Modeling a Fibre Channel Switch with Stochastic Petri Nets

Gianfranco Ciardo

Dept. of Computer Science, College of William and Mary

Williamsburg, VA 23187-8795, USA

ciardo@cs.wm.edu

Ludmila Cherkasova, Vadim Kotov, and Tomas Rokicki

Hewlett-Packard Labs

1501 Page Mill Road, Palo Alto, CA 94303, USA

{cherkasova,kotov,rokicki}@hpl.hp.com

As computing systems become increasingly complex and concurrent, performance analysis becomes more important, especially when designing overall architecture. Simple numerical approximations can often yield bounds on performance. Queueing theory and Markovian analysis can provide insight into the steady-state performance of the system. Discrete-event simulation can provide a more detailed view of the system behavior. We used all of the above techniques in the performance study of a Fibre Channel switch. An approximate stochastic Petri net (SPN) model was employed, obtained by exploiting symmetries in the system. The most challenging aspect was determining transition firing probabilities that accurately reflected the dynamic behavior of the exact SPN. This model was valided against a detailed simulation and found to be accurate within 3%. See [2] for details and references.

1 System description

We consider a switch with N ports, numbered from 0 to $N-1$ (Fig. 1). The switch contains buffer pools organized by input port, a router to transfer header information to output queues, and a scheduler to schedule the transfer of data from an input buffer across the crossbar to an output port. When a message arrives at a port, it is read into a buffer as soon as one is available. The router cyclically polls each port, bypassing ports without new messages. During the poll of a particular port, the router reads the header information of the new message and inserts it into a scheduler table. This table is organized as N FIFO queues, one per output port. The router starts routing a message as soon as the header is in the buffer. A message from port p has a destination of one of the $N-1$ other ports, all of which are equally probable. The scheduler table is used to control the $N \times N$ crossbar switch. At most one message from each input port buffer can be transferred at any time, and only when it is at the top of an output queue. These restrictions cause head-of-line blocking, which is the main performance bottleneck: when different output ports have messages at the top of their queues that are from the same input port, only one of them can be transferred at any time, while the others are blocked.

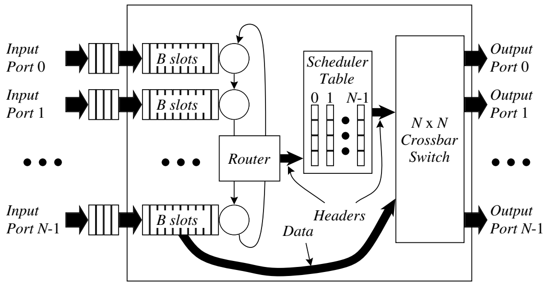

Figure 1: System structure.

We considered a Fibre Channel switch with $N = 16$ ports and port bandwidth of 26.6 Mbytes/sec. Each port has $B = 15$ buffers. The port bandwidth determines the rate at which data moves through each of the components of the switch. Hence, the time to transfer a message of length L_{Msg} into a buffer slot or through the crossbar is $T_{Msg} = L_{Msg}/26.6\mu\text{sec}$. The time required by the router to route a message is T_{Rtr}, while the time between the instant a message arrives in the queue and the router can begin routing it is T_{Hdr}. We assume that all activities have exponentially distributed durations. This is realistic if the arrival streams are roughly Poisson and if the message length is geometrically distributed.

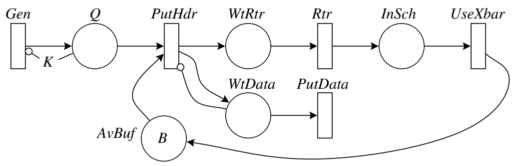

Figure 2: Approximate SPN model of the switch.

Transition	**Firing rate**
Gen	LF/T_{Msg}
$PutHdr$	$1/T_{Hdr}$
$PutData$	$1/(T_{Msg} - T_{Hdr})$
Rtr	$1/(T_{Rtr}(1 + (N-1)\alpha))$
$UseXbar$	$\gamma(\#(InSch))/T_{Msg}$

Table 1: Transition firing rates for the SPN model.

