Utility function based packet scheduling over DVB-S2

E. Chaput*, M. Verloop †, A.-L. Beylot *, C. Baudoin‡
*Université de Toulouse ; INP ; F-31071 Toulouse, France
†CNRS ; IRIT ; F-31062 Toulouse, France
‡Thales Alenia Space
*†Email: {name}@enseeiht.fr, ‡ cedric.baudoin@thalesaleniaspace.com

***Abstract*—Satellite Digital Video Broadcasting is deeply changing : the next generation is dedicated to packet based communications and introduces (like many modern physical layers : WiMax, HSDPA, ...) fade mitigation techniques leading to variable throughput. Such tremendous changes need to be taken into account and the scheduling entity needs to be revisited.**

The purpose of this paper is to investigate utility function based algorithms. Such techniques have already been studied but never within this context. DVB-S2 frames can encapsulate numerous IP packets, can offer variable payload length as well as variable transmission time, because of ACM techniques. A more general approach is needed to encompass theses properties. As a consequence, the number of solutions among which a scheduler has to find the better is increased. We will then show that the algorithm implemented is an important issue.

***Index Terms*—Satellite, DVB-S2, scheduling, ACM, utility function.**

I. Introduction

Satellite networks present an efficient mechanism called adaptive coding and modulation (ACM) technique adopted by DVBS-S2 standard which matches transmission parameters to the channel conditions [1], [2]. The first generation of DVBS-S standards only supported data transport using MPEG-TS. The new encapsulation called generic stream encapsulation (GSE) allows a backward compatibility with MPEG-TS as well as generic encapsulation for carrying arbitrary packets of variable length. GSE protocol reduces overhead by using the fragmentation and increases the throughput gain by up to 15 % [3] compared to MPE encapsulation over MPEG-TS by using packets of variable length witch matches IP functionalities.

The main problem we address here is the one of packet scheduling over a ACM link such as a DVB-S2 satellite downlink. A previous study has shown that a very basic scheduler could lead to good overall performance but may induce unfairness [4]. The aim here is to be able to tackle both the application level QoS requirements (and then some kind of fairness) and the spectral efficiency optimisation within the scheduler.

We believe that utility functions could be of great help for this purpose. It has already been used for packet scheduling [5] [6], but never adapted to this specific context.

The remainder of this paper is organised as follows. Section II will settle the general context and challenges, section III will briefly discuss utility functions. We will then be able to settle the problem in section IV and to describe a scheduler based on the resolution of this problem in section V. We will then give some results in section VI before concluding in section VII.

II. System description

Let us assume a simple architecture in which the forward link (on which we will focus) is implemented by a DVB-S2 channel, and a return link could be implemented, *eg* through a DVB-RCS channel.

The problem we tackle is the one of packet scheduling on the forward link. Each packet is supposed to be part of a unicast or multicast stream (a stream could be a single application stream or a larger aggregation). Any quality of service enforcement will be done on a stream basis.

Packets will be encapsulated in BBFRAMEs that will be sent in numerous consecutive slots, according to an encapsulation scheme shown in figure 1 and based on the DVB-S2 low layers [2] and the GSE protocol [3].

ACM is implemented with the help of BBFRAMEs and lower layers. Each BBFRAME is associated to a modulation and coding scheme (hereafter called a MODCOD). The payload size depends on the coding scheme and the transmission duration depends on the modulation scheme.

When a packet has been scheduled, it has to be sent in a BBFRAME associated to a MODCOD that suits to the receiver(s) channel conditions. It may be the MODCOD selected by the ACM management system, but it could be

978-1-4673-3122-7/13/$31.00 ©2013 IEEE

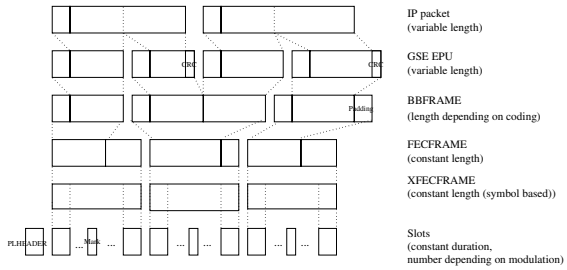

Fig. 1. DVB-S2 encapsulation scheme

a MODCOD with a less efficient spectral efficiency (we call this technique *reclassification*).

