Comparison of fixed size and variable size packet models in an optical ring network: Algorithms and performances

Dominique Barth, Johanne Cohen, Lynda Gastal, Thierry Mautor, Stéphane Rousseau

Comparison of fixed size and variable size packet models in an optical ring network:

Algorithms and performances

D. Barth[1], J. Cohen[2], L. Gastal [2], T. Mautor[1], S. Rousseau[1]

1:PRiSM, UMR 8144, Univ. Versailles-Saint Quentin, 45, Av. des Etats-Unis, 78035 Versailles Cedex
2: LORIA, UMR 7503, B.P.239, 54506 Vandoeuvre Lès Nancy, France.

Abstract: In this paper, we compare the use of two packets models in slotted optical ring networks: a model where each message has to be routed in consecutive slots, and a model where the slots can be routed independently. We first focus on the algorithmic complexity of the related problems. Then, we give the results we obtain with an OMNET simulator in terms of messages delay and jitter and of communication resources really used at each step.

1. Introduction and description of the problem

In this paper, we compare the communication efficiencies of a simple optical ring network under two packet models: a fixed size and a variable size models. Consider a ring network *R* connecting *N* nodes with only one slotted wavelength (with *K* slots on each link). This study is presented in the context of the DAVID project of the 5[th] European PCRD. At each communication step, each node "sees" the slot of the ring located on him. If the data contained by this slot is intended for him, the node reads it and removes the data from the slot (it becomes empty). If the slot is empty, the node can use it to send a new message (or a part of the message) to its destination.
The purpose of this study is to compare variable size packet and fixed size packet from two points of view:

- What is the difficulty of determining an optimal scheduling in these two models?
- What are the performances of each model in terms of packet delay and jitter?

As we want the difficulty to be linked only to the models behaviours and not to the network control, to focus on these two questions we assume a very simple optical network, i.e., a slotted ring with only one wavelength. Each node could have some messages to send to another nodes. Each such message M is characterised by the origin node: **or(M)**, the destination node: **dest(M)**, the size in number of slots: **sz(M),** the time at which it becomes available in or(M): **dispo(M)** and the distance (number of slots=number of steps) from or(M) to dest(M): **dist(M)** = *K * ((dest(M) – or(M)) mod N).* We also consider the time **First(M)** (resp. **Last(M)**) at which its first slot (resp. last slot) is sent on the ring. Different measures can be defined and applied on each message M:

- The delay, defined by **Delay(M)=(dist(M)+Last(M)-dispo(M)),** represents the time between the arrival of the message on the origin node and the end of its transmission, i.e. the end of its reading on the destination node.
- The jitter, defined by Jitter **(M)=(Last (M)-First (M)),** represents the time between the beginning and the end of the emission of a message.
- The overdelay, defined by **OvDel(M)=Last(M)-Dispo(M)-sz(M)+1,** represents the difference between the delay of M and its minimal possible delay (equal to dist(M)+sz(M)-1). This measure, inspired from some works on scheduling, is interesting to compare the delays of packets of different size.

The **variable size packet (VSP) model** consists here in saying that the jitter is a constraint: for each message M, we want **Jitter(M)=sz(M)-1**. This implies that all the slots of a same message have to be **sent contiguously** on the ring. Let us remark that, in this model, the overdelay of a message M is equal to : **OvDel(M) = First(M)-Dispo(M)**.
The **fixed size packet (FSP) model** says that all the slots of a same message can be sent independently the ones from the others. Thus, the jitter is in this case an evaluation parameter of the behaviour of the network.

The paper is organized as follows. First, we present the theoretical problems we focus on, especially in a static centralized model (i.e., where the (finite) set of messages to be sent is known in advance). Then, we give some simulation results under a (more realistic) distributed online model.

2. Considered versions of the problem and complexity

As previously indicated, several objective functions can be considered for this problem (Delay, Jitter, Cmax, …). We have decided to focus in this section on two of these evaluation parameters. The first criterion (Cmax) is the classical makespan criterion that consists in minimizing the number of steps needed to send all messages to their destination. The second criterion called OvDel consists in minimizing the maximal overdelay of the messages.
In order to formalize these two problems, we consider static instances: each node contains a finite and given set of messages. Moreover, we consider the centralized model where each node has a global vision of the network. In terms of complexity, we have thus to consider the following problems :

Problem MS-VSP-Scheduling
Given: A ring *R*, a set S of messages, an integer B.

Question: Does there exist a scheduling of S on R such that all the messages have reached their destinations after at most *B* steps, with the constraint *Jitter (M)=sz(M)-1* for each message *M*?

Problem MD-VSP-Scheduling

Given: A ring R, a set S of messages, an integer B.

Question: Does there exist a scheduling of S on *R* such that the maximal OverDelay *OvDel(M)* over all messages *M* is less or equal to *B*, with the constraint *Jitter(M)= sz(M)-1* for each message *M*?

