still exhibits relatively high speed up among the applications used in this study (Figure 6). The first reason is that its execution on the uniprocessor has high miss rate, and the processor busy rate is for only 27%. Simple has a certain level of nearest communication pattern, which is beneficial to the bidirectional ring, and it becomes stronger for larger N. As a result, we can still expect a speed up of 33.6% for $N = 32 \rightarrow 64$ if the ring speed is fast enough. On the other hand, the nearest communication pattern of Simple does not benefit to the crossbar. Even with an optimistic assumption of communication latency growth rate ($O(N)$), its performance degrades for $N = 32 \rightarrow 64$.

Weather (Figure 7) has relatively low miss rate, and large fraction of misses are destined to local memory. Thus, the effect of interconnection network is small and both the crossbar and the bidirectional ring achieve nearly linear speed up for $N = 16 \rightarrow 32$ (Figure 8). However, for $N = 64$, local miss rate decreases (from 39.7% to 23.9%) and miss rate increases by 75%. The performance of the crossbar decreases for $N = 32 \rightarrow 64$ even with a very fast switches ($ts = 16$). On the bidirectional ring, we can still expect a speed up of 2.7% ($tr = 5$) to 35.5% ($tr = 1$). In addition to the fast link speed of the bidirectional ring, the nearest communication pattern of Weather (Figure 2) that becomes stronger for larger N is considered to be the source of this speed up.

The miss rate and local miss rate of Speech are constant over increasing N. Therefore, the growth of communication latency of both interconnection networks have strong effect on their performance. In addition, since Speech has the all-to-all communication pattern, relative performance of Speech is merely affected by the speed of both networks (Figure 9). For $N = 32 \to 64$, the bidirectional ring still provides some level of speed up (19% to 64%), while the performance of the crossbar can be either increased by 21.8% or decreased by 44% depending on the speed of the switch (Figure 10). The average (geometric mean) of relative performance and speed up of both networks are shown in Figures 11 and 12. For $N = 16$, the bidirectional ring is 8% faster than the crossbar with $tr = 1$ and $ts = 4$. For $N = 32$ and 64, the bidirectional ring achieves 32% and 77% better performance than the crossbar even if the communication growth rate of the crossbar is $O(N)$.

Application	N	16		32		64		
	ts	4	8	8	16	16	32	64
FFT		1	2	1	3	2	4	> 5
Simple		1	2	2	3	2	> 5	> 5
Weather		1	3	3	> 5	> 5	> 5	> 5
Speech		1	3	2	4	2	4	> 5
Average		1	2	2	4	2	> 5	> 5

Table 3: Ring Speed to Outperform Crossbar

The slowest ring speed for the bidirectional ring to outperform the crossbar are shown in Table 3. For $N = 16$, we need a very fast ring that can operate at the same speed as the processor clock ($tr = 1$) to outperform a very fast implementation of a crossbar switch network ($ts = 4$). If the speed of a crossbar switch is moderate, a ring that operates at half the speed of the processor clock is sufficient. For $N = 32$, a bidirectional ring with half and one forth the speed of the processor clock suffice to outperform the crossbar switch network with very fast and moderate ($ts = 8$ and 16) speed respectively. For $N = 64$, it is much easier for a bidirectional ring to outperform a crossbar switch network due to the latter' physically slow operating speed. However, although it outperforms the crossbar, without sufficiently fast ring speed, we cannot expect speed up on the bidirectional ring. For example, on the average of four applications, we have speed up of 37% on the bidirectional ring with $tr = 1$ when N is increased from 32 to 64. On the other hand, if $tr = 5$, which is sufficient to outperform a moderately fast crossbar, the speed up is only 2.5% for $N = 32 \to 64$.

5 Conclusions

In this paper, we have evaluated the performance of the bidirectional ring by comparing it to the performance of the crossbar network. We used a hybrid evaluation method which is a combination of an analytical model and workload parameters extracted from the memory traces of four parallel applications. Our study indicates that for a 16 nodes configuration, which is a typical size of crossbar switches used in current multiprocessors, both architecture achieve similar performance for the link speed we have assumed. For a 32 nodes configuration, the bidirectional ring outperforms the crossbar by 32% on the average of four application programs, with an optimistic assumption that the growth rate of the communication latency of the crossbar is suppressed to $O(N)$. For a 64 nodes configuration, we can still expect speed up on the bidirectional ring (37% on the average) while the performance of the crossbar decreases from that of a 32 nodes configuration.

Further investigations could include analysis of dynamic behavior of both networks (such as hot spot contention) using execution-driven simulations and the hierarchical network models.