The choice of the next BBFRAME to transmit has then consequences on the packets that can be sent in this frame. However, the choice of the next packets to send has also some consequences on the BBFRAME that can be used, as far as the receiver(s) must be able to decode the BBFRAME. These relations alos lead to the need for a more sophisticated scheduling algorithm.

BBFRAMEs can be large (up to 8 Kbytes) and with the help of GSE, multiple packets can be sent in the same BBFRAME (fragmentation is also available, as illustrated in figure 1).

This scheduling problem is thus characterised by some interesting properties (ACM, large frames, fragmentation, reclassification opportunity, ...). The scheduler also needs, in addition, to enforce a tradeoff between stream level quality of service requirements and spectral efficiency (or, more generally, resource management).

III. Utility function specification

A. Utility functions

In order to take into account both user's perception of the quality of service and the network provider's interest, utility functions have been proposed [5] [6] [7] [8] [9] [10].

A utility function $U(s)$ describes how an application can benefit (or suffer) from a service s provided by the network. The service s may be a simple scalar metric or a tuple encompassing many parameters such as throughput, delay, loss rate, ...

For a given application i, the system should try to increase the utility $U_i(s_i)$ through the appropriate s_i setting. For this purpose, the system should allocate more resources (as denoted by s_i) to this application. Of course, because of the system limitations, the maximum value of $U(s)$ can not be enforced for all the applications at the same time.

The basic idea is then to allocate resources among the applications in order to improve the sum of the utility functions : $\mu = \sum_i U_i(s_i)$. There are obvioulsy some constraints on such an allocation as far as resources are limited.

For the sake of simplicity, we will focus here on throughput, and then utility functions will be based on this single QoS parameter. Introduction of other components such as jitter or loss rate is an interesting challenge.

We will then describe utility function for stream i as $U(r_i)$ where r_i is the throughput allocated to stream i and evaluated, for example, through a moving average encompassing previous scheduling decisions.

B. Spectral efficiency optimisation

The very first scheduling strategy that could be implemented is the one leading to the maximal spectral efficiency :

$$U_{se}(r) = k.r$$

where k is a constant that could be used as a weight in order to enforce some kind of priority.

C. Improving fairness

The main problem with spectral efficiency optimisation is its unfairness. This could then be improved with a utility function leading such as throughput variation consequences are larger for lower bandwidth values. We can use

$$U_f(r) = k.log(r)$$

D. Best effort streams

Propositions have been made to define a utility function for best effort traffic such as the following [11] :

$$U_{be}(r) = r + (1 - e^{k.(r_{min} - r)})$$

Here, r_{min} is the minimal bandwidth required by a given stream and k, here again, can help to introduce some form of priority.

IV. Problem statement

Let us now formalise the problem such a scheduler will have to solve in order to maximise μ as defined in section III (the sum of utility functions).

A. General problem specification

Transmission through the DVB-S2 link can be seen as a sequence of frames $F(n), n \geq 1$. The aim of the scheduler is to decide which packets will be sent in the next BBFRAME.

The optimisation problem we have to solve for every n is thus the following

$Max_{s_n=\{m(n),L_1(n),\ldots,L_{k(n)}(n)\}} \sum_{i\in[1\ldots k(n)]} U_i(r_i(t(n)))$
$\sum_{i\in[1\ldots k(n)]} L_i(n) \leq P_{m(n)}$
$\forall i \in \{1,\ldots,k(n)\} L_i(n) \geq 0$

where

- k is the number of clients (streams) ;
- r_i is the throughput allocated to client i ;
- $L_i(n)$ is the number of bytes from client i in ;
- U_u is the utility function associated to client i BBFRAME n ;
- P_m is the payload size in a BBFRAME send through MODCOD m ;
- $m(n)$ is the MODCOD chosen for BBFRAME n.

$m(n)$ has to be chosen among a finite (and small) number of available MODCODs depending on the scenario.

