Evaluation of Interconnection Network Performance under Heavy Non-uniform Loads

C. Izu[1], J. Miguel-Alonso[2], J.A. Gregorio[3]

[1]Department of Computer Science, The University of Adelaide, SA 5005 Australia
cruz@cs.adelaide.edu.au
[2]Dep. of Computer Architecture and Technology, The University of the Basque Country, 20080 San Sebastian, Spain
miguel@si.ehu.es
[3]Computer Architecture Research Group, Universidad de Cantabria, 39005 Santander, Spain
monaster@unican.es

Abstract. Many simulation-based performance studies of interconnection networks are carried out using synthetic workloads under the assumption of independent traffic sources. We show that this assumption, although may be useful for some traffic patterns, can lead to deceptive performance results for loads beyond saturation. Network throughput varies so much amongst the network nodes that average throughput does not reflect anymore network performance as a whole. We propose the utilization of burst synchronized traffic sources that better reflect the behavior of parallel applications at high loads. A performance study of a restrictive injection mechanism is used to illustrate the different results obtained using independent and non-independent traffic sources.

1 Introduction

Methods to evaluate the performance of an interconnection network range from the construction and measurement of its hardware prototype, to the utilization of overly simplified simulations. During the first stages of a new interconnection project, a fast simulation environment is critical, because it allows researchers to test and tune their design. Once a good tradeoff between expected performance and cost has been attained, the design can be rounded off using more detailed simulators. The evaluation of expensive prototypes goes just before the manufacture (and, again, evaluation) of the final product. In all these stages, evaluation has to be done using some kind of workload that resembles, with the higher possible fidelity, the actual workload that will be processed by the final network.

For practical reasons, most studies are carried out using synthetic workloads, running a simulator for a large number of cycles (simulated time) to get performance re-

This work has been done with the support of the Ministerio de Educación y Ciencia, Spain, under grants TIN2004-07440-C02-01 and TIN2004-07440-C02-02, and also by the Diputación Foral de Gipuzkoa under grant OF-846/2004.

sults with the network in steady state. Although this may not be realistic, we consider the obtained results as indicators of the level of performance the network could provide under real conditions. For some SPLASH applications such as Radix or LU, it has been shown to be a reasonable approach [10].

A synthetic workload is defined by three parameters: the injection process, the spatial traffic pattern and the message size [4]. This can be done in a per-node basis, although very often all nodes share the same behavior. The spatial pattern determines the distribution of destinations for each source node. The injection process determines the temporal distribution (in other words, when a packet is generated). The size distribution determines the message length.

Traffic patterns include permutations such as bit-reversal or matrix transpose, uniform (also called random) and hot-spot. Each of them represents a worst-case scenario: uniform has no locality, permutations make an uneven use of resources, and hot-spot models nodes that receive a higher proportion of the traffic.

In general, we cannot assume that applications running on a parallel computer use fixed-size messages. However, networks often impose a maximum packet size and messages have to be segmented to fit in several of those packets. For this reason, most studies are done with fixed-size messages of 8-32 phits [2, 4, 10, 11]. In some cases, message length follows a bimodal distribution which reflects network workload for a cc-NUMA system [9]. In this study, we will limit our discussion and experiments to fixed-size packets, although conclusions are valid for other length distributions.

Regarding the injection process, nodes are “programmed” to inject packets using some probability distribution (*independently* of the others). Injection times usually follow a Poisson or Bernoulli distribution, which are smooth over large time intervals. Recent works added on-off models that better characterize the self-similarity of traffic in some applications [11]. These widely used workloads treat each node as an independent traffic source (ITS).

The purpose of this paper is to show that performance results obtained with ITS for non-uniform traffic patterns under heavy loads process are misleading because they do not reflect the way actual parallel applications make use of the communication subsystem: their processing nodes may advance tightly or loosely coupled, with all the possibilities in between *but they are never totally uncoupled.* To better illustrate this issue, we describe an experimental setup designed to evaluate the impact on network performance of a restrictive injection mechanism, and we compare the results obtained using ITS with those obtained using burst-synchronized traffic (BTS).

The rest of the paper is organized as follows. Section 2 defines all the relevant parameters of our experimental setup. Section 3 presents, discuss and compare the disparate results obtained using independent and non-independent traffic sources. Section 4 summarizes the findings of this work.

