Multipath packet switch using packet bundling

Berger, Michael Stübert

Published in:
High Performance Switching and Routing

Link to article, DOI:
10.1109/HPSR.2002.1024244

Publication date:
2002

Document Version
Publisher's PDF, also known as Version of record

Multipath packet switch using packet bundling

Michael Berger
Research Center COM
Building 345v, Technical Univ. Of Denmark
DK-2800 Kgs. Lyngby, Denmark
Phone: +45 45 25 38 53 Fax: +45 45 93 65 81
E-mail: msb@com.dtu.dk

Abstract

The basic concept of packet bundling is to group smaller packets into larger packets based on, e.g., quality of service or destination within the packet switch. This paper presents novel applications of bundling in packet switching. The larger packets created by bundling are utilized to extend switching capacity by use of parallel switch planes. During the bundling operation, packets will experience a delay that depends on the actual implementation of the bundling and scheduling scheme. Analytical results for delay bounds and buffer size requirements are presented for a specific scheduling algorithm, and compared to simulation results.

1. INTRODUCTION

Networking technologies such as ATM, IP and MPLS have one thing in common which is the need for packet switch fabrics within the switches or routers. The capacity of optical transport networks increases rapidly because of an increase in both the number of wavelengths and the bit-rate. This will in turn create a demand for high capacity switch fabrics in the network core nodes both with respect to port speed and aggregate bandwidth. A packet switch node is generally comprised of a switch fabric and a number of traffic managers that perform header analysis, QoS queuing and traffic shaping. The switch fabric may switch only fixed length packets and in this case the traffic manager must support segmentation and reassembly if the network layer uses variable length datagrams. The capacity of a packet switch node is often limited by the minimum packet size that is supports. If the packet size is sufficiently long then the switch fabric capacity is easy extendable by cutting the packet into slices that are switched over parallel planes. However, if the packet size is too long then it is impossible to obtain an efficient filling.

High switching capacity can be achieved in multipath/ multistage switch systems where small switch units are interconnected to form a larger switch fabric. Banyan and Clos are examples of such interconnection networks [1]. A multistage fabric may have more than one route between each pair of inputs and outputs and a routing function is then required [2].

This paper presents another approach for scaling the switch capacity: As discussed above increasing the packet length can increase switch capacity. This can be achieved by bundling a number of smaller packets into one larger packet at a higher bit rate. This principle is widely utilized within TDM (Time Division Multiplexed) networks, e.g. PDH and SDH/SONET. A number of lower order frames are multiplexed into one higher order frame, for instance in SONET that may group 4 STS-3 signals into one STS-12 frame. Thus, the frame length measured in seconds is identical for lower and higher order frames.

In this paper, packet bundling within packet switching is considered in order to determine the feasibility of this concept. Traffic bundling generally requires buffering and scheduling because packets are grouped together subject to specific constraints such as QoS class and destination. The switch fabric architecture and the concept of packet bundling is presented in section 2. In section 3, the queuing and scheduling issues related to packet bundling are considered. The bundling operation will delay packets and a scheduling algorithm that can provide bounded delay is presented. Analytical expressions for the delay bound and maximum queue size are then derived. Section 4 presents simulation results in order to compare the delay bound with actual delay distributions for different traffic distributions and to compare different scheduling algorithms. Finally, in section 5, some concluding remarks are given.

2. SWITCH ARCHITECTURE

The principle of traffic bundling is illustrated in Figure 1. M incoming packet flows are aggregated into one outgoing flow. It is assumed that the packet size is fixed and that the packet size (in bits) of the outgoing flow is k times that of the incoming flow, so the outgoing packet can hold k incoming packets as a maximum. It is assumed that the duration of incoming and outgoing packets are identical which implies that the bit rate of the outgoing packets is k times that of the incoming packets. Note that k must be greater than or equal to M, otherwise it is not possible to operate the queues without packet loss.

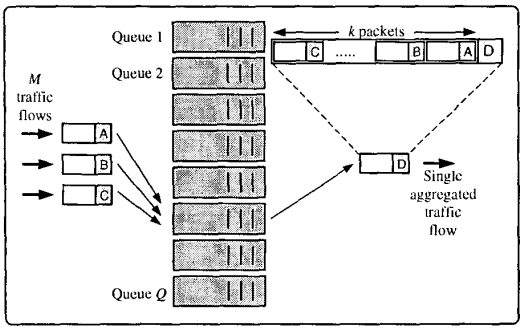

Figure 1: Traffic bundling unit

This is due to the fact that, if M packets arrive each time slot then M packets must be removed on average. k must be greater than M in order to provide bounded delay because it may sometimes be necessary to transmit fewer than M packets. As a compensation, more than M packets must be transmittet in some other time slots.

