Unslotted deflection routing: a practical and efficient protocol for multi-hop optical networks

Thierry Chich[*], Johanne Cohen[†] and Pierre Fraigniaud

Laboratoire de Recherche en Informatique – CNRS

Bât. 490, Univ. Paris-Sud

91405 Orsay cedex, France

Abstract

Slotted Optical Time Division Multiplexing Deflection networks make use of the synchronous arrival of the packets to the routers in order to optimize locally the number of deflections. In this paper, it is shown that the difference of performances between slotted and unslotted networks is mainly due to the fact that unslotted networks cannot directly make benefit of such local optimization. It is also shown that, unfortunately, optimizing locally the routing in unslotted networks gives rise to an NP-complete problem. Therefore a heuristic for routing in unslotted networks is proposed. In the experimental context considered, our heuristic enhances unslotted routing almost at the same level as slotted routing. It makes unslotted deflection routing a competitive alternative to slotted deflection routing for OTDM networks.

Keywords: All-optical networks, deflection routing.

[*]Supported by the Centre National d'Etude des Télécommunications (CNET) grant 9408B54.

[†]Supported by the Direction des Etudes Techniques (DRET) of the DGA.

1 Introduction

All-optical networks provide high bandwidth and fault-tolerant communications by avoiding the bottleneck due to the electro-optic conversion. Several types of optical networks are considered in the literature [2, 5], and several methods are used to share the bandwidth of optical networks. Among them, the TDM technique (Time Division Multiplexing) is a technique used to improve the bandwidth of a single wavelength channel [23]. In TDM networks, every message is decomposed in packets. Several studies (including experimental testbeds) have been carried out to high-capacity optical packet routers [15, 22]. Deflection routing is a frequently proposed protocol because it does not require large buffers [7, 5, 20]. In deflection routing, a packet requesting a given output link already used by another packet is deflected on an alternate path. This technique allows to avoid packet destruction inside the network. It simplifies management and avoids congestion.

Most of the papers in the literature assume a slotted system for OTDM (Optical TDM) networks [3, 10, 25]. In such a system, each packet is inserted in a time-slot of fixed duration, which includes the header and the payload that are conveyed at different bit rates, and all incoming slots entering a router on different input links are synchronized. However, unslotted systems are sometimes proposed as an alternative for OTDM networks [8, 9, 12]. In such systems, each packet has a length proportional to its size.

Slotted systems offer many advantages. For instance, packets can be inserted in the network as soon as a free time-slot is available. Slot synchronism allows to locally optimize the requests of the packets, and therefore to limit the number of deflections. Moreover, synchronous routing can be performed by rearrangeable multi-stage switches. However, slotted systems present some drawbacks. They require additional hardware that produces important degradation of the signal [23]. Furthermore, fixed slot length does not allow to adapt the packet size to the need of the application. Finally, slotted networks are very sensitive to faults in the synchronization system. Unslotted systems do not require synchronization hardware (but a standard non blocking switch). Packet sizes are tuned according to the needs of the users. However, it is often said that unslotted systems present three major problems. First, such systems do not succeed to make use of the whole bandwidth of the network because of the variability of the inter-packet spaces. Second, asynchronous packet arrival does not allow to maximize the global requests of the packets (a "don't care" packet can force the deflection of a forthcoming packet). Thirdly, it is pointed out in [8, 9] that unslotted systems may cause important congestion phenomenons.

In this paper, we show that the three previously listed problems related to unslotted OTDM deflection networks are either of minor influence or can be overcome. Indeed, we show that the bandwidth lost in inter-packet spaces of unslotted networks is not more significant than the bandwidth lost in the filling up of the slots by small packets. Moreover, we show that in unslotted networks, congestion appears in a particular configuration of the network that can be easily avoided. The second problem, that is the local optimization of the requests, is actually the most important problem. Indeed, it cannot be solved via small hardware adjustments, but requires an algorithmic approach. We show that the underlying problem is NP-complete, but that it is possible to derive a fast and efficient heuristic. This heuristic allows to increase the throughput of unslotted networks of about 35%. It makes unslotted deflection routing a competitive alternative to slotted deflection routing for OTDM networks.

2 OTDM networks

2.1 Routing in all-optical networks

In all-optical TDM networks, a message is composed of its payload and its header. The payload contains the data (files, images, sounds, etc.), and the header includes useful information for the routing function (destination label, packet number, source label, etc.). The payload circulates at the photonic rate whereas the bandwidth allocated to headers is limited by the electronic bottleneck (622 Mb/s). When a message arrives at a given router, its header is converted in electronic format, and it is decoded by the routing control processor (RCP) which takes the routing decision. Once the decision has been taken according to some simple rules, that is when a single output port has been selected, the router connects the input port to the output port so that the payload can cut through the photonic switch. The payload is just slightly delayed in a loop while the RCP is computing the route. The RCP generates a new header which is added to the outgoing payload.

The routing is performed according to a routing table T. On a given node x, the entrance $T_x[i, y]$ of T measures the "quality" of routing along link i a packet currently in x, and of destination y. For instance, the shortest path routing is specified by $T_x[i, y] = 1$ if the link i is on a shortest path from x to y, and 0 otherwise.

2.2 Slotted and unslotted networks

Slotted networks impose packets of fixed length whereas unslotted networks allow to adapt the length of the packets to the user requirements. It is somewhat difficult to choose the "optimal" packet size in slotted networks. Indeed, too short packets create ordering problems and overcosts due to the reconstruction of the original message. On the other hand, too long packets imply a significant waste of bandwidth.

Inserting packets in slotted networks is easy: it only requires to test whether a slot is empty among the incoming slots. Insertion is a bit more complex in unslotted networks. Four solutions have been proposed in [9]. One of them requires to discard inserted packets as soon as they contend with incoming packets. Two others do not always give priority to transit packets. This suggests to adopt the fourth solution of [9]. A fiber loop is added to each input link in order to delay the arrival of the packets. The arrival times of packets entering the loop are taken into account by the RCP to decide whether there is enough space to insert a packet. This strategy is applicable as soon as packets are of bounded length. However, as opposed to the case of slotted networks, this system does not change the fact that the links cannot be fully occupied by packets. Indeed, the inter-packet space is *a priori* arbitrary.

Slotted networks require to synchronize the arrival time of the incoming packets because providing links of length multiple of the packet length is not enough due to temperature variation and fiber chromatic dispersion [22]. Synchronization is performed by the introduction of switchable delay lines [6, 23]. In any case, the number of traversed optical couplers is at least logarithmically proportional to the product of the precision of the synchronization by the maximum delay between two packets. This induces power losses and decreases the robustness of the network.

Slotted networks make benefit of the simultaneous arrival of the packets. Given k incoming packets $m_1, \ldots, m_k$, let us denote by $p_{i,j}$ the "preference" of packet m_i for the output link j, $j = 1, \ldots, n$. In all our experiments, the preference of a packet is precisely defined by the routing table although it could have been set according to many other parameters such as priority level or alternative routing strategies (source-routing [24], centralized protocol, etc.). These preferences induce a weighted complete bipartite graph whose first set of the partition represents the k packets, and the second one represents the n output links. A natural way to optimize routing in slotted networks locally is to compute a maximum weighted matching in this bipartite graph [18]. If the edge (i, j) belongs to the matching, the packet m_i is said to be *assigned* to the output link j. The polynomial complexity of the maximum weighted matching [17] makes this solution realistic in this context. A packet is said to be *deflected* when it is assigned to an output link which does not correspond to a shortest path between the current node and the destination.