References

[1] Hitoshi Oi and N. Ranganathan, "Performance Analysis of the Bidirectional Ring-Based Multiprocessor", in *Proceedings of the ISCA 10th International Conference on Parallel and Distributed Computing Systems*, 397–400, October 1997.

[2] Hitoshi Oi and N. Ranganathan, "Effect of Message Length and Processor Speed on the Performance of the Bidirectional Ring-Based Multiprocessor", in *Proceedings on the International Conference on Computer Design*, 267–272, October 1997.

[3] T. Lang, M. Valero and I. Alegre, "Bandwidth of Crossbar and Multiple-Bus Connection for Multiprocessors", *Transactions on Computers*, IEEE, Vol. c31, No. 12, 1227–1234, December 1982.

[4] G. Ravindran and M. Stumm, "A Performance Comparison of Hierarchical Ring- and Mesh-connected Multiprocessor Networks", in *Proceedings of International Symposium on High Performance Computer Architecture*, 58–69, February 1997.

[5] L. A. Barroso and M. Dubois, *Performance Evaluation of the Slotted Ring Multiprocessor*, Transactions on Computers, IEEE, Vol. 44, No. 7, 878–890, July 1995.

[6] "Exemplar System Architecture", `http://www.hp.com/wsg/products/servers/exemplar/sx-class/exemplar2.html` , Hewlett Packard.

[7] T. Boku, et al, "Architecture of massively parallel processor CP-PACS", in *Proceedings of the 1997 2nd Aizu International Symposium on Parallel Algorithms/Architecture Synthesis*, 31–40, Fukushima, Japan, 1997.

[8] X. Zhang and Y. Yan, "Comparative Modeling and Evaluation of CC-NUMA and COMA on Hierarchical Ring Architecture", *Transactions on Parallel and Distributed Systems*, IEEE, Vol. 6, No. 12, 1316–1331, December 1995.

[9] K. Farkas, Z. Vranesi and M. Stumn, "Scalable Cache Consistency for Hierarchically Structured Multiprocessors", *Journal of Supercomputing*, Vol. 8, 345–369, 1995.

[10] H. Davis, S. R. Goldschmidt and J. Hennessy, "Multiprocessor Simulation and Tracing Using Tango", in *Proceedings of the 1991 International Conference on Parallel Processing*, Vol. II, 99–107, 1991.

[11] L. Bhuyan, D. Ghosal and Q. Yang, "Approximate Analysis of Single and Multiple Ring Networks", *Transactions on Computers*, IEEE, Vol. 38, No. 7, 1027–1040, July 1989.

[12] D. Chaiken et al., "Directory-Based Cache Coherence in Large-Scale Multiprocessors", *IEEE COMPUTER*, Vol. 23, No. 6, 49–58, September 1990.

[13] ftp://tracebase.nmsu.edu/pub/, Parallel Architecture Research Laboratory, New Mexico State University.