The header of the incoming packet determines the destination queue in the bundling unit. The number of different queues is denoted Q. A specific queue can for instance be related to a specific destination and service class within a switch. Therefore, the outgoing packet contains only packets from one specific queue in each time slot.

The switch architecture that employs packet bundling is shown in Figure 2. It is a $(M \cdot Q)x(M \cdot Q)$ switch comprised of k QxQ switch elements and additional Mxk and kxM input and output elements. The switch elements form a so-called Clos-network [1]. Note that M and k in Figure 2 corresponds to M and k within Figure 1. The Mxk input elements perform the packet bundling. The scheme shown in Figure 1 needs to be modified slightly, because the k packets are now sent across k parallel planes and not in one larger packet. That is, the aggregated packet is cut into k slices. The k switch planes will therefore receive an identical input traffic distribution and the delay through each plane will be identical, thereby ensureing that no packet will arrive out of sequence. Each bundling unit in the ingress Mxk block holds a number of queues Q that equals the number of inputs and outputs in each switch plane times the number

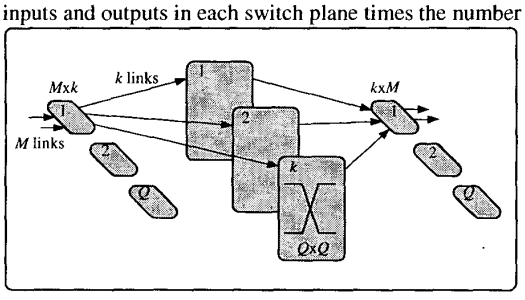

Figure 2: multipath packet switch

of service classes. In case that less than k packets are transmitted from a specific bundling queue, empty packets must be transmitted (to the same destination) over the remaining switches. In order to provide bounded delay in the bundling unit k must be greater than M to compensate for the case where empty packets are transmitted.

The switch must be able to support multicast. The destinations of a multicast packet are determined by its multicast group identifier, which is converted to a multicast mask with Q bits. Packets with identical group id can be bundled together, however, this will require a bundling queue for each multicast id. Another option is to manipulate multicast masks of bundled packets by calculating the logical OR of masks. Thereby identical traffic distributions across each switch plane are achieved. By having larger multicast fan out than specified by the sender, the excess packets must be discarded by the egress kxM stage. The simplest solution is a scheme where all multicast traffic is bundled as one type, and this might waste a lot of bandwidth. If one of the packets is a broadcast packet, then all packets in the bundle will be broadcasted. A better solution is to make several multicast bundling queues and in the bundling try to gather packets with limited differences in their multicast mask. If the multicast mask only differs slightly, the bandwidth waste will be limited.

Switch fabric protection can be achieved easily by adding an additional QxQ switch plane, i.e by increasing k. This will furthermore increase the performance during normal operation.

Many multistage/multipath switch architectures have been proposed in the literature; one scheme is denoted single stage port expansion [4][5] where the number of switch chips grows quadratically with the expansion factor. The number of switch chips only grows linearly with the expansion factor for the bundling scheme in Figure 2. The Atlanta architecture [6] is based on a Clos network similar to that in Figure 2. The central switch elements are bufferless crossbars, so all packets will receive identical delays independent of the selected crossbar slice. This scheme requires a so-called concurrent dispatching algorithm to solve output contention in the crossbars. The required speedup (expansion factor) for non-blocking operation is 5:8, that is, M=5 and k=8 [6]. The bundling scheme can be non-blocking for a smaller expansion factor of e.g. 5:6 as shown in the next section.

3. SCHEDULING

This section analyzes the traffic bundling unit shown in Figure 1 in more detail with respect to queuing and scheduling. A scheduling algorithm is generally required in order to determine the queue from which to transmit an outgoing packet. The objective of the scheduling algorithm is to ensure that packets are bundled efficiently

but at the same time it must ensure that the packet delay is bounded. The objective is not to ensure fairness among traffic from the Q queues, so scheduling algorithms like WFQ (Weighted Fair Queuing) or similar approaches [3] are not considered.