3 An experimental model for OTDM networks

3.1 Topology and routing

We have considered the *bidirectional Manhattan street network* that is the symmetrically oriented torus. Each router is therefore supposed to be a 5×5 crossbar: one of the bidirectionnal link is devoted to the input-output of the optical network. The size of the torus is fixed at 12×12.

We have used the so called Z^2 shortest path routing [4]. This routing selects the output link supporting the maximum number of shortest paths from the current node to the destination. For instance, if the source and the destination are the two opposite corners of a square, then (in absence of contention), the route will zig-zag between these two corners. More formally, for any two nodes x and y, and any output link i of node x, let $N_x[i, y]$ be the number of shortest paths from x to y that pass through link i. The routing table relative to the Z^2 routing is defined by: $T_x[i, y] = \frac{N_x[i,y]}{\sum_i N_x[i,y]}$.

3.2 Traffic generation

We have fixed the size of the input queue to be 100 packets at each node. The bandwidth of the links is supposed to be $10Gb/s$, and each link is supposed to have a length of 2 kilometers.

3.2.1 Packet of variable length

According to the typical IP-traces (see [1]), the length of the packets follows a bimodal law polarized at (1) the length of the acknowledgment packets, and (2) the length of the maximum packet size. Indeed, every message is decomposed in packets by the network-application interface. Therefore, a lot of packets are of length the maximum packet size. However, a few other packets are smaller, in particular the packets corresponding to the remaining part of messages which are not of a length multiple of the maximum packet size.

The choice of the "optimal" packet size, in a fixed packet size context, has been the source of many discussions. In deflection routing, a too small packet (as an ATM cell) would not be suitable because of the need of reordering packets, and of the too large ratio header size over pay load. (Recall that we are in a datagram mode.) A too large packet would produce a waste of bandwidth as small messages will not fill up the slot.

The time slot is set to be $1\mu s$ in slotted networks ($1\mu s$ at $10Gb/s$ represents $10Kb$). The minimum size of a packet is $200ns$ (that is the size of a header). Let L be the length of a packet. We set $\text{Prob}(L = 200ns) = 0.3$, and $\text{Prob}(L = 1\mu s) = 0.4$. The other packet lengths are chosen as multiple of $0.1\mu s$, uniformly in the interval $[0.3\mu s, 0.9\mu s]$. The average length of a packet is then 642 ns, that is the bit load of a slot is 6.42 Kb.

Unslotted networks support packets of different lengths. We have fixed the maximum size of a packet at $1\mu s$ (that is the size of the slot of slotted networks) in order to facilitate the comparison of slotted and unslotted networks. This equality implies that the packet length distribution is the same as previously described. In particular, the average length of a packet is 642 ns.

3.2.2 Problems arising with time scaling

Simulations of slotted networks are usually performed using a simulation tick exactly set to the time slot. At each tick, all the routers of the network are scanned and the routing is performed. The packet emissions are set according to a Bernouilli law. Unslotted networks could be simulated in a similar way using a shorter simulation tick. However, if the simulation tick is shortened, the average of the Bernouilli law must also be decreased in order to obtain the same average offered load as in slotted networks. The side effect would be that the emission laws of slotted and unslotted simulations would not be exactly the same. Therefore, to perform slotted and unslotted simulations in the same setting, we have separated the simulation tick and the emission tick. Precise details are given in Appendix A. Using this setting, we have performed experiments which showed that simulations performed with a tick equals to 1/200 of a time slot give similar results as simulations performed with a tick equals to the 1/10 of a time slot. Therefore, in all our experiments, the simulation tick is set to 1/10 of the time slot. Finally, in order to simulate sporadic traffic observed in real networks [19, 21], we have modeled the packet emission law by a two states Markovian chain. The method used to obtain the same law for slotted and unslotted networks is detailed in the Appendix B.

3.3 Experimental measures

All measurements are performed at the steady state, on a single run of $10^6 \mu s$ (the steady state is always reached after at most 5 $10^3 \mu s$). We have measured the *throughput* of the network as a function of the input demand. More precisely, we have counted the average number of packets that arrive at destination every μs, divided by the number of nodes (that is 144). The *throughput* is in $[0, 1]$ for slotted networks. Note that the throughput per node and per μs expressed in byte can be obtained in both slotted and unslotted networks by a simple multiplication by $6.42Kb$. The input demand is the average number of packets that each node sends at each step. Since input queues are of bounded size, packets can be lost when the network approaches the saturation. The number of *lost packets* is then inversely proportional to the throughput. We have also considered the *link occupation* of the network, that is the percentage of the bandwidth used at the steady state.

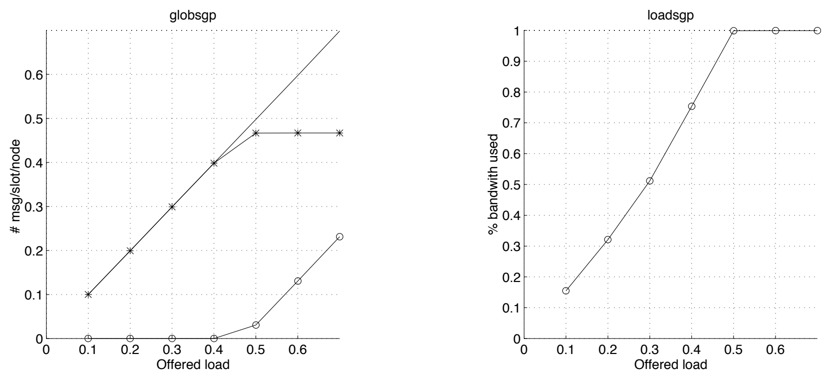
Furthermore, we have created a specific traffic, called *spy traffic*, between two given nodes in order to obtain local measurements. In our experiments, node $(2, 2)$ sends packets to node $(9, 9)$ according to a Poisson law of mean 0.01 (i.e., at a low rate). We have reported the average number of times spy packets are deflected. For a sake of uniformity, we have normalized the results as a function of the number of received packets.

4 A comparison of slotted and unslotted networks

The aim of this section is to show why the slotted mode allows better performances than the unslotted mode.

4.1 Slotted routing

Figure 1 presents the well know behavior of synchronous routing under Poisson traffic. Figure 1(a) shows the two states of the network: a linear increase of the throughput until the network gets saturated. When the network saturates, the throughput becomes constant, and the number of lost packets increases (whereas no packets are lost for a low offered load). One can check that the network starts to saturate for an offered load larger than 0.45 (either by comparison with the diagonal line, or by looking at the number of lost packets).



(a) Throughput ($*$) and lost packets ($\circ$) **(b)** Link occupation

Figure 1: Slotted routing under Poisson traffic

Figure 1(b) presents the average number of packets per link. Again, the result is not surprising. When the offered load increases, the number of packets per slot increases super linearly. This is due to the interactions between the network load on one hand, and the number of packet deflections on the other hand. When the network reaches the saturation, the whole bandwidth of the network is used. This is always the case for slotted networks.

Figure 2 shows that, under low traffic condition (that is for an offered load at most 0.5), the number of deflections increases super linearly. After this threshold, the number of deflections does not significantly change. The same behavior can be observed for standard deviation.