2 Approximate SPN model

An exact SPN model for the switch can be built, but its state space is exceedingly large. Because of the extensive symmetries in the system and of the FIFO policy used in the scheduler table, a colored SPN would greatly simplify the description of the model. We choose instead to perform an approximate study employing truncation of the state space, "folding" of the colored SPN, and fixed-point iteration.

The resulting SPN considers the switch from the point of view of a particular input port, say 0 (Fig. 2). An arriving message (firing of transition Gen) must wait (in place Q) for a buffer. The inhibitor arc from Q to Gen with multiplicity K enforces a truncation of the state space. When a buffer is available (place $AvBuf$ is not empty, $\#(AvBuf) > 0$), the message header is copied into it (transition $PutHdr$). After this, the rest of the message can be copied (transition $PutData$), but, concurrently, the header (in place $WtRtr$) can start requesting service from the router (transition Rtr), which puts the header information in the appropriate output queue (place $InSch$). The message chooses a particular queue with uniform probability over $\{1, 2, \ldots, N-1\}$, but the model does not represent this choice explicitly ($\#(InSch)$ is the total number of messages queued for any destination from source 0). After using the crossbar switch (transition $UseXbar$), a message at the top of its queue leaves the switch and releases the buffer (arc from $UseXbar$ to $AvBuf$). The firing rates of the transitions are shown in Table 1. The key parameters α, the probability that the router finds a new message at any given port during its polling cycle, and $\gamma(i)$, the probability that at least one output queue has a message from source 0 at the top when there are i messages from source 0 in the scheduler table ($\#(InSch) = i$), are approximated using a fixed-point iteration. For α, we simply set its value for iteration $k+1$ as

$$\alpha^{[k+1]} \leftarrow \Pr\{\#(WtRtr) > 0\}^{[k]}.$$

The right-hand side is the probability that source 0 (and any other source, because of symmetry) has at least one new message waiting to be routed, computed at the k-th iteration. If the average number of messages from other sources competing with messages from source 0 at each output queue is σ, $\gamma(i)$ can be approximated by $\Gamma(i, N-1)$, where Γ is defined recursively as:

- $\forall j \in \{1, \ldots, N-1\}, \Gamma(0, j) = 0.$
- $\forall i \in \{1, \ldots, B\}, \Gamma(i, 1) = i/(i + \sigma).$
- $\forall i \in \{1, \ldots, B\}, \forall j \in \{2, \ldots, N-1\}, \Gamma(i, j) = \sum_{k=0}^{i} c(i, k, j)(k + \sigma\Gamma(i-k, j-1))/(k + \sigma).$

where $c(i, k, j)$ is the probability that, when i messages are distributed over j queues, queue j contains exactly k messages; this is given by the binomial distribution. Using a convolution-like tableau, Γ is computed in $O(B^2N)$ time, before each solution of the SPN. Finally, σ is obtained iteratively, as

$$\sigma^{[k+1]} \leftarrow (N-2)/(N-1)\mathrm{E}[\#(InSch)]^{[k]}.$$

The factor $(N-2)/(N-1)$ is due to the fact that a source does not send to itself.

However, the value of $\gamma(i)$ obtained with this heuristic is overly optimistic under heavy traffic. We thus incorporate known results for saturated $N \times N$ crossbar switches. In particular, with exponentially distributed unit service time, the expected bandwidth for each port has been shown to be approximately $N/(2N-1)$. We set $\gamma(i)$ to a weighted sum of our original heuristic, $\Gamma(i, N-1)$, and of the known upper bound:

$$\gamma(i) = (\Gamma(i, N-1) + N/(2N-1) \cdot \sigma)/(1 + \sigma).$$

A second heuristic was also used, which scales all the values of $\gamma(i)$ so that, for i "close" to σ, $\gamma(i)$ is not greater than $N/(2N-1)$.

Experimentally, we observe that the two heuristics bound the results obtained from a discrete event simulation, and that the approximation error is never more than 3%. The number of fixed point iterations is dependent on the load on the switch. Very low or high load result in few iterations, while the highest number of iterations, 60, occurs for medium loads.

References

[1] G. Ciardo, L .Cherkasova, V. Kotov, T. Rokicki. Modeling A Fibre Channel Switch with Stochastic Petri Nets. *HP Laboratories Tech.Report, HPL-94-107*, November, 1994.