The scheduling scheme has its impact on the amount of buffering that is required, and the maximum delay that a packet will experience. A scheduling algorithm that is easy to implement is Round Robin (RR) where backlogged queues are selected in turn. If the objective is to maximize the utilization of the outgoing packets, then a queue is backlogged if it contains at least k packets. It is impossible to provide any delay guarantees in that case since a single packet can wait forever in a specific queue. To overcome this problem a queue must be considered backlogged if it contains at least one packet. In this case the maximum queue size and the maximum delay is bounded if the traffic is distributed equally across the queues: The size of each queue will initially grow until the system reaches equilibrium where M packets can be removed each time slot. However, there exists a traffic pattern for which the delay and queue size is unbounded. Consider a situation where a single packet is destined for each of the first (Q-1) queues. It is then assumed that packets are destined for the last queue in the subsequent time slots until all the first (Q-1) queues have received service. If this scenario is repeated, then the delay and queue size of queue number Q will grow infinitely (assuming that $Q>>k$).

It can be avoided that a queue grows infinitely by selecting a scheduling algorithm that always serves the longest queue. In the following, this scheme will be denoted (LQ). The total queue size is upper bounded by $M \bullet Q$ because if the total queue size is at this bound then at least the longest queue must contain M packets. In this case at least M packets is removed, and a maximum of M packets will arrive and thus, the total queue size will not increase. However, the delay is not bounded for LQ since a single packet in a specific queue may wait forever to receive service when another longer queue exists.

In the following, another scheduling approach is considered, that can provide bounded delay for packets entering the bundling buffers. The scheduler works as follows: Each arriving packet is time stamped, and the backlogged queues are sorted according to the time stamp of the head of line packet. The queue with the lowest time stamp value is selected for transmission, and up to k (incoming) packets are removed from that queue. This scheme is denoted Time Stamp (TS). The maximum delay D, measured in timeslots, is given by:

$$D = \left\lceil \frac{(Q-1)\cdot(k-1)}{k-M} \right\rceil \qquad (1)$$

And the maximum total queue size B, measured in packets, is given by:

$$B = D \cdot M = \left\lceil \frac{(Q-1)\cdot(k-1)}{k-M} \right\rceil \cdot M \qquad (2)$$

From equation (1) it is observed that k must be greater than M in order to provide bounded delay. The minimum value of k is thus $k = M+1$. In this case the following equations are obtained:

$$D = (Q-1)\cdot M \;,\; B = (Q-1)\cdot M^2 \qquad (3)$$

The equations (1) and (2) are derived as follows: the worst-case scenario must be identified, where a high number of outgoing packets only contain a single incoming packet. The input traffic distribution is selected such that a single packet is transmitted to each of the first Q-1 queues. This is repeated as often as possible with the restriction that only one cell must be transmitted from these queues, i.e., each of the (Q-1) queues must at a maximum contain one packet. In the meantime, incoming packets are transmitted to queue number Q. Figure 3 shows the en-queue and de-queue operations for this worst-case scenario. The value of M is 2 and the value of k is 4. The squares show arriving packets and the circles show departing packets. It is assumed that packets can be transmitted from the queue in the time slot where they arrive. However, this assumption has no impact on the worst-case delay. Note that two packets are en-queued each time slot, which gives a slope of (–2) for the 'en-queue' graph. At time t1, each queue has received one packet, and the first $Q/2$ queues have received service. At time t2, service starts for the packets that arrived in the interval [t1:t2]. The last packet that arrived within this interval is transmitted at t5. In the interval [t2:t3] packets are en-queued in FIFO number Q. t3 is selected such that the en-queue and the de-queue graphs intersect at t5. If t3 is moved forward in time, then some of the first (Q-1) queues will contain more than one cell at the time of transmission, and it is no longer a worst-case scenario. On the other hand, if t3 is moved backward in time, then a higher number of packets will be en-queued to queue number Q, which can

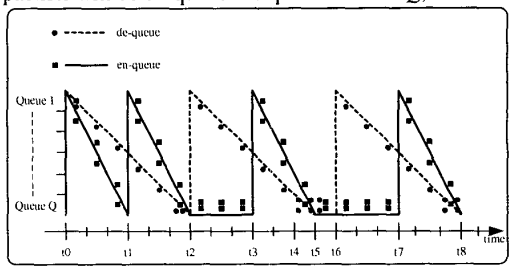

Figure 3: Worst case packet service (M=2, k=4, Q=6)

be efficiently removed, thereby leaving the worst-case situation. The packets en-queued in [t2:t3] are transmitted in the interval [t4:t6]. The duration of [t4:t6] is 2/3 of [t2:t3] because 2 timeslots are required to de-queue six packets.