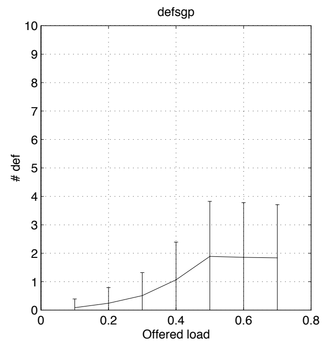
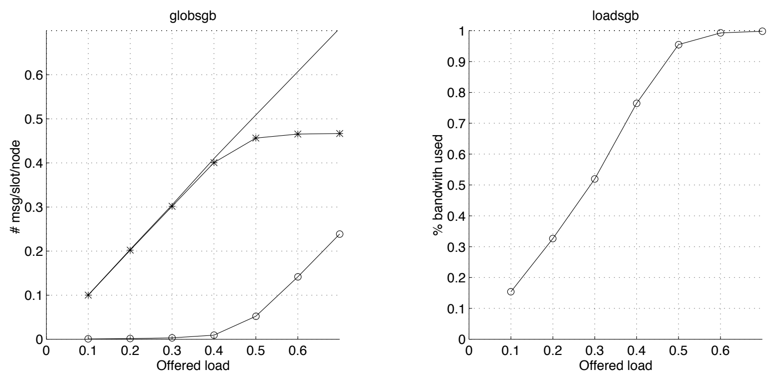


Figure 2: Average and standard deviation of the number of deflections for slotted routing under the Poisson traffic.

Figures 3(a) and (b) show the influence of a sporadic traffic on slotted routing. Figure 3(a) shows that, when the network is not yet saturated, the number of lost packets is larger under the sporadic traffic than under the Poisson traffic. The difference looks rather small on the curves, but such a small difference corresponds to thousands packets that are lost in case of sporadic traffic (actually many packets are lost even for an offered load of 0.1). This is due to the large standard deviation of the bi-Poisson traffic. When the network is saturated, the routings of the two types of traffic offer the same behavior. As one can check on Figure 3(b), the loss of packets under a sporadic traffic imply that the links saturate for a larger offered load than for Poisson traffic. The number of lost packets is the major difference between Poisson and sporadic traffic. However, for a same number of packets inside the network, the behavior of these packets is roughly the same for both traffics.

We did not noticed a significant difference between sporadic and Poisson traffics when looking at the distribution of the delays. Tiny improvements under the sporadic traffic come from the smaller average number of packets per slot in this mode. This confirms the fact



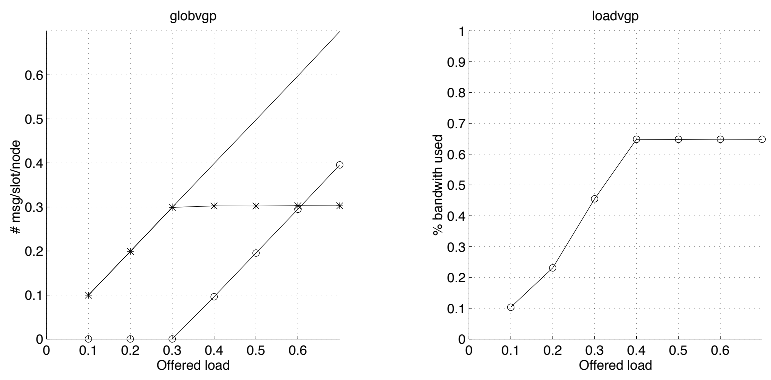
(a) Throughput ($*$) and loss ($\circ$) **(b)** Link occupation

Figure 3: Synchronous routing under sporadic traffic

that, as we already pointed out, the internal behavior of a deflection network is roughly independent of the traffic nature.

4.2 Unslotted routing

Figure 4(a) shows that the throughput of unslotted routing is qualitatively the same as the one of slotted networks. In particular, the saturation state is stable, that is there is no degradation of the throughput as the offered load increases. This is in contradiction with a similar study in [8, 9] which observed a severe degradation of the throughput. However, the experiments performed in [8, 9] assume packets of fixed length (although the routing is unslotted). It is shown in [11] that unslotted routing with fixed size packets induces resonance phenomenons when the length of the links is a multiple of the packet size. Moreover, it is pointed out in [9] that "assuming a fully occupation of the links, any packet arriving at a node finds just one output link free and is forced to follow the path of its predecessor". All these phenomenons induce possible livelocks that strongly reduce the throughput of the network. Finally, the throughput is sensible to the length of the delay loops. Anyway, in standard unslotted networks, packets are of variable length and all these boundary phenomenons do not occur.



(a) Throughput ($*$) and loss ($\circ$) **(b)** Link occupation

Figure 4: Asynchronous routing under Poisson traffic

Figure 4(b) presents the link occupation of unslotted networks. At the saturation state, the link occupation is near 0.7. It corresponds to 1.1 packets per μs. Unslotted networks cannot totally fill up the links because too small inter-packet space does not allow to insert packets (we have measured an average inter packet space of roughly $0.3\mu s$). We did not present results on bursty traffic since, as shown in [12], and as it was the case for slotted networks, as far as the internal traffic is concerned, there is no big difference between Poisson and sporadic traffic in unslotted networks.

Even if unslotted networks present qualitatively the same behavior as slotted networks, there is quantitatively a big difference. In order to understand why such a difference, we have run experiments on greedy routing in slotted networks. Greedy routing considers sequentially the packets arriving at a node in the same time slot. It assigns to the current packet the not yet assigned output link which maximizes the preference of the packet. This strategy is similar to the usual unslotted routing since it does not make use of the global preferences of the packets arriving at a node in the same time slot.

Figure 5 presents the behavior of greedy routing under a Poisson traffic. One can notice a large degradation of the performances in comparison with synchronous routing. For instance, the network get saturated for a much smaller offered load (roughly 0.3 rather than 0.45).

Thus, as observed in [14], a sequential rather than simultaneous treatment of the packets does not allow to reduce the number of deflections locally, and the throughput decreases: the throughput of greedy slotted routing is roughly the same as the throughput of unslotted routing. Moreover, as shown in [12], the distribution of the number of deflections for greedy slotted networks and for unslotted networks offers roughly the same shape (same median, same standard deviation, etc.). This shows that the performance degradation of unslotted routing is mainly due to the difficulty of minimizing the number of deflections locally. This is an algorithmic problem rather than an intrinsic problem of the network asynchronism.

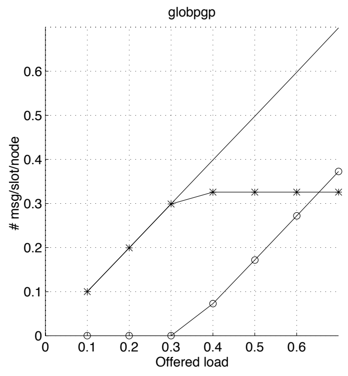


Figure 5: Throughput $(*)$, and number of lost packets $(\circ)$, for the greedy slotted routing under Poisson traffic

5 A heuristic for routing in unslotted networks

In this section, we will make use of the delay loops introduced in Section 2 to predict the future arrival of packets. This prediction will allow us to detect the possible contentions between packets, and thus to minimize the conflicts. Formally, we introduce the *maximum assignment problem* which is, given a set of incoming packets, to maximize the preferences of these packets. Unfortunately, as shown in Section 6, the maximum assignment problem is NP-complete. Therefore, the current section presents a heuristic. This heuristic will be experimentally shown to be quite efficient, and improves the performances of unslotted deflection routing networks of about 35%.