The departure time is then given by $t(n) = t(n-1) + \tau(n)$. However, from now on, we will not need to use $t(n)$ anymore, and so we will use $r_i(n)$ as a shorthand for $r_i(t(n))$.

B. Parameter evaluation with adaptive modulation

If $L_i(n)$ bytes from stream i are sent in BBFRAME $F(n)$, then the throughput assigned to this stream during this slot (of duration $\tau(n)$, the transmission time of $F(n)$ which depends on the associated MODCOD) is given by $b_i(n) = \frac{L_i(n)}{\tau(n)}$.

However, throughput can not be evaluated through this simple short term memoryless formula. A classical solution is given by the use of an exponential moving average, but such a tool should be used with some precautions within the context of this study. We have shown through many simulations that as far as the frame transmission time is not constant, a constant smoothing factor leads to a bad throughput evaluation and thus to a poor scheduler behaviour.

The smoothing factor must thus adapt to the frame duration, then we use the following formula

$$r_i(n) = \alpha^{\tau(n)}.r_i(n-1) + (1-\alpha^{\tau(n)}).b_i(n)$$

where α is close to one.

C. Looking for an optimal scheduling

With optimisation techniques, it has been proven that a system optimum could be reached with the help of an equilibrium between users and network choices [6]. Convergence to such an equilibrium remains a challenge.

Our scheduling problem consists in the search for a BBFRAME achieving such an optimisation every time the channel is free. Such an interpretation, however, introduces several problems, because of the specific properties of our context. The first one is the problem of the metrics evaluation, already described in section IV-B.

The second problem introduced by such an interpretation has been described in [11]. There is definitely no reachable equilibrium between user requests and network resources as far as both of them may vary with time. The theoretical optimum changes over time and has then to be reconsidered for each scheduling decision. As a consequence, this theoretical optimum is probably unreachable, because each scheduling decision has a very short impact on the metrics and then on μ.

For this purpose (following a classical approach [8] [11]), assuming the stream j is served during slot n so that it throughput is increased by ϵ (and throughput of other streams are not changed), we can rewrite the function to be maximised :

$$F_j(n,\epsilon) = U_j((1+\epsilon).r_j(n-1)) + \sum_{i\neq j} U_i(r_i(n-1))$$

Taking the derivative in zero :

$$F'_j(n,0) = r_j(n-1).U'_j(r_j(n-1))$$

We will then use a steepest ascent method to find the "direction" to the maximal value for our optimisation problem. We will then serve clients according to this direction.

However, here again, our precise context needs to be taken into account : a single BBFRAME can carry several packets for multiple streams, so we need to maximise the sum of $F'_j(n,0)$ for all j.

For each slot n, we will thus search for maximisation of

$$\sum_i U'(r_i(n-1)).\frac{L_i(n)}{\tau(n)}$$

V. Packet scheduling

Each time the link is ready and the BBFRAME has to be built and sent, the scheduler has to find the tuple $(m(n), L_1(n), \ldots, L_k(n))$ that maximises the previous sum. The value $m(n)$ determines the MODCOD and then the BBFRAME duration, $\tau(n)$. Each value $L_i(n)$ determines the number of bytes (and then of packets) from stream i that will be encapsulated within this next BBFRAME.

The search for such a tuple may be time consuming and then an exhaustive search may not be affordable, so simple algorithms may be implemented. With the

help of simulation, however, exhaustive searches can be implemented in order to study performances for utility function based algorithm and to settle upper bounds for heuristic performance.

A. Resolution through a basic algorithm

We have implemented in our simulator a scheduler based on a single evaluation of utility functions. Basically, it will fill a BBFRAME with packets from the queue with the largest U_i' function. If some room is available, it will search for the second largest U_i' and so on.

B. Resolution through the knapsack model

In order to evaluate the quality of the results given by the previous scheduler, we ran simulations with a scheduler implementing a knapsack problem resolution.

The knapsack is a traditional combinatorial optimization problem. The objective is to determine the optimal choice among all possible combinations of objects to put in a bag. Each object is characterised by a value and a weight. The solution is then the set of objects with a total weight lower than or equal to the bag capacity that cumulates the largest value.