2 Evaluation environment

In this section we define the experimental setup used to illustrate the impact of the choice of synthetic workload (focusing on the injection process) on the simulation results. First, we present the interconnection network as modeled for this study. Then we describe the context in which the injection process is analyzed, and the rest of simulation parameters.

2.1 The simulated network

For this work we use FSIN (Functional Simulator for Interconnection Networks), an in-house simulator, developed to simulate k-ary n-cube networks based on virtual cut-through (VCT) router architectures.

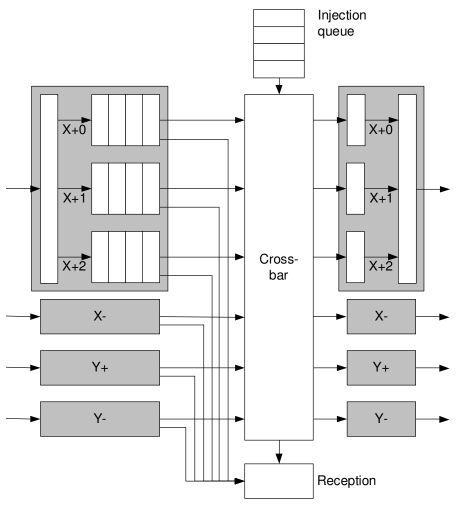

Fig. 1. Architecture of the adaptive VCT router used in the experiments.

Fig. 1 shows the architecture of an adaptive virtual cut-through (VCT) router. It uses three VCs per physical channel, to map a deadlock-free oblivious (dimension-order routing) sub-network and a minimal adaptive sub-network. Each VC has a buffer with capacity for 8 packets (128 phits). One of the VCs is used for the escape sub-network, relying on Bubble Flow Control (BFC) [10] to avoid deadlock in each dimension. The adaptive sub-network uses the other two virtual channels. Any blocked packet in the adaptive sub-network can resort to an escape path to break a potential deadlock cycle [5]. Such combination provides low-cost, deadlock-free adaptive routing.

In order to reduce the number of figures and better focus our discussions, in this paper we show results for a 32x32 tori. However, conclusions are valid for other network configurations.

2.2 The evaluation context

The choice of synthetic workload has a definite influence on any kind of performance experiment we may carry out. In order to be more specific, and to show this influence in a particular (but relevant) context, we describe an experimental setup that was used to study the advantages of implementing restrictive injection techniques to prevent network congestion

Congestion control mechanisms limit injection when the network reaches a given level of congestion, which can be estimated locally or globally. In this paper, we apply a local method called *in-transit-priority restriction* (IPR): for a given fraction P of cycles, priority is given to in-transit traffic; in those cycles, injection of a new packet is only allowed if it does not compete with packets already in the network. P may vary from 0 (no restriction) to 1 (absolute priority to in-transit traffic), although in this paper we will consider only the two extreme cases. This method is used in IBM's BG/L [1] and in the Alpha 21364 network [7]. A more detailed discussion of congestion control mechanisms can be found in [6].

When studying congestion, which appears at high loads, the main figure of merit is the maximum sustained throughput for loads beyond saturation. However, unexpected results lead us to examine throughput figures in more detail and identify a significant level of throughput unfairness, which rends average values to be meaningless. That finding lead us to redefine the temporal distribution of packets for the synthetic workloads used in the experiments, as reported in the next section.

2.3 Network workload

We have considered fixed-size packets of 16 phits. The traffic patterns used in the experiments are:

- **UN**: uniform traffic. Each node selects destinations randomly in a packet-by-packet basis.
- **TR**: transpose permutation. In a 2-D network, the node with coordinates (x, y) communicates with node (y, x).
- **SH**: perfect-shuffle permutation. The node with binary coordinates (a_{k-1}, a_{k-2}, ..., a_1, a_0) communicates with node (a_{k-2}, a_{k-3}, ..., a_0, a_{k-1})—i.e., rotate left 1 bit.

We use two types of injection processes:

- **Normal**: independent traffic sources, each one following a Bernoulli distribution with a parameter that depends on the applied load. This load is varied from 0 to 1 phit/cycle/node. The simulator runs for a warm-up period of 100,000 cycles, plus a measurement period of 100,000 cycles.
- **Burst-synchronized**: non-independent sources, to reflect the synchronized nature of parallel applications. The injection method is similar to that described in [2]. The same workload (*b* packets) is assigned to each source of traffic. A burst starts with an empty network. Nodes inject their *b* packets as fast as the network accepts them. The burst ends when all packets of all the traffic-generating nodes have been consumed. In the experiments, the simulator runs for 5 bursts of 1K packets.