5.1 The maximum assignment problem

Assume that the loops used for inserting packets in unslotted networks produce a delay Δt. The routing decision taken on a packet at time t can benefit from the knowledge of all the future arrivals of packets between times t and $t + \Delta t$. Let us consider a packet m_0 routed at time t, and assume that k packets m_i, $i = 1, \ldots, k$, will arrive within Δt times. One does not want to only maximize the preference of m_0 according to the free output links, but rather to maximize the global preferences of all the $k+1$ packets. For that purpose, we have to take into account the possible contentions between these packets. Let us denote a packet by a couple (t_1, t_2) where t_1 denotes the arrival time of the packet in the router, and $t_2 - t_1$ denotes its length.

Definition 1 *There is a* conflict *between two packets* (t_1, t_2) *and* (t'_1, t'_2) *if and only if* $t_1 \leq t'_1 \leq t_2$ *or* $t'_1 \leq t_1 \leq t'_2$*. The* conflict graph *is a graph* (V, E) *where* V *denotes the set of incoming packets* m_i*,* $i = 0, \ldots, k$*, and* E *denotes the set of conflicts between these packets.*

Note that the conflict graph is an interval graph [17]. Note also that this graph contains information on the future but not on the past of the current router at the current time. Indeed, none of the packets currently routed are considered in the conflict graph. This notion is captured by another structure. Let s_j be the time at which the output link j will bè freed by the packet currently using link j ($s_j = t - 1$ if no packet is using the link j at time t). A packet (t_1, t_2) will not be allowed to request link j if $s_j > t_1$.

Definition 2 *Let* $p_{i,j}$ *be the preferences of packet* i*,* $i \in \{0, \ldots, k\}$ *toward link* j*. The* preference graph *is a weighted bipartite graph* $G = (V_1, V_2, E)$ *where* V_1 *denotes the set of* *incoming packets m_i, $i = 0, \ldots, k$, V_2 denotes the set of the n output links, and there is an edge between a packet $m = (t_1, t_2) \in V_1$ and a link $j \in V_2$ if and only if $s_j \leq t_1$. An edge between packet m_i and link j has the weight $p_{i,j}$.*

Both the conflict graph and the preference graph allow to define the maximum assignment problem.

Definition 3 *An* assignment *is a function ϕ from $\{0, \ldots, k\}$ to $\{1, \ldots, n\}$ such that* **(1)** *if $\phi(i) = j$ then (i, j) is an edge of the preference graph, and* **(2)** *if $\phi(i) = j$ and $\phi(i') = j$ then (i, i') is not an edge of the conflict graph.*

The maximum assignment problem: Finding the assignment ϕ which maximizes

$$\sum_{i \in \{0,\ldots,k\}} p_{i,\phi(i)}$$

Solving the maximum assignment problem is NP-complete (see Section 6). Thus, the next section is devoted to a heuristic for that problem.

5.2 A heuristic for the maximum assignment problem

5.2.1 Description of the heuristic

The maximum assignment problem is polynomial in slotted networks for two reasons: the conflict graph is the complete graph, and there are at least as many output links as the number of routed packets. The idea of our heuristic consists of simplifying the general problem for unslotted networks in order to get a situation similar to that of slotted networks. As a consequence, it will allow us to use the standard routing algorithms devoted to slotted networks. Our simplication is based on the fact that, in general, the routing decision on a packet must take more care of the next packets than of packets arriving much later. Let us formalize this idea.

If a packet m_0 has to be routed at time t, let n' be the number of free output links at that time. Let $m_1, \ldots, m_{k'}$, $k' \leq k$, be the next arriving packets (ordered by their arrival time), and such that, for every i, $1 \leq i \leq k'$, m_i is in conflict with m_0. Note that two such packets are not necessarily in conflict, although they are both in conflict with m_0. To route m_0, our heuristic takes into account m_0 and the $r = \min(k', n'-1)$ next arriving packets. According to slotted routing, we assume that all these $r+1$ packets are pairwise in conflict. This makes the conflict graph complete. We take, as the preference graph G' of our restricted problem, the subgraph of the original preference graph G induced by the n' output links, and the $r+1$ packets.

Remark G' contains a complete bipartite graph whose two partition sets are the $r+1$ packets on one side, and the n' output links free at time t on the other side.

The transformation above yields a problem similar to the assignment problem seen in Section 2.2 for slotted networks. It can be solved by a polynomial maximum weighted matching algorithm in the preference graph G'. The maximum weighted matching in G' can be completed in a maximum matching by adding edges of weight 0. Hence, every considered packet, that is each of the next $r+1$ arriving packets, is assigned to an output link. Therefore the current packet can be routed, and this routing avoids as many deflections as possible for the r next packets.

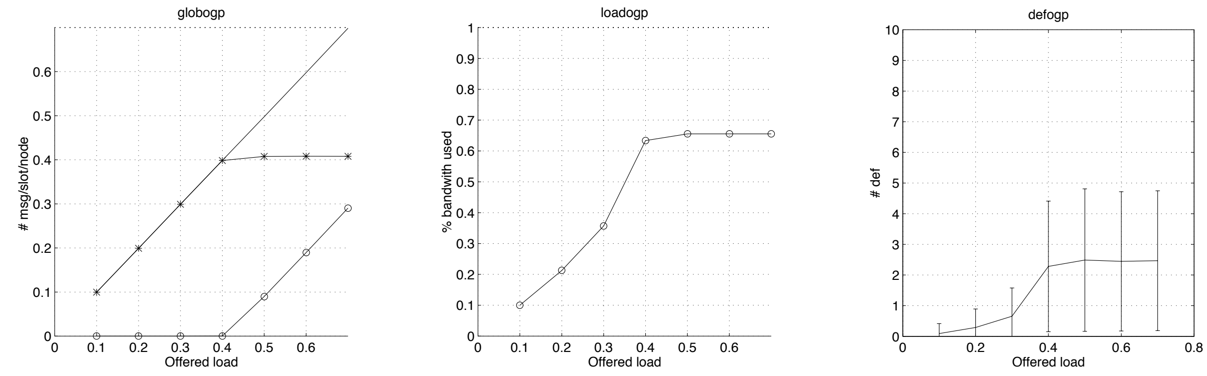
5.2.2 Property of our heuristic

The complexity of our heuristic is similar to the complexity of the preference optimization in the slotted case. Indeed, the complexity of our algorithm is dominated by the search of a maximum weighted matching in a bipartite graph, as in the slotted case. Even if such a search is time consuming, an efficient linear heuristic have been proposed in [25] that can be applied to implement 4×4 switches with few levels of preferences.

Note however that a run of our heuristic is *a priori* required for each packet entering the router. If the difference between the arrival times of two packets m and m' is too small, it is possible that the assignment of the first packet m is not completed before the assignment process of the second packet m' should start. A solution to this problem is to assign m' to the output link specified by our heuristic applied on m. This solution is however not suitable to the case where the heuristic determines the assignment for m_0 and the r next arriving packets $m_1, \ldots, m_r$, and a $r+1$th packet arrives before our heuristic completes. As a solution, packets $m_0, m_1, \ldots, m_r$ are routed according to the assignment of our heuristic, and m_{r+1} is routed according to its preference as in the standard unslotted deflection routing. This stays true if more than a single packet arrive too early. Actually, it is difficult to evaluate the influence of this phenomenon since it depends on many architectural parameters such as the computational power of the RCP, the number of input links, the distribution of the packet size, etc. Although we will see that our heuristic performs quite efficiently, it is of course not optimal as the following example shows. Let us consider a 2×2 switch with 2 input and output links called *North* and *South*. Assume a huge packet requesting the north output link arrives at the same time as a short packet requesting the south output link. Our heuristic will satisfy both requests. Assume now that a sequence of packets arrive briefly after the short packet, and that all these packets request the north link. As the huge packet is currently routed on the north link, the sequence of packets will be deflected. The maximum assignment would have deflected the huge and the short packet, and would have routed all the sequence of other packets according to their request. Of course, such a situation rarely occurs because the average inter-packet time is relatively large compared to the average packet size, and the delay loop used to "predict" future cannot contain more than a small amount of packets.