Of course, we have the same problems for the FSP model called **MS-FSP-Scheduling** and **MD-FSP-Scheduling** (the constraint on the jitter is removed). Problem **MS-VSP-Scheduling** (see [1]) and **MD-VSP-Scheduling** are NP-complete (reduction from the 3-partition problem). Moreover, for the FSP model, we conjecture that the both problems are NP-complete.

Of course, the different problems previously defined are scheduling problems in which we have to schedule the sending of the different messages on the ring. Moreover, the links with classical scheduling formulations can be reinforced by seeing the messages as tasks and the nodes as machines. However, let us underline that these problems have a very original particularity in comparison to classical scheduling problems: it is not the end of the execution of a task on a machine that activates the beginning of the same task on the following machine, but K units of time. Consequently, the same task can be executed simultaneously on different machines.

Using the classical scheduling techniques, we can describe two approximation algorithms: one for the centralized model and the other for distributed model. We focus on a makespan criterion. The scheduling *S* is built according to the following policy: if a node has a message of length *l* to send and there are at least *l* consecutive slots free, the node puts his message into. Let L_{max} be the maximal length of all messages. Note that all messages having the same origin node can be sorted using arbitrary order. Let $C_{max}(S)$ be the makespan of schedule *S* and $C_{max}(OPT)$ be the makespan of the optimal schedule.

For the VSP model, the quality of this scheduling *S* is:

$$C_{max}(S) \leq (L_{max} + 1)\, C_{max}(OPT)$$

For the FSP model, this schedule is a 2-approximation, that is, $C_{max}(S) \leq 2\, C_{max}(OPT)$.

It gives an indication of how well the heuristic is guaranteed to perform in the worst case.

Now, we focus our analysis on the distributed model with assuming that the maximal length of all messages is less than a constant L_{max}. Each node only knows its own set of messages to be sent. We define a frame as L ($L_{max} \leq L$) consecutives slots on the ring. Moreover, the first slot of each frame contains the number of free slots in this frame. Thus, ring R is cut into $(NK)/L$ frames (recall that *N* is the number of nodes and that *K* is the number of slots on each link). Now, when a frame crosses over on a node, if this node has a message of length *l* to send and if the frame has at least *l* consecutive slots free, the node puts his message into this frame. The quality of this scheduling *S* is for the VSP model:

$$C_{max}(S) \leq (2L + 1)\, C_{max}(OPT)$$

3. Algorithmic and Simulation studies

After the theoretical results presented in the previous section, this section is much more applied and consists in simulating and studying the behaviour of the network. For this, we consider now a distributed online model in which each node has to decide locally for the messages it sends and where new messages could appear at any time on any node. These messages are stored in local buffers (of infinite size). We begin by describing the distributed algorithm we have developed. Then, after a description of our simulation model, we analyse the influence of the length and the distribution of the messages on the utilization ratio of the network and on the average delay of the messages.

3.1 Distributed Control for MD-VSP-Scheduling

Let us first remark that the distributed protocol is very simple in the FSP model: as soon as a slot is free, the node can use it to send a part of one of its messages. Concerning the distributed MD-VSP-scheduling model given in the previous section (with the constraint of contiguity), we consider that a node receiving a message of S slots creates a free zone of S slots. Similarly, a node putting a message of L slots into a free zone of S>L slots creates after it a free zone of S-L free slots. The consequence of such a basic distributed control under the OPADM model is that the ring can become made of a cyclic sequence of free zone steps, each one too short to be used by any node. Consequently, we have to enhance this distributed control in order to be able to merge consecutive free zones.

When a node is in front of a free zone, it tries to put one of its messages into it. However, it is possible that this free zone is not large enough to put any message into. In such a case, the node can reserve this zone in order to merge it with another one. Thanks to this reservation, the node ensures itself to find it free again in the next turn. Another node can use a reserved zone to send a message if the owner (the node that has reserved this zone) is not between the source and the destination. If a node finds adjacent free zones reserved by it, it merges all of them into one.

A node *N* can reserve a free zone *Z* under the following conditions:

- *N* has at least one message to send (too large for the free zone),
- *N* has not currently any other free zone reserved or *Z* is adjacent to the zone it has reserved,
- *Z* has not been reserved yet, with the exception where N holds the priority. In this last case, it can reserve a free zone yet reserved by another node.

Without this priority, it is still possible to fall in a locked situation where each zone is reserved by a different node with no possibility to merge two free zones.

At any unit of time, a single node holds this priority. This node changes periodically in order to ensure a good balance. Indeed, if the priority holder never changed, this node would be favored as it would be the only one able to merge adjacent free zones and then the first which is able to use them.

This distributed algorithm ensures that it will be possible to send any message of any size. Indeed we have proved that this algorithm ensures merging at least two contiguous free zones in two turns, or at least three contiguous zones in **3 + [number of stations-1]/(number of stations)** turns, in the case where the priority changes during the two turns.

3.3 Simulation results

- **The simulation model:**

We have considered a ring of *10* nodes and *9* slots between each one. So, *the nodes to send their messages can use 90 slots*.