3 Performance for independent and burst-synchronized traffic sources

Most interconnection network simulators model the processing nodes as ITS which are continuously generating packets. Network performance is reported using two figures: latency (time from packet generation until its delivery) and throughput, which is measured as the number of packets delivered in a given time interval divided by the interval length and the network size. In other words, this is the average load accepted by the network (i.e., the network throughput), which is expected to be even amongst the network nodes.

In this section we will show such expectation is incorrect for non-uniform loads once the network has reached saturation, and we will question the validness of average throughput as the figure of merit under heavy loads. The evaluation of the impact that a restrictive injection mechanism (IPR) has on the performance of an adaptive VCT torus network is provided only to illustrate this issue. We could have selected different router architecture, topology or congestion-control mechanism. It would not matter because conclusions would be the same: throughput under non-uniform patterns for loads beyond saturation varies widely amongst the network nodes.

3.1 Network performance under independent traffic sources

Fig. 2 represents network performance under three different traffic patterns (UN, TR and SH), with and without IPR, using a typical plot of average throughput versus applied load.

For the UN pattern, results show that utilization of a restrictive injection mechanism eliminates the throughput loss for loads beyond congestion. However, we cannot extend this conclusion to the permutations. In fact, results indicate that restrictive injection is counterproductive for TR and SH traffic under heavy load. This result was unexpected as non-uniform loads suffer more from congestion than UN, so we expect restrictive injection should be more effective, not less.

Another indicator of network performance is channel utilization: the higher the channel utilization, the better, because more resources are being productive. Let us focus on TR traffic without/with IPR. Fig. 2 indicates that, in saturation, throughput is higher without IPR. However, simulation results also indicate that channel utilization is higher with IPR. Which figure of merit is correct? How can channel utilization increase while delivering fewer packets? Does IPR increase performance, or not?

3.2 Discussion of performance figures under independent traffic sources

In [4], Dally & Towles suggest that performance of a network for a given traffic pattern in which the node injection rate is not the same for all nodes should be reported as the lowest injection rate that matches the desired workload.

Following this approach, in Table 1 we report maximum, minimum and average injection rates for the six configurations under study. Notice the vast differences between these values for the TR and SH permutations.

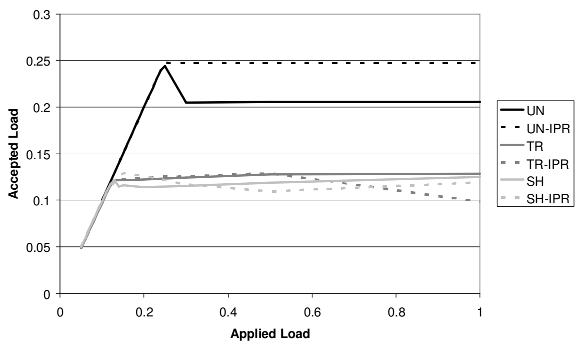

Fig. 2. Applied load vs. throughput (phits/cycle/node) for UN, TR and SH patterns, without/with IPR.

Table 1. Maximum, minimum and average network throughput (phits/cycle/node), for applied loads beyond saturation, for UN, TR and SH patterns, without/with IPR.

	UN		*TR*		*SH*	
	IPR off	*IPR on*	*IPR off*	*IPR on*	*IPR off*	*IPR on*
Max.	0,219	0,267	0,559	0,716	0,973	0,974
Min.	0,194	0,217	0,013	0,000	0,002	0,000
Avg.	0,205	0,243	0,132	0,098	0,125	0,119

Such large variations of throughput under TR and SH permutation patterns were also observed in other popular IN simulators such as Flexsim [12] and the chaos simulator [3] for a range of network designs. Dally & Towles [4] state that average and minimum rate differ in some routers due to their unfair design, citing the chaos router with prioritizes traffic in its internal queue over incoming or new packets as an example of that unfairness.

We should note that the time a packet awaits in an injection buffer before entering the network depends not only on the arbitration method, but also on the local router state. Under UN traffic, the network load is evenly distributed, so that all nodes have a similar view of network status and are able to inject packets at a similar rate. However, under non-uniform loads the degree of utilization of resources (buffers, output channels) may vary widely from one router to another. Therefore, at high loads, nodes connected to busy routers[1] have lower chances to inject their load than nodes in less used areas—a difference that causes wide variations in the number of packets injected by each node. In other words, the differences shown in Table 1 are not caused by an unfair routing or arbitration method, but by the fact that network resources are used unevenly by the applied workload, which is the case for all non-uniform loads.