5.3 Experimental results

Figure 6(a) presents the throughput of the optimized deflection routing in unslotted networks. As we can check on the figure, the performance increases of about 35% in comparison with the standard unslotted routing. In the experimental context considered in this paper, it enhances unslotted routing almost at the same level as slotted routing.



(a) Throughput ($*$) and loss ($\circ$) **(b)** Link occupation **(c)**Average number of deflections

Figure 6: Throughput, link utilization and average number of deflections of optimized unslotted routing

The link occupation (Figure 6(b)) does not increase in comparison with the link occupation of the non optimized unslotted routing (Figure 4(b)). Therefore, 0.7 seems to be the "probabilistic saturation level" of the unslotted routing under the experimental context of this paper. In any case, the optimization of the unslotted routing does not allow to reduce the inter packet space. In some sense, it is a good news since it reduces the dependences between packets.

Figure 6(c) shows that our heuristic allows to strongly reduce the average number of packet deflections. As far as the number of deflections is concerned, it makes unslotted routing almost as good as slotted routing. The difference between these two modes is not only a consequence of our approximated solution of the NP-complete maximum assignment problem. Indeed, even the optimal solution would not eliminate the fact that, firstly, delay loops are of bounded length, and, secondly, the dependency chain between packets can be very long (optimally, the routing decision on a packet at time t should take into account the state of the network since it was turned on).

6 NP-completeness of the maximum assignment problem

In this section, we will show that the assignment problem is NP-complete. Note that an extended version of the assignment problem that is to take an arbitrary graph as the conflict graph would lead to an NP-complete problem in a trivial manner. Indeed, such a problem could be easily transformed from the clique-maximum problem, which is NP-complete [16]. The transformation to the clique-maximum problem cannot be applied since our conflict graphs are interval graph (for which the clique-maximum problem is polynomial [17]).

Theorem 1 *The following problem is NP-complete:*

MAXIMUM ASSIGNMENT (MA)

Instance: *a set of n output links, a set of $k+1$ packets (together with their arrival time, length, and preferences $p_{i,j}$, $i = 0, \ldots, k$, $j = 1, \ldots, n$, and an integer K;*

Question: *Does there exist an assignment ϕ of these $k+1$ packets to the n output links such that $\sum_{i \in \{0,\ldots,k\}} p_{i,\phi(i)} \geq K$?*

Proof. MA is in clearly in NP since we can check in polynomial time whether a correct assignment has a weight at least K. The proof of the NP-completeness is by transformation from the Maximum 2-Satisfiability problem, which has been proved to be NP-complete in [16]:

Maximum 2-Satisfiability (2-SAT)

Instance: a set U of variables, a collection C of clauses over U such that each clause c in C contains two literals, and an integer K'.

Question: Is there a truth assignment for U that simultaneously satisfies at least K' of clauses in C?

Let us describe a polynomial-time algorithm to transform any instance of 2-SAT (with clause neither of type $(x, \overline{x})$ nor of type (x, x)) in an instance of MA. Let an arbitrary instance of 2-SAT be a set U of ν variables $x_1, \ldots, x_\nu$, a set C of μ clauses $c_1, \ldots, c_\mu$, and an integer K'. Let us build an instance of MA. Each variable x in U is represented by a set $\mathcal{P}_x$ of 2μ packets, among which μ packets $m_{i,x}, i = 1, \ldots, \mu$ correspond to the literal x, and μ other packets $m_{i,\overline{x}}, i = 1, \ldots, \mu$ correspond to the literal $\overline{x}$: $\mathcal{P}_x = \{m_{i,x}, m_{i,\overline{x}}, i = 1, \ldots, \mu\}$. The arrival time[1] t_{i,x_j} of the ith packet of the literal x_j, $j \in \{1, \ldots, \nu\}$, satisfies $t_{i,x_j} = 20j\mu + 16(i-1)$. Similarly, the arrival time $t_{i,\overline{x_j}}$ of the ith packet of the literal $\overline{x_j}$, $j \in \{1, \ldots, \nu\}$, satisfies $t_{i,\overline{x_j}} = t_{i,x_j} + 8$. All packets are supposed to be of the same length 10 (see Figure 7).

The $\mu\nu$ packets $m_{i,x}$ are supposed to arrive by the same input link, say link 1, for all i, and all x. Similarly, the $\mu\nu$ packets $m_{i,\overline{x}}$ are supposed to arrive by the same input link, say link 2, for all i, and all x. The packets corresponding to a same variable arrive consecutively in a period of time of 20μ (see Figure 7).

Moreover, let us denote each clause c_i by a couple of literals (u, v), where $t_{i,u} < t_{i,v}$. Then each clause $c_i = (u, v)$ in C, $1 \leq i \leq \mu$, is represented by 6 packets denoted by $M_{i,u}$, $M_{i,v}$, and $M_{i,j}$, $j = \alpha, \beta, \gamma, \delta$, and the arrival times and lengths of these packets are set as follows:

- $T_{i,u} = t_{i,\overline{u}} + 2,\ L_{i,u} = 6$
- $T_{i,v} = t_{i,\overline{v}} + 2,\ L_{i,v} = 6$

[1]All times and length are given in an arbitrary time-unit.

- $T_{i,\alpha} = t_{i,\overline{u}} + 7$, $L_{i,\alpha} = 3$
- $T_{i,\beta} = t_{i,\overline{u}} + 9$, $L_{i,\beta} = t_{i,\overline{v}} - t_{i,\overline{u}} - 10$
- $T_{i,\gamma} = t_{i,\overline{v}} - 2$, $L_{i,\gamma} = 3$
- $T_{i,\delta} = t_{i,\overline{v}}$, $L_{i,\delta} = 3$

The 2μ packets $M_{i,u}$ $(1 \leq i \leq \mu)$ arrive on the same input link, different from link 1 and link 2. The 2μ packets of type α arrive on the same input link, different from all the previous links. Each of the 2μ packets of type β arrives on a different input link. Similarly, every packet of type γ arrives on a different input link. Finally, the μ packets of type δ arrive on the same input link. Therefore, there are $2\mu + 5$ input (and output) links in total labeled from 1 to $2\mu + 5$ $(n = 2\mu + 5)$. In the structure of our proof, packets $M_{i,u}$ and $M_{i,v}$ play the same role. Similarly, packets $M_{i,\alpha}$ and $M_{i,\delta}$ play the same role. Although packets $M_{i,\beta}$ and $M_{i,\gamma}$ look different, they also play the same role (to get a totally symmetric situation, it would be enough to balance the length of packet $M_{i,\beta}$ and packet $M_{i,\gamma}$).

There are $2\mu\nu + 6\mu$ packets in total in the system, that is $k + 1 = 2\mu\nu + 6\mu$. Related to this number of packets, we set $\rho = 2\mu\nu + 6\mu + 1$. Note that an assignment that would contain just one edge of weight ρ is of weight larger than any assignment that would contain only edges of weight 1.