We have chosen to implement two algorithms. The first one is an exhaustive research of the best solution. However it is not always possible to use such an algorithm, even in a simulator, so we also implemented a more classic algorithm based on dynamic programming. We assumed that a knapsack solution for a bag of capacity K can be built as the max, for any object weight w, of the solution for a bag of capacity $K - w$ with the most valuable object of weight w.

In our problem, the bag is the BBFRAME and the objects are the packets. As far as packets in multiple FIFO queues, some of them may be available or not depending on the previous choices. A consequence is that the assumption made for the non exhaustive implementation does not hold.

VI. Simulations and results

A. Simulated scenarios

In our simulated system, the receivers are divided in four groups, depending on their channel conditions. In a first scenario (called "uniform"), the trafic is evenly spread among the four groups, in order to alleviate the bias introduced by the throughput. In the "clear sky" scenario (which is more realistic), most of the trafic is sent to the group with the best channel condition.

B. Previous results

During a previous study, we noticed that a scheduler based on static parameter (such as spectral efficiency) is, of course, unfair, and that different dynamic parameters (like queue bit length, BBFRAME fill rate, ...) give very similar results [4]. Most interestingly, we have shown that both reclassification and fragmentation can help improving the system performance.

However, no distinction was made between the multiple streams. That means that quality of service was not enforced. The aim of this work is then to be able to tune fairness.

A last comment on the previous results is that there is a need to implement a scheduling strategy encompassing both spectral efficiency and preformance (including fairness). The first one can be efficiently implemented by static parameter based scheduling, while the second one seems easy to implement with a dynamic one.

C. Some results and analysis

1) Algorithms comparison: The exhaustive implementation is only usable for low value of the system load, and without reclassification. On the other hand, the dynamic programming version execution time is $O(k.L)$ where k is the number of queues and L the BBFRAME length. This implementation is thus less sensitive to the system load. The basic algorithm is of course the most efficient as far as execution time is concerned.

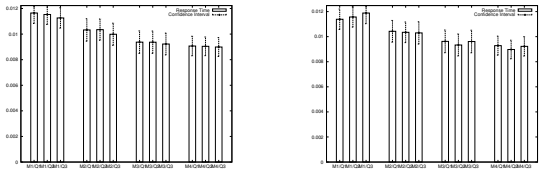

Fig. 2. Uniform scenario, knapsack (right) and exhaustive (left)

The performances of the knapsack and exhaustive implementations are very close, as depicted by figure 2 for the uniform scenario and figure 3 for the clear sky scenario. These figures show the mean waiting time for three streams sent to the four groups of receivers. We can notice that these implementations are fair in the uniform scenario (with no throughput bias).

This result was predictable as a consequence of the low probability of the cases that have not been considered by the knapsack implementation.

On the other hand, figure 4 shows the system performances for the basic implementation. We can notice that the overall performances are both lower and more

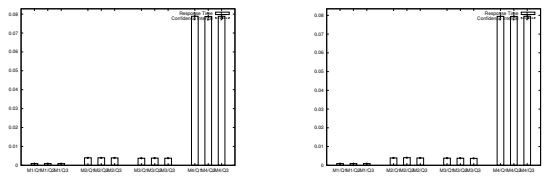

Fig. 3. Clear sky scenario, knapsack (right) and exhaustive (left)

unfair, although fairness remains within a given group of receivers with the same channel conditions).

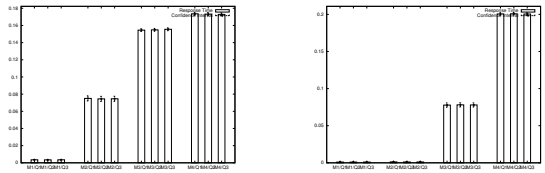

Fig. 4. Basic implementation, uniform (left) and clear sky scenarios

2) Changing channel conditions: We have simulated the consequences of a rain event. For this purpose, in a "clear sky scenario", we have changed the MODCOD used in the second most loaded queue. During a short period (12 minutes out of one hour of simulated time) it is replaced by a MODCOD with a lower spectral efficiency (leading to an overloaded system).