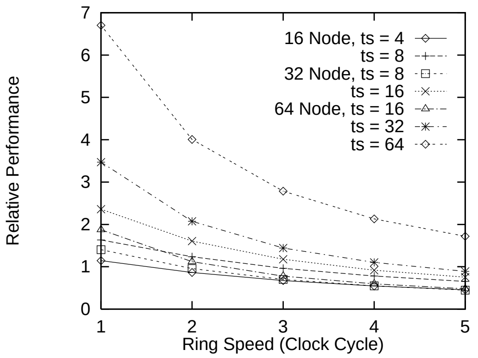

Figure 3: Relative Performance / FFT

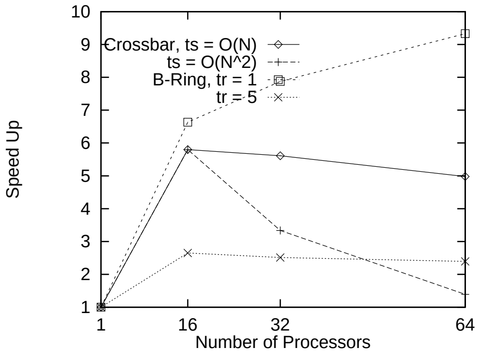

Figure 4: Speed Up / FFT

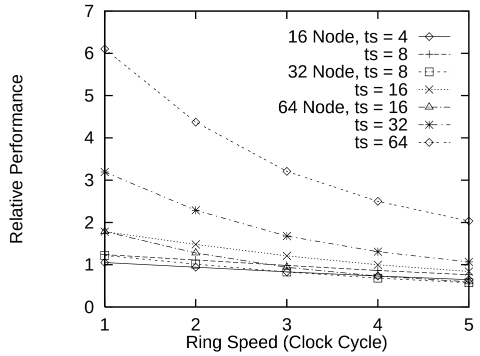

Figure 5: Relative Performance / Simple

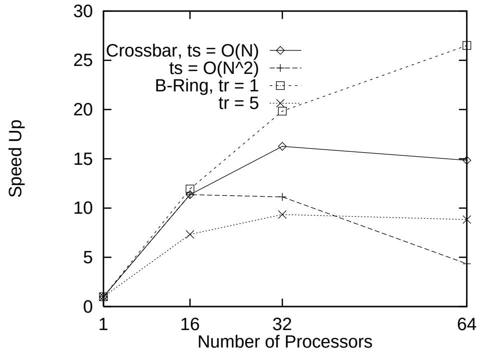

Figure 6: Speed Up / Simple

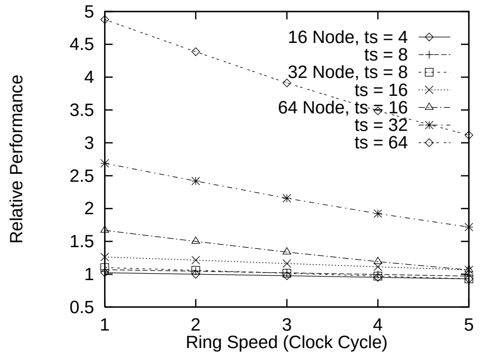

Figure 7: Relative Performance / Weather

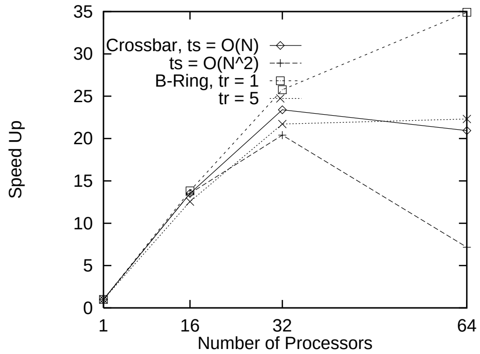

Figure 8: Speed Up / Weather

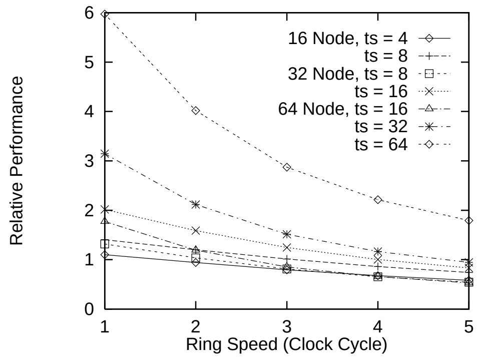

Figure 9: Relative Performance / Speech

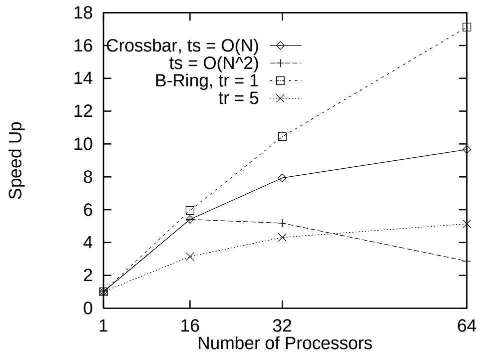

Figure 10: Speed Up / Speech

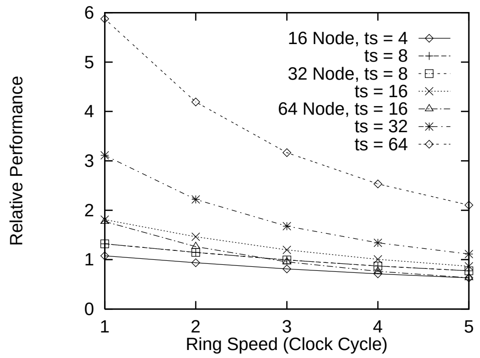

Figure 11: Relative Performance / Average

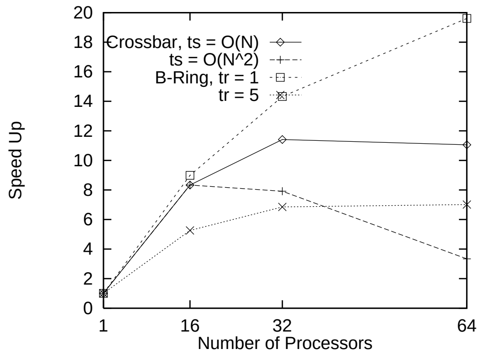

Figure 12: Speed Up / Average