Recall that the preference graph is a complete bipartite graph between the packets and the the links. For each (positive or negative) literal u, the preferences of the ith packet $m_{i,u}$, $i = 1, \ldots, \mu$, toward links 1 and 2 are set to ρ^2. Moreover, for each clause $c_i = (u, v)$ in C, $1 \leq i \leq \mu$, the preferences of packets $M_{i,u}$ and $M_{i,v}$ toward both output links 2 and $2i + 1$ are equal to ρ. The preferences of $M_{i,\beta}$ and $M_{i,\gamma}$ toward output links $2i + 1$ and $2i + 2$ are set to ρ^2. Finally, the preferences of $M_{i,\alpha}$ and $M_{i,\delta}$ toward output link $2i + 1$ are set to 1. All the other preferences are set to 0. Note that we are only using $2\mu + 3$ output links.

The construction is completed by setting $K = (2\mu\nu + 2\mu)\rho^2 + 2\mu\rho + K'$.

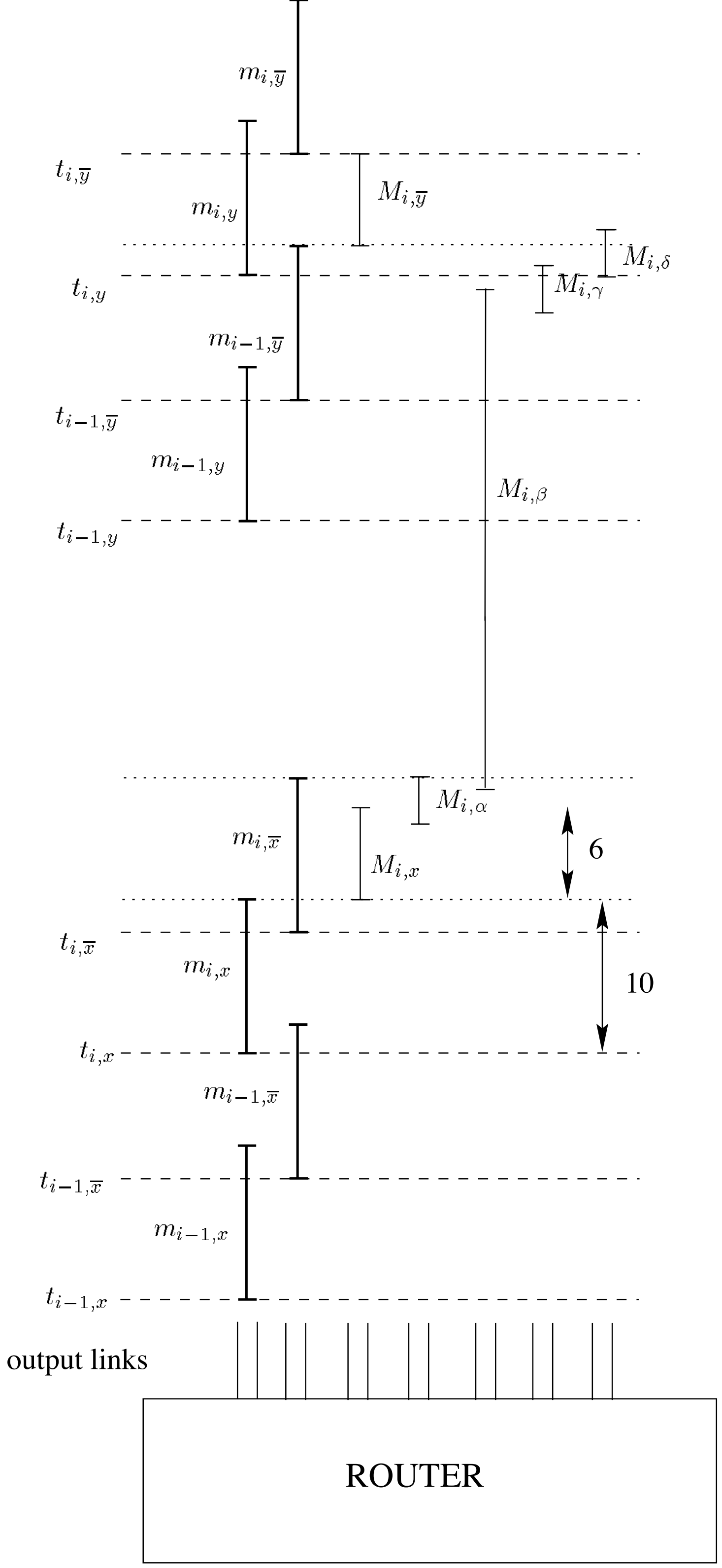


Figure 7: An example of the transformation in Theorem 1 where x, y are two variables of U, and $c_i = (x, \overline{y})$ is a clause of C.

Clearly this transformation is polynomial. Figure 7 is an example of this transformation. In order to prove the equivalence of 2-SAT and MA by this transformation, we derive below a general property that must satisfy the maximum assignment of the instance issue from the transformation.

Lemma 1 *Any assignment ϕ of value at least K, satisfies the following properties:*

1. *ϕ involves $2\nu\mu + 2\mu$ edges of weight ρ^2, and 2μ edges of weight ρ.*
2. *If a packet m is incident to an edge of weight at least ρ in the preference graph, then ϕ contains an edge incident to m that has a weight at least ρ.*
3. *All packets $m_{i,x}$, $i \in \{1, \ldots, \mu\}$, are assigned to the same output link. This link can be link 1 or link 2. The same property holds for packets $m_{i,\overline{x}}$. Moreover packets $m_{i,x}$ and $m_{i,\overline{x}}$ are not assigned to the same link.*

Proof. First, we prove Property 1. Let us focus on edges of weight ρ^2. Note that there are only $2\nu\mu + 2\mu$ packets incident to the edges of weight ρ^2. So, there are at most $2\nu\mu + 2\mu$ edges in ϕ whose weights are equal to ρ^2. Assume that in ϕ, there are a_2 edges of weight ρ^2, a_1 edges of weight ρ, and a_0 edges of weight 1. If $a_2 < 2\nu\mu + 2\mu$, then $a_0 + a_1\rho \geq \rho^2$. This is in contradiction with the definition of ρ, and with the fact that $a_0 + a_1$ is at most equal to the number of packets. Thus ϕ contains at least $2\nu\mu + 2\mu$ edges of weight ρ^2. We apply the same argument for the edges of weight ρ to show that the assignment ϕ contains exactly 2μ edges of weight ρ. And so property 1 holds.

The property 2 can be easily deduced from property 1.

For proving property 3, assume without loss the generality that ϕ contains the edge $(m_{1,x}, 1)$. Since there is a conflict between $m_{1,x}$ and $m_{1,\overline{x}}$, the edge $(m_{1,\overline{x}}, 2)$ is in ϕ. Since there is a conflict between $m_{1,\overline{x}}$ and $m_{2,x}$, the edge $(m_{2,x}, 1)$ is in ϕ. This argument applies successively for all i's and the property 3 holds.

$\blacksquare$

Based of the previous lemma, let us show that there exists a truth assignment for the variables in U such that at least K' of the clauses in C are simultaneously satisfied if and only if there exists an assignment ϕ of weight at least K.

Sufficient condition. Assume that there exists an assignment ϕ of weight at least K. Let us construct a truth assignment τ. For every packet $m_{i,x}$, $i \in \{1, \ldots, \mu\}$ and $x \in U$, we set $\tau(x)$ as follows:

- if $(m_{1,x}, 1)$ is in ϕ, then $\tau(x) = true$;
- if $(m_{1,\overline{x}}, 1)$ is in ϕ, then $\tau(x) = false$.

Let us count the number of clauses that are simultaneously satisfied by τ.