The duration of the interval where packets are transmitted to queue number Q is increasing with time (e.g. [t5:t7] > [t3:t2]). However, after a given amount of time the system will be in equilibrium. The equilibrium condition is shown in more detail within Figure 4. Note that the shown period corresponds to the interval [t2;t6] in Figure 3. It is still assumed that $M = 2$ and $k = 4$. Furthermore the number of queues Q is set to 6 in this example. Note that the total number of squares equals the total number of circles because of the equilibrium condition.

It is now possible to calculate the number of time slots in Figure 4. This number is equal to the maximum delay of a packet. To see this, consider a packet that arrives at t4, this packet will receive service at time t8, that is, one period later. The number of timeslots is denoted D, and is given by the following equation.

$$M \cdot D = (Q-1) + k \cdot (D-1-(Q-1)) + r \qquad (4)$$

The left side is the number of squares, which is always M per timeslot. The right side expresses the number of circles. There is one circle for the first (Q-1) time slots and the following (D-1-(Q-1)) time slots contain k circles each. The last timeslot contains r circles ($M < r \leq k$). Solving for D gives

$$D = \frac{(Q-1)\cdot(k-1)}{k-M} + \frac{k-r}{k-M} \qquad (5)$$

Since D is an integer and the last part of the equation above is less than 1, the result given by equation (1) is finally obtained.

The total number of packets in the queues will show a local maximum at time t4, t8... At equilibrium, this is also the global maximum. To obtain the number of packets at e.g. t4, the en-queue operation is stopped at that time and the numbers of packets that leave the queues in the following period are counted. At equilibrium this number is $M \cdot D$ which leads to equation (2).

The worst-case delay and buffer requirements are useful in an actual physical implementation because the maximum number of different timestamp values is D, and B gives the memory requirement.

4. Simulation and Results

The goals of the simulation are to examine the presented bundling scheme with respect to mean delay and delay variations. The delay distribution of packets in the queue

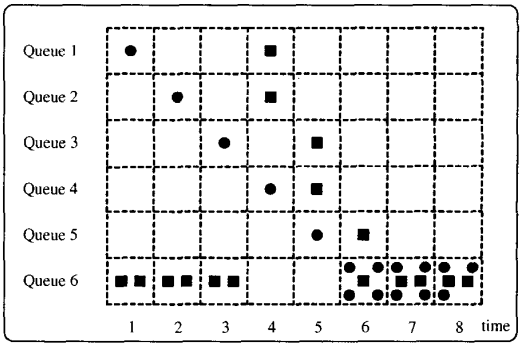

Figure 4: One period of de-queue and en-queue operations

system in Figure 1 depends on the distribution of arriving packets and the scheduling mechanism.

In general, the packet inter-arrival time and the destination queue are given by stochastic variables. However, it is assumed that the system is fully loaded with a packet arriving in each timeslot on each of the M channels. The only stochastic variable is thus the destination queue. It is assumed that the different queues are selected with equal probability $1/Q$. A switch fabric of size 128x128 is considered which is generated from three 64x64 switch elements; the parameters in Figure 2 are thus: $Q=64$, $M=2$ and $k=3$. Figure 5 shows the (un-normalized) probability distribution for delays for three different scheduling methods, LQ, RR and TS. The mean values are as follows: LQ=17.7, RR =34.3 and TS = 28.7. The delay bound can be calculated for the TS scheduler according to eq. (1) as D= 126. It is noted that the LQ scheduler has the lowest mean value so that most of the packets obtain a low delay, however, the tail of the distribution is much longer than for RR and TS. Actually, the LQ distribution takes values far beyond 126. The mean value of RR is higher than for LQ, but the tail of RR is reduced compared to LQ, which makes RR more attractive than LQ. The TS scheduler has a mean value that is lower than RR, furthermore, the slope of the distribution falls steep towards zero at a delay value around 50, which is far below the theoretical maximum at 126. The fact that TS has a lower mean value and a smaller tail than RR makes TS the most well performing scheduling scheme for this traffic scenario.

A number of experiments has been carried out with different distribution functions for the destination queue; in experiment one, the probability of selecting a specific queue was proportional to the queue number, and in experiment two the queue number was selected according to a exponential distribution (truncated and normalized). The resulting probability distributions for delay does not vary much from that shown in Figure 5, so the above conclusions regarding LQ, RR and TS hold for a number of different destination queue distributions.