Let us focus again on the TR permutation. In a 32x32 network, and assuming that all nodes inject at the same rate, the average distance packets traverse is 16.516. In fact, this is what the simulator reported when network load was below its saturation point. For those loads the map of packets injected per node is flat (except the nodes in the diagonal, which do not generate traffic for themselves), as shown in Fig. 3a.

[1] "Busy" routers are those that are traversed by numerous in-transit packets.

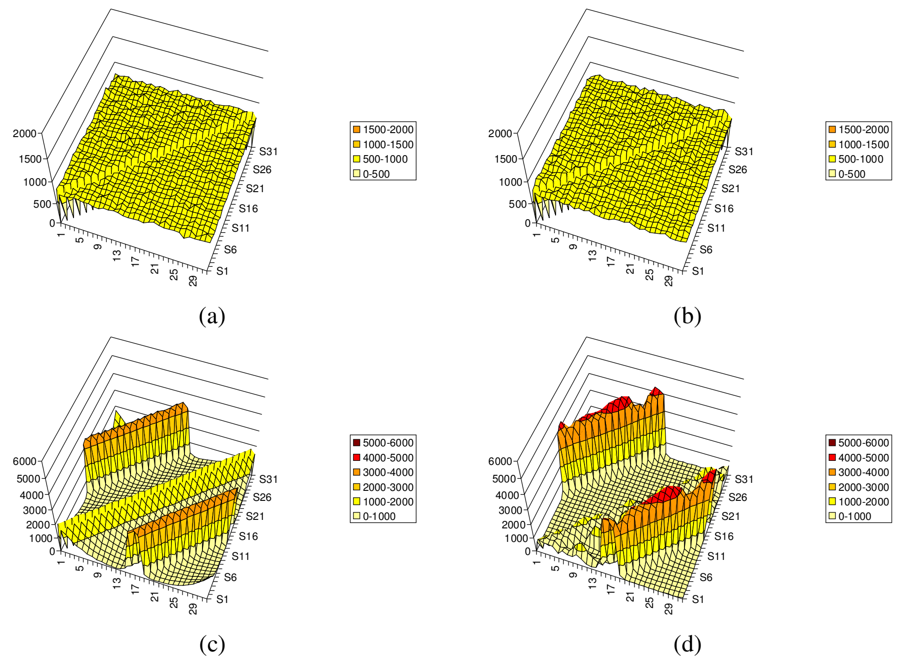

Fig. 3. Maps of injected packets for TR traffic. Each surface point (x, y) represents the number of packets a node with coordinates (x, y) injected in 100.000 cycles. (a) Below saturation, no IPR. (b) below saturation, IPR. (c) beyond saturation, no IPR. (d) beyond saturation, IPR.

The scenario changes drastically when saturation is reached. The simulator reflects this in a change in average distance (17.12) and in a very different map of injected packets (Fig. 3c). Note we have change nothing but the applied load. The problem is that some nodes can inject packets in their routers at very high rates, while others can hardly access the network because their routers devote most resources to passing-by packets. Fig. 3c shows that "lucky" nodes (those that have more opportunities to inject packets) are located close to the diagonal and in a pair of bands parallel to it. It gives the impression that the network is *unfair* for TR traffic.

We are interested to know why adding the IPR congestion control mechanism appears not to be beneficial in this scenario. Network response does not change below saturation (Fig. 3b) but for loads beyond saturation network unfairness is worst as shown in Fig. 3d: the "lucky" area close to the diagonal shrinks, and the two parallel bands are narrower and taller than without IPR[2]. As nodes in these bands are injecting packets addressed to distant destinations, the average distance rises up to 23.47. In other words, IPR magnifies the fairness problem.

[2] A digression of interest: we have a collection of nodes capable of injecting more than 4000 packets in 100.000 cycles, while some others are unable to inject a single one (they suffer starvation).. Any router that imposes restrictions to the injection of new packets may suffer from starvation. Although the adaptive router (without IPR) is starvation-free, it exhibits a high degree of unfairness under non-uniform traffic.