If clause $c_i = (u, v)$ is not satisfied by τ, then both $(m_{i,u}, 2)$ and $(m_{i,v}, 2)$ are in ϕ. Since packets $m_{i,u}$ and $M_{i,u}$ are in conflict, the edge $(M_{i,u}, 2)$ is not in ϕ. Note that this edge is of weight ρ. Property 2 implies that the edge $(M_{i,u}, 2i+1)$ is in ϕ. Moreover, since packets $M_{i,u}$ and $M_{i,\alpha}$ are in conflict, the edge $(M_{i,\alpha}, 2i+1)$ is not in ϕ. Similarly, the edge $(M_{i,\delta}, 2i+1)$ is not in ϕ. Thus, the weights of the two edges corresponding to the assignment of packets $M_{i,\alpha}$ and $M_{i,\delta}$ are both equal to zero.

If clause $c_i = (u, v)$ is satisfied by τ, then there are two cases to consider:

1. One and only one of the two literals is true. Assume without loss of generality that $\tau(u) = true$ and $\tau(v) = false$.
2. Both literals are true.

In case 1, since $\tau(v) = false$, then, by definition of τ, and thanks to property 2, both edges $(m_{i,v}, 2)$ and $(M_{i,v}, 2i+1)$ are in ϕ. Since $\tau(u) = true$, the edge $(m_{i,u}, 1)$ is in ϕ. If both edges $(M_{i,\alpha}, 2i+1)$ and $(M_{i,u}, 2)$ are in ϕ then the sum of the weights of the two edges corresponding to the assignment of packets $M_{i,\alpha}$ and $M_{i,\delta}$ is equal to 1, otherwise this sum is equal to 0.

In case 2, $(M_{i,u}, 1)$ and $(M_{i,v}, 1)$ are both in ϕ. Moreover, either $(M_{i,\alpha}, 2i+1)$ or $(M_{i,\delta}, 2i+1)$ is in ϕ, but not both. Indeed, Property 2 implies that one and only one of the edges $(M_{i,\beta}, 2i+1)$ and $(M_{i,\gamma}, 2i+1)$ is in ϕ. Assume without loss of generality that $(M_{i,\beta}, 2i+1)$ is in ϕ. Since there is a conflict between packets $M_{i,\beta}$ and $M_{i,\alpha}$, the edge $(M_{i,\alpha}, 2i+1)$ is not in ϕ. Thus, only one of the two edges $(M_{i,\alpha}, 2i+1)$ and $(M_{i,\delta}, 2i+1)$ is in ϕ. Therefore, the sum of the weights of the two edges in ϕ that are incident to $M_{i,\alpha}$ and $M_{i,\delta}$ is at most equal to 1.

Hence, the number of clauses satisfied by τ is at least the number of edges of weight 1 in ϕ. If a_0 denotes the number of edges of weight 1 in ϕ, then since the weight of ϕ is at least K, we have

$$(2\nu\mu + 2\mu)\rho^2 + 2\mu\rho + a_0 \geq (2\nu\mu + 2\mu)\rho^2 + 2\mu\rho + K',$$

and thus $a_0 \geq K'$. Therefore, the truth assignment τ for U simultaneously satisfies at least K' of the clauses in C.

Necessary condition. Assume that there exists a truth assignment τ for U that simultaneously satisfies at least K' of the clauses in C. We construct an assignment ϕ as follows:

- if $\tau(x) = true$, then $(m_{i,x}, 1)$ is in ϕ, $i = 1, \ldots, \mu$;
- if $\tau(x) = false$, then $(m_{i,\overline{x}}, 1)$ is in ϕ, $i = 1, \ldots, \mu$.

Moreover,

- for $i = 1, \ldots, \mu$, if $c_i = (u, v)$ is false, then $(M_{i,u}, 2i+1)$, $(M_{i,v}, 2i+1)$, $(M_{i,\beta}, 2i+1)$, and $(M_{i,\gamma}, 2i+2)$ are in ϕ;

- for $i = 1, \ldots, \mu$, if $c_i = (u, v)$ is true, then

  1. if only one of the two literals u and v is true, say u, then $(M_{i,u}, 2)$, $(M_{i,v}, 2i+1)$, $(M_{i,\alpha}, 2i+1)$, $(M_{i,\beta}, 2i+2)$, and $(M_{i,\gamma}, 2i+1)$ are in ϕ;
  2. if both literals u and v are true, then $(M_{i,u}, 2)$, $(M_{i,v}, 2)$, $(M_{i,\alpha}, 2i+1)$, $(M_{i,\beta}, 2i+2)$, and $(M_{i,\gamma}, 2i+1)$ are in ϕ.

By construction, there exists a truth assignment for U that simultaneously satisfies at least K' of the clauses in C if and only if there exists an assignment ϕ such that its weight is at least K. This completes the proof.

■

The following result is a direct consequence of Theorem 1:

Corollary 1 *The following problem is NP-complete:*

General Maximum Assignment (GMA)

Instance: *An interval graph $G = (V, E)$, a weighted bipartite graph $H = (V_1, V_2, F)$ where $V_1 = V$, and an integer K;*

Question: *Does there exist a subset $\phi \subset F$ such that (1) every vertex of V_1 is the extremity of exactly one edge in ϕ, (2) if two edges of ϕ are incident to the same vertex of V_2, say $e = (v_1, v_2)$ and $e' = (v_1', v_2)$, then the edge $(v_1, v_1') \notin E$, and (3) the global weight of the edge of ϕ is $\geq K$?*

Acknowledgments. The authors are thankful to Fabrice Clérot for many helpful comments on many aspects of our research on all-optical networks.

A Time scaling

We have considered two time scaling in order to separate the behavior of the network from the behavior of the applications using the network. The main purpose of these two ticks is to perform simulation on synchronous and asynchronous networks using exactly the same probabilistic law for the emission process. We have considered:

1. Simulation tick, or network tick; and
2. Processor tick, or emission tick.

At each simulation tick, we consider possible emission of packets at each node, and we route packets in the network. The traffic demand is simulated as follows. At each node, the decision to introduce or not a packet in the input queue is taken according to a probabilistic law, and destinations are chosen uniformly at random among the other nodes. Each node follows the same law. At each processor, the emission follows a Bernoulli law (when a processor sends, it sends exactly one packet). This Bernoulli law is in turn simulated by a Binomial law at the simulation tick. We denote by t_n (resp. t_p), the tick of the network (resp. of the processor). In our experiments, we have fixed $t_n = 0.1\mu s$, and $t_p = 20ns$. Note that other experiments done with a smaller simulation tick ($t_n = 20ns$) produced the same results.

Most of our experimental results are presented as a function of the load offered to the network. The offered load is expressed in packets per node and per *slot*. (In unslotted networks, the slot is an abstract measure expressing the maximum size of a packet.) The time slot is denoted by t_s. We have fixed the time slot at $t_s = 1\mu s$. Hence, to get a fixed offered load $\mathcal{L}$, we have forced the parameter of the Bernoulli law $B(\lambda)$ followed by the emissions to be $\lambda = \frac{\mathcal{L}}{t_s/t_p}$. Then the emission law of a network is $B(\lambda, t_n/t_p)$. This protocol produces the same emission law for both slotted and unslotted networks. Note that it would not have been the case if we would have followed the naive approach consisting of setting $t_n = t_s$ for synchronous simulation.

B Sporadic traffic

We mainly consider two different emission laws for two different kinds of experiments: Poisson traffic, and sporadic traffic. In the Poisson traffic, every processor follows the same Bernoulli law. This is the most commonly studied traffic in the literature.