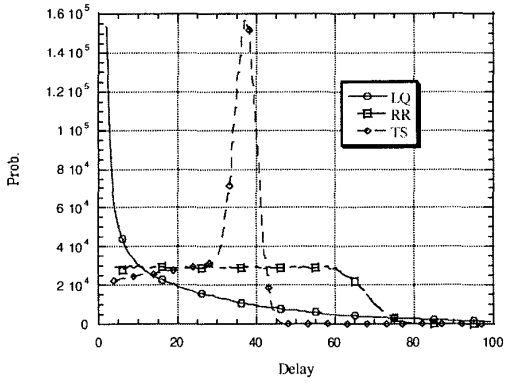

Figure 5: Probability distribution for delay (M=2,k=3,Q=64)

The probability distributions will now be examined for a larger switch fabric of size 256x256, defined by the following parameters: Q=64, M=4 and k=5. The result is shown in Figure 6. The mean values are LQ = 25.1, RR = 70.6 and TS = 38.0. The worst-case delay for TS given by eq. (1) is 252. By comparison of Figure 6 and Figure 5 the same conclusions regarding LQ, RR and TS are reached, actually the TS scheduler performs even better than the other two in the M=4 case than the M=2 case.

For a given switch size e.g. 256x256, Q=64, M=4, the value of k can be increased in order to reduce the bundling delay. Also the worst-case delay given by eq. (1) is reduced towards (Q-1) for large k values. Table 1 shows the delay mean values for LQ, RR, TS and the maximum for different values of k. The worst case bound for TS is shown as well.

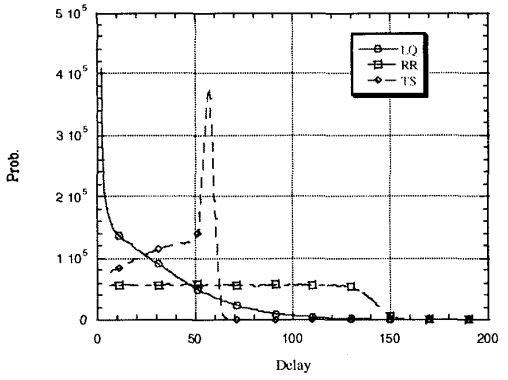

Figure 6: Probability distribution of delay (M=4,k=5,Q=64)

Table 1: Mean delay vs. k

k	LQ	RR	TS	TSmax
5	25.1	70.6	38.0	252
6	25.1	42.4	32.9	158
7	25.1	35.7	31.3	126
8	25.1	34.0	30.8	111
16	25.1	33.0	30.6	79

As discussed previously, protection can be introduced by increasing the value of k. By using the value of 6, instead of 5, the mean value is reduced by 13% and the maximum value by 37 % for the TS scheduler according to Table 1. Non-blocking operation is still possible if one of the six switch plane fails, but with the cost of increased delay.

5. CONCLUSION

Traffic bundling is not utilized in packet switching; however, this paper demonstrates that the concept of traffic bundling has attractive properties that can be utilized within multistage/multipath packet switch fabrics where aggregated packets are transmitted over identical parallel planes.

A simple scheduling algorithm was proposed that timestamps arriving packets, and serves packets in order of increasing timestamp. Worst case scheduling delay and buffer occupancy was derived for this specific scheduling algorithm. The proposed scheduling algorithm performs bundling efficiently (i.e., with the smallest possible bandwidth overhead), and on the same time, bounded delay is provided. Simulation results demonstrated that the actual delay for different distributions is much smaller than the derived worst-case value.

6. REFERENCES

[1] Hui, J.Y. " Switching and traffic theory for integrated broadband networks" Boston : Kluwer, 1990.

[2] A. Herkersdorf, L. Heusler, E. Maehle "Route discovery for multistage fabrics in ATM switching nodes" Performance evaluation 22 (1995).

[3] A. Varma, D. Stiliadis "Hardware Implementation of Fair Queuing Algorithms for Asynchronous Transfer Mode Networks", IEEE Communications Magazine, December 1997.

[4] W.E. Denzel, A.P.J. Engbersen, I. Iliadis "A flexible shared-buffer switch for ATM at Gb/s rates" Computer Networks and ISDN systems 27 (1995) 611-624.

[5] Minkenberg, C.; Engbersen, T. "A combined input and output queued packet switched system based on PRIZMA switch on a chip technology" IEEE Communications Magazine, Dec. 2000.

[6] Chiussi, F.M. Kneuer, J.G. Kumar, V.P. "Low-cost scalable switching solutions for broadband networking: the ATLANTA architecture and chipset" IEEE Communications Magazine, 35(3).