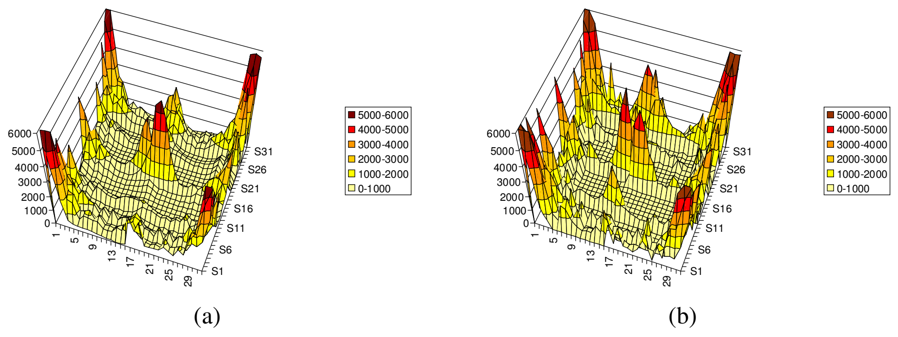

Fig. 4. Maps of injected packets for SH traffic beyond saturation. (a) IPR off. (b) IPR on.

For other non-uniform workloads results are similar. As an example, Fig. 4 shows the maps of injected packets for the SH pattern for loads beyond saturation, without and with IPR.

In conclusion, simulations report again that the implementation of a congestion-control mechanism is counterproductive for all traffic patterns under study—except for UN. Although IPR increases channel utilization, the number of packets delivered per cycle diminishes. This unexpected result is explained by the fact that network unfairness favours packets that travel longer paths.

3.3 Performance of the network under burst-synchronized traffic

The above conclusion could be considered correct as numerous previous works using this simulation methodology and workload. But luckily in this case we have several indicators (channel utilization, average distance and unfairness) that something is wrong. And what we think is wrong is the used workload.

Application processes are somehow *coupled*, because they work to perform a given task in a cooperative way. Most (if not all) applications use synchronization barriers, perform collective operations or use other mechanisms that make all the processes advance at a similar rate. It is true that worst-case performance for data exchanges is important (as shown in [8]) because it may halt progress of computation nodes, which are not able to perform additional operations, or communicate any further, until the data exchange has been completed. However, we cannot conceive a realistic scenario in which, *in the same parallel application*, a process is sending packets to its selected destination *ad infinitum* while other nodes do the same at a *much* smaller rate.

We consider burst-synchronized injection as described in section 2.3 to be a better alternative to model the communication structure of a parallel application at heavy loads. We have made a complete performance analysis similar to that reflected in Fig. 2, but using burst-synchronized traffic (BTS). Fig. 5 shows the time to complete 5 bursts of 1K packets for the six scenarios under consideration. For comparison purposes, Table 2 shows their throughput computed as the total workload delivered divided by the completion time. For this workload, maps of injected packets are meaningless (all nodes inject exactly the same number of packets), and the reported average distance traversed by packets is always the expected one[3]. Under burst-synchronized workload, the use of restricting injection policies is positive for the three traffic patterns: the time to deliver the 5 bursts of packets is lower with IPR than without it.

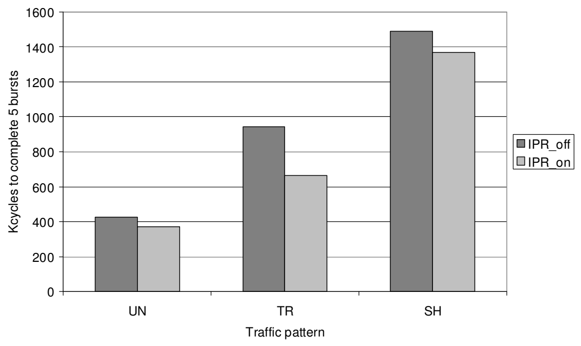

Fig. 5. Time to deliver 5 bursts of 1K packets for UN, TR and SH patterns, without/with IPR.

Table 2. Network throughput (phits/cycle/node) averaged for 5 bursts under UN, TR and SH patterns, without/with IPR.

UN		*TR*		*SH*	
IPR off	*IPR on*	*IPR off*	*IPR on*	*IPR off*	*IPR on*
0,192	0,220	0,087	0,123	0,055	0,060

The performance reported under BTS contradicts the results obtained under ITS. Which one is *correct*? As both are based on synthetic workloads, both are just approximations to the reality. But the behavior of real parallel applications at heavy loads is clearly closer to the burst-synchronized source model than to the independent source model. In fact, tests carried out with real applications show that this congestion control mechanism does improve throughput under heavy loads.