In order to simulate a sporadic traffic, we have used an emission law denoted by $S(\mathcal{L}_g, p, \mathcal{L}_b, p')$ [13]. This law is nothing else than a two states Markovian chain. More precisely, each node is in two possible states called *ground* and *bursty*. These states alternate according to two probabilities p and p'. From the ground state, the probability to enter the bursty state is p. From the bursty state, the probability to enter the ground state is p'. In the ground state, the emission law is Poisson. In the bursty state, we allow processors to send a large number of packets within one slot (such packets will be stored in the input queue). When a processor is in the bursty state, its offered load is of average $\mathcal{L}_b > 1$ (to be compared with the global offered load in the Poisson traffic which is always strictly less than 1).

As in [21], we have considered that bursty traffics are mainly caused by ftp-data-like applications. Moreover, whatever the load of the network is, a burst offers the same characteristic. Thus, we decide to set $\mathcal{L}_b = cst$, independently from the global load $\mathcal{L}$. For the same reasons, the probability p' to get out of a bursty application is not related to the global load, and thus it is set as a constant. The ground emission rate $\mathcal{L}_g$ is defined as a linear function of the offered load of the network $\mathcal{L}$. Indeed, the ground traffic is induced by telnet-like connections [21] whose number grows linearly with the number of running applications. We have set $\mathcal{L}_g = c\,\mathcal{L}$. Note that c should not be larger than the saturation threshold of the network. The constant c is hence set to 0.3. For a given offered load, the probability p is fixed to $p' \frac{\mathcal{L}(1-c)}{\mathcal{L}_b - \mathcal{L}}$ so that the mean of the law $S(\mathcal{L}_g, p, \mathcal{L}_b, p')$ is $\mathcal{L}$. Thence, in our sporadic model, an increase of the load will be induced by an increase of the frequency at which we enter in the bursty state.

We have fixed (somewhat arbitrarily) the value of $\mathcal{L}_b$ at 5 (smaller bursts would not be significant, and larger bursts would saturate the input queues). We also set $p' = 0.02$. This value was fixed according to the value of $\mathcal{L}_b$. It implies that a burst will fill up the input queue with 25 packets on average, that is with 25% of the size of the queue. One can see on Figure 8 that Poisson and sporadic traffics are indeed very different.

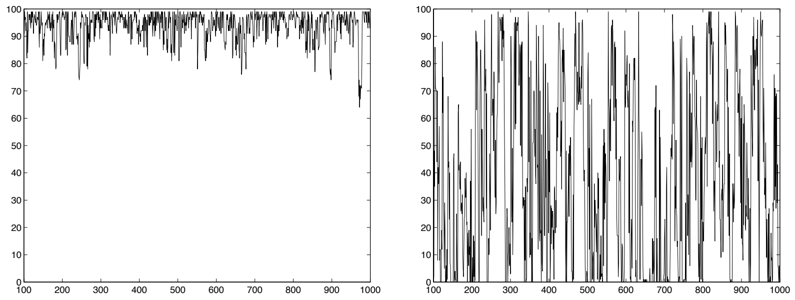


Figure 8: Poisson traffic (on the left) versus sporadic traffic (on the right): number of packets in a queue as a function of the time.

References

[1] Traces in the internet traffic archives. http://ita.ee.lbl.gov/html/traces.html.

[2] Acampora, A. S., and Karol, M. J. An overview of lightwave packet networks. *IEEE Networks* (Jan. 1989), 29–41.

[3] Acampora, A. S., and Shah, S. I. Multihop lightwave networks : A comparison of store–and–forward and hot–potato routing. *IEEE Trans. on Communication 40*, 6 (June 1992), 1082–1089.

[4] Badr, H., and Poder, S. An optimal shortest-path routing policy for network computers with regular mesh-connected topologies. *IEEE Trans. on Computers 38*, 10 (Oct. 1989), 1362–1371.

[5] Baransel, C., Dobosiewicz, W., and Gburzynski, P. Routing in multihop packet switching : Gb/s challenge. *IEEE Network* (May 1995), 38–61.

[6] Blumenthal, D. J., Prucnal, P. R., and Sauer, J. R. Photonic packet switches : Architectures and experimental implementations. In *Proc. of the IEEE* (Nov. 1994), vol. 82(11), IEEE, pp. 1650–1667.

[7] Borgonovo, F., and Fratta, L. Deflection networks: Architectures for metropolitan and wide area networks. *Computer Networks and ISDN Systems 24*, 2 (1992), 171–183.

[8] Borgonovo, F., Fratta, L., and Bannister, J. Unslotted deflection routing in all–optical networks. In *Globecom'93* (1993), IEEE, pp. 119–125.

[9] Borgonovo, F., Fratta, L., and Bannister, J. On the design of optical deflection–routing networks. In *Infocom'94* (1994), IEEE, pp. 120–129.

[10] Brassil, J., and Cruz, R. Bounds on maximum delay in networks with deflection routing. *IEEE Transactions on Parallel and Distributed Systems 6*, 7 (July 1995), 724–732.

[11] Chich, T. *Optimisation du routage à déflexion pour les réseaux de télécommunications métropolitains*. PhD thesis, Ecole Normale Supérieure de Lyon, Dec. 1997.

[12] Chich, T., and Fraigniaud, P. An extended comparison of slotted and unslotted deflection routing. In *ICCCN'97* (1997), IEEE, pp. 92–97.

[13] Collange, D. Reclassification des modèles de trafic. Tech. rep., CNET, France, Jan. 1995.

[14] Fang, C., and Szymanski, T. An analysis of deflection routing multi–dimensional regular mesh networks. In *Infocom'91* (1991), vol. 3, IEEE, pp. 859–868.

[15] Fehrer, J., Sauer, J., and Ramfelt, L. Design and implementation of a prototype optical deflection network. In *Sigcomm'94* (1994), vol. 24, ACM, pp. 191–200.

[16] Garey, M. R., and Johnson, D. S. *Computers and intractability: a guide to the theory of the NP-completeness.* Freeman, 1979.

[17] Golumbic, M. C. *Algorithmic graph theory and perfect graphs.* Academic press, 1980.

[18] Kuhn, H. The Hungarian method for the assignment problem. *Naval Research Logistics Quaterly 2* (1955), 83–97.

[19] Leland, W., Taqqu, M., Willinger, W., and Wilson, D. On the self-similar nature of ethernet traffic. *IEEE/ACM Transactions on Networking 2*, 1 (Feb. 1994), 1–15.

[20] Maxemchuk, N. F. Routing in the MSN. *IEEE Trans. on Communication 35*, 5 (1987), 503–512.

[21] Paxson, V., and Floyd, S. Wide area traffic : The failure of poisson modelling. *IEEE/ACM Transactions on Networking 3*, 3 (June 1995), 226–244.

[22] Renaud, M., Masetti, F., Guillemot, C., and Bostica, B. Network and system concepts for optical packet switching. *IEEE Communications Magazine* (Apr. 1997), 96–102.

[23] Seo, S.-W., Bergman, K., and Prucnal, P. R. Tranparents optical networks with time-division multiplexing. *Journal of Selected Area in Communications 14*, 5 (June 1996), 1039–1051.

[24] Wang, Z., and Crowcroft, J. Qos routing for supporting multimedia applications. *IEEE Journal on Selected Areas in Communications 14*, 7 (1996), 1228–1234.

[25] Wong, J. S., and Kang, Y. Distributed and fail-safe routing algorithms in toroidal-based metropolitan area networks. In *Computer Networks and ISDN Systems* (1989/90), vol. 18, pp. 379–391.