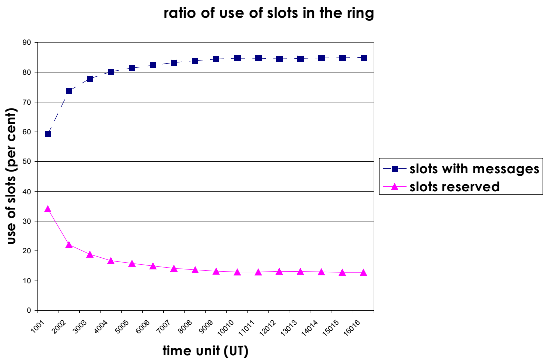
- **The traffic model:**

The messages are defined by different parameters (origin, destination, size, release date). The generation of new messages is simultaneous on all the nodes and occurs with a given and constant periodicity called inter-arrival-time. During one generation, one message is created on each node. The destination of each message is chosen uniformly among all the stations. The length may be small (2 slots), or large (7 or 16 slots). This corresponds to the fact that in the DAVID project, it has been identified that there are mainly two classes of messages: the short ones and the long ones (video data). So, in our model, the length follows a MMUP2 distribution (Markov Modulated Uniform Process with two states). Keeping the same average length of messages, we have realized simulations for different ratios of small and large messages in order to compare different uses of the network.

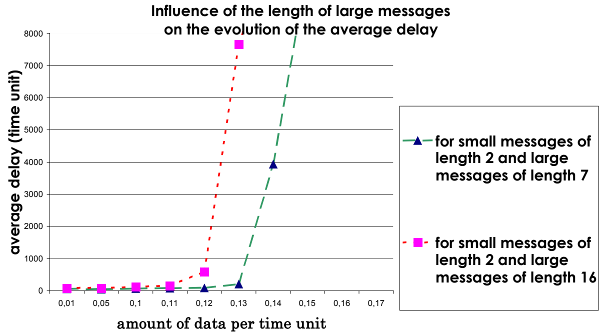
Results analysis

Even if a lot of measures become possible with such a simulation model, we have focused on two parameters. The first one is the ratio of occupation of slots by messages or by reservation. The aim is to compare the ratio of use of slots respectively for the VSP-Scheduling and for the FSP-Scheduling. The second one is the average delay of the messages in order to study the influence of the length of messages on the average delay.

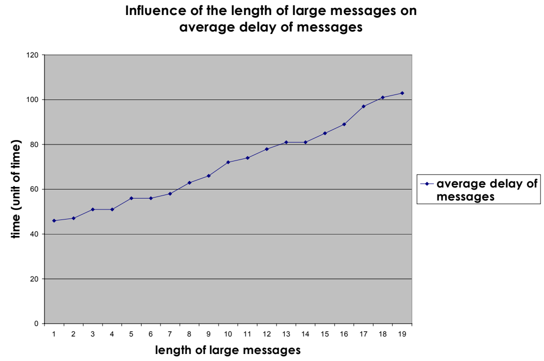
1. The ratio of used slots and reserved slots.
   As illustrated on the following figure, the ratio of slots used by messages increases at the beginning of the simulation to quickly converges to a limit. This limit varies between 70 and 90%, according to the instances. This shows that the constraint of contiguity requires around 20% for the control (reservation of the free zones) whereas the FSP model no requires control.



2. The influence of the length of the largest messages on the average delay:



The two last figures illustrate that, for a given data flow (average amount of data per unit of time), the average delay increases with the length of large messages. The first figure gives the delays for two lengths of the large messages, whereas the second figure shows how this delay increases in function of this length. When we have very large messages, we need to merge adjacent free zones more frequently. However, this increase in the delay is rather low and does not change a lot the ratio of used slots that remains superior to 70%.



5. Conclusion

We have shown that finding optimal packets scheduling is difficult in both FSP and VSP models. For this last one, in an online and distributed context, there is also the problem of merging free consecutive free zones to avoid live-locks. The cost of a protocol realizing this concatenation can be evaluated in terms of resource use, and the one we give here insures that at each step, about 80% of the slots are used to carry data, and this is not to the detriment of the delays of large messages.

Let us finally note that we also make a similar study on a network made of many optical rings with a same common node making commutations between them (such a network is studied inside the DAVID project).

6. References

[1] J. Carlier: " *One machine problem*". European Journal of Operational research 11(42-47), 1982.

[2] M. Garey and D. Johnson: " *Computers and Intractability: A Guide to the Theory of NP-Completeness* ", W.H. Freeman, 1979.

[3] M. Pinedo: "*Scheduling: theory, algorithms and systems*". Prentice Hall ed., 2002.

[4] G. Tell. "*Introduction to distributed algorithms*". CamBridge Int. series on Parallel Computation Vol.1, Cambridge University Press, 1984.

[5] V.V. Vazirani : " *Approximation algorithms* " . Springer Verlag ed., 2001.