4 Conclusions and future work

Performance of interconnection networks is evaluated using a widely accepted set of synthetic workloads which model uniform, hot spot and traffic permutation patterns. Each node generates packets independently following a Poisson or Bernoulli distribution.

Evaluation of a congestion control mechanism using these workloads lead us to identify the vast differences in network throughput observed by each processing node at heavy non-uniform loads. This network unfairness is not caused by the mechanism itself but by the uneven nature of the workload. Consequently, we question the valid-

[3] In this context starvation is not an issue: if the network somehow favors some nodes, they will send their workload faster than others, but will eventually stop, allowing the rest of the nodes to progress faster, until all of them have sent their packets.

ity of average peak throughput as the figure of merit under non-uniform heavy loads and independent traffic sources (ITS). In fact, changing the injection model to burst synchronized sources (BTS), a workload closer to the pattern generated by real parallel applications, leads to different conclusions about the goodness of that congestion control mechanism under non-uniform loads.

In short, the ITS model fails to reflect the communication behavior of loosely coupled parallel applications. This leads to incorrect conclusions when evaluating *any* router mechanism at loads beyond saturation. BTS is used instead to model the synchronized behavior exhibited by coupled parallel applications at high loads.

We are conscious that further characterization of application workloads is needed to guide the development of synthetic workloads that reflect the communication structure (various levels of message coupling) that exist in most parallel applications.

References

[1] M. Blumrich, D. Chen, P. Coteus, A. Gara, M. Giampapa, P. Heidelberger, S. Singh, B. Steinmacher-Burrow, T. Takken, P. Vranas. “Design and Analysis of the BlueGene/L Torus Interconnection Network” IBM Research Report RC23025 (W0312-022) December 3, 2003.

[2] T. J. Callahan, S.C. Goldstein. “NIFDY: A Low Overhead, High Throughput Network Interface”. Proc. of the 22nd Annual International Symposium on Computer Architecture, ISCA '95, Santa Margherita Ligure, Italy. pp. 230-241, June, 1995

[3] The Chaotic Routing Project at the U. of Washington. Chaos Router Simulator. Available at http://www.cs.washington.edu/research/projects/lis/chaos/www/chaos.html

[4] W.J. Dally & B. Towles. Principles and Practices on Interconnection Networks. Morgan Kaufmann, 2004.

[5] J. Duato. “A Necessary and Sufficient Condition for Deadlock-Free Routing in Cut-Through and Store-and-Forward Networks”. IEEE Trans. on Parallel and Distributed Systems, vol. 7, no. 8, pp. 841-854, 1996.

[6] C. Izu, J. Miguel-Alonso, J.A. Gregorio. "Packet Injection Mechanisms and their Impact on Network Throughput". Technical report EHU-KAT-IK-01-05. Department of Computer Architecture and Technology, The University of the Basque Country. Available at http://www.sc.ehu.es/acwmialj/papers/ehu_kat_ik_01_05.pdf.

[7] S. Mukherjee, P. Bannon, S. Lang, A. Spink and David Webb, “The Alpha 21364 Network Architecture”, IEEE Micro v. 21, n. 1 pp 26-35, 2002.

[8] F. Petrini, D. Kerbyson and S. Pakin. “The Case of the Missing Supercomputer Performance: Achieving Optimal Performance on the 8,192 Processors of ASCI Q”. In IEEE/ACM SC2003, Phoenix, AZ, November 2003.

[9] V. Puente, J.A. Gregorio, R. Beivide and C. Izu, “On the Design of a High-Performance Adaptive Router for CC-NUMA Multiprocessors”, IEEE Trans. on Parallel and Distributed Systems, Vol. 14, NO. 5, May 2003.

[10] V. Puente, C. Izu, R. Beivide, J.A. Gregorio, F. Vallejo and J.M. Prellezo (2001). “The Adaptative Bubble Router”. Journal of Parallel and Distributed Computing. Vol 61 - n. 9.

[11] J Shin, TM Pinkston. “The Performance of Routing Algorithms under Bursty Traffic Loads”. Proc. Int. Conf. on Parallel and Distributed Processing Techniques and Applications. Las Vegas (USA), Jun. 2003.

[12] SMART group at the U. of Southern California. FlexSim 1.2. Available at http://ceng.usc.edu/smart/FlexSim/flexsim.html