The knapsack implementation is very efficient in this situation. Of course the mean waiting time increases, and then some packets are lost. However, this implementation complies with utility functions : with U_{se}, the streams with the worst channel conditions suffer more from the performance degradation. On the other hand, using U_f, the streams with the highest throughput are the most impacted. Finally, reclassification can be really helpful, especially for streams with good channel conditions (*ie* with reclassification opportunities) : with U_f, the number of losses during the rain event has been decreased by 25% to 60%, depending on the stream.

The basic implementation seems unable to achieve any comparable results. The overall performance decreases and the utility functions are not enforced. Reclassification does not help in any way in this implementation.

VII. Conclusion and future works

Packet scheduling over GSE/DVB-S2 introduces lots of new challenges. These are consequences of the system properties : ACM, large frames, fragmentation, reclassification, ... Utility functions allow to encompass in a single expression multiple different constraints. They have been used to implement packet scheduling, but never in this specific context.

We have shown in this paper that with a slightly more general implementation (variable time slot, multi-dimensional gradient, ...) this technique could suit our problem. A drawback is that the search for an optimal scheduling could be very expensive. We have shown that basic algorithms are inefficient and that a knapsack-based algorithm could be both efficient and affordable.

Some interesting challenges remain, however. One of the most general (and maybe the most difficult) is how to translate high level QoS specifications (*eg* DiffServ PHBs) into utility functions. Another question is the tuning and the behaviour of the system when multiple different utility functions are used simultaneously. Finally, we plan to study in which conditions the system can be proved to remain stable.

References

[1] EBU-UER, "Digital video broadcasting (DVB) user guidelines for the second generation system for broadcasting, interactive servides, news gathering and other broadband satellite applications (DVB-S2)," European Telecommunications Standards Institute, Tech. Rep. v1.1.1, February 2005.

[2] ——, "Digital video broadcasting (DVB) ; second generation framing structure, channel coding and modulation systems for broadcasting, interactive servides, news gathering and other broadband satellite applications," European Telecommunications Standards Institute, Tech. Rep. v1.1.2, June 2006.

[3] DVB-GBS. W. Group, "IP/S.2 study, GSE protocol specification," European Telecommunications Standards Institute, Tech. Rep. Document rev 10, 2007.

[4] E. Chaput, A.-L. Beylot, and C. Baudoin, "Packet Scheduling Over DVB-S2 through GSE Enscapsulation," in *IEEE Global Communications Conference (GLOBECOM), La Nouvelle Orleans, 30/11/2008-04/12/2008.* http://www.ieee.org/: IEEE, 2008, pp. 1–5.

[5] S. Shenker, "Fundamental design issues for the future internet," *Selected Areas in Communications, IEEE Journal on*, vol. 13, no. 7, pp. 1176–1188, 1995. [Online]. Available: http://dx.doi.org/10.1109/49.414637

[6] F. Kelly, "Charging and rate control for elastic traffic," *European Transactions on Telecommunications*, vol. 8, pp. 33–37, 1997.

[7] A. L. Stolyar, "On the asymptotic optimality of the gradient scheduling algorithm for multiuser throughput allocation," *Oper. Res.*, vol. 53, no. 1, pp. 12–25, 2005.

[8] G. Song and Y. Li, "Utility-based resource allocation and scheduling in ofdm-based wireless broadband networks," *Communications Magazine, IEEE*, vol. 43, no. 12, pp. 127–134, Dec. 2005.

[9] ——, "Cross-layer optimization for ofdm wireless networks-part i: theoretical framework," *Wireless Communications, IEEE Transactions on*, vol. 4, no. 2, pp. 614–624, March 2005.

[10] ——, "Cross-layer optimization for ofdm wireless networks-part ii: algorithm development," *Wireless Communications, IEEE Transactions on*, vol. 4, no. 2, pp. 625–634, March 2005.

[11] P. Hosein, "Qos control for wcdma high speed packet data," *Mobile and Wireless Communications Network, 2002. 4th International Workshop on*, pp. 169–173, 2002.