Dynamic Resource Allocation based on a TCP-MAC Cross-Layer Approach for Interactive Satellite Networks

Paolo Chini, Giovanni Giambene
CNIT - University of Siena
giambene@unisi.it

Danilo Bartolini, Michele Luglio, Cesare Roseti
University of Rome "Tor Vergata"
{luglio, cesare.roseti}@uniroma2.it

Abstract— **DVB-RCS is an open standard for interactive broadband satellite services. According to the standard, interactive terminals communicate with the network control center through the return channel adopting MF-TDMA. In this scenario, classical bandwidth allocation schemes do not take into account TCP evolution, leading to sub-optimal performance when TCP-based traffic share the return link. A cross-layer approach, based on exchange of information between not-adjacent layers, can help to improve efficiency. This paper presents an innovative allocation algorithm based on a cross-layer interaction between TCP and MAC layers. Such an algorithm aims to synchronize the requests of resources with the TCP transmission window trend. The obtained results show that our scheme permits both to reduce the delay, to increase the utilization of air interface resource and to achieve a fair share of resources among competing flows.**

I. Introduction

In this paper we focus on an *Interactive Satellite Network* (ISN), based on the DVB-RCS standard [1],[2]. In particular, we consider a geostationary (GEO) bent-pipe satellite, *Return Channel Satellite Terminals* (RCSTs) and a *Network Control Center* (NCC), according to a classical star topology (see Fig. 1). Typically, an RCST is connected to a local network where different terminals are present. For the sake of simplicity, in the following study we refer to RCSTs with a single connected terminal; the extension of such work to the case of multiple terminals per RCST is feasible, but beyond the scope of this paper. In fact, we are interested to prove the importance of the cross-layer interaction between layer 2 and layer 4 to achieve both a higher TCP throughput (layer 4) and an efficient utilization of resources (layer 2).

The NCC is the core of the network: it provides control and monitoring functions and it manages network resources. RCSTs are fixed with *Return Channel via Satellite* (RCS) that allows transmitting data or control signaling, using an MF-TDMA access scheme. Resources are time slots on different available carrier frequencies. DVB-RCS return link resources are divided in super-frames that are characterized by suitable portions of time and frequency bands; each super-frame is divided in frames that are composed of several time-slots.

RCSTs send their resource requests to the NCC. The NCC looks at the available return link resources and sends a broadcast response to the RCSTs (forward channel) through the *Terminal Burst Time Plan* (TBTP), a message belonging to the *Service Information* (SI) tables, that allows the RCSTs to transmit data in allocated time slots, in some carrier frequencies and with a given power level. Definitively, the NCC assigns to each active RCST a set of bursts, each of them

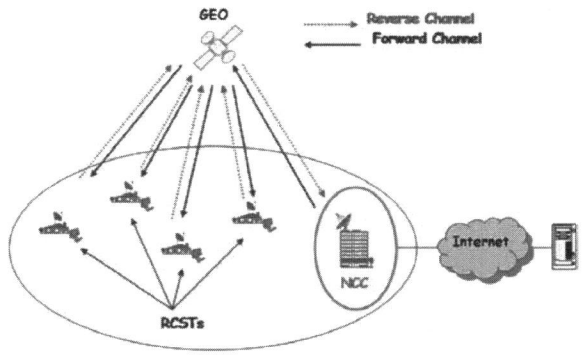

Fig. 1. System architecture.

is defined by a frequency, a bandwidth, a start time and a duration.

The DVB-RCS standard envisages 5 different types of resource allocation schemes; in particular, time slot allocations can be of fixed type (constant bandwidth and time duration) or dynamic one (variable bandwidth, time duration, transmission rate and code rate). In this paper, we refer to a *Bandwidth on Demand* (BoD) scheme that is able to follow the TCP behavior of each RCST and that also takes into account the congestion of the satellite network. Hence, among the DVB-RCS resource allocation types, the two major candidates are VBDC (*Volume Based Dynamic Capacity*) and AVBDC (*Absolute Volume Based Dynamic Capacity*) due to the fact that in both cases time-varying resource requests are made by the RCSTs. In the VBDC assignment, an RCST dynamically requests the total number of slots needed to idle its queue: requests are *cumulative*. Whereas, in the AVBDC case, an RCST dynamically requests the number of slots, but requests are *absolute*. Even if VBDC is the default mode in the DVB-RCS standard, an AVBDC-like scheme should be used in our

Work funded by the European Commission under the framework of the "SatNEx" NoE project www.satnex.org (contract No. 507052) – A&TCP research group.

scenario, since, according to our envisaged cross-layer approach, requests are not exactly related to the status of the buffer and are not cumulative, but rather they refer to the estimated next value of the TCP congestion window for each TCP flow.

II. Why Cross-Layer?

The Internet protocol architecture is based on a layering paradigm. Unfortunately, a strict modularity and layer independence may lead to non-optimal performance in IP-based next-generation wireless systems. In fact, many good assumptions for the wired networks are inadequate for wireless environments. For example, the assumption in the TCP protocol that all losses are due to congestion events is not particularly true in the wireless channel, where packets can be lost due to the corruption of the radio channel.

Furthermore, the growth of heterogeneous networks entails the need of adaptive actions. In this frame, a cross-layer approach would be more simple and flexible, with respect to the addition of a specific layer to implement the adaptation tasks [3]. Finally, in wireless systems, since both radio resources and power are strongly constrained, a network optimization is needed. Such optimization is not guaranteed by the current layered protocol stack, where, for instance, error correction schemes are implemented at both link and transport layer.

Of course, a cross-layer interaction needs an information exchange among not-adjacent layers. This signaling can be realized with different approaches [4]:

1. By encoding the cross-layer information in an additional header (i.e., *Interlayer Signaling Pipe* (ISP) [5]);
2. By using ICMP *(Internet Control Message Protocol)* [6],[7] to punch holes in the protocol stack. Such a method has been already implemented on Linux operating systems by suitable APIs (*Application Program Interfaces*);
3. By using "third parties" (i.e., distributed WCI -*Wireless Channel Information*- servers) to gather, abstract and manage information coming from different layers [8].

III. TCP Mechanisms

TCP is a transport protocol that provides a reliable, byte stream-based and connection-oriented service to the application layer. Three functions can be identified in the TCP protocol: flow control, congestion control and error recovery. The flow control scheme allows an efficient exchange of data not exceeding the capacity of the receiver buffer. In fact, using the *sliding window* mechanism, TCP continuously controls the amount of "in-flight" data [9]. On the other hand, the congestion control scheme probes the network status gradually, increasing the window of data that can be transmitted at a certain time, until the network drops a segment. To limit the transmission rate according to network conditions, TCP uses a variable, called "congestion window", which represents the maximum number of packets in flight not yet acknowledged. The minimum between the congestion window and the "advertised window", indicating the free space on the receiver buffer, determines the transmission window. TCP uses "cumulative acknowledgements" (ACKs) to notify to the sender the last segment received correctly and in order [9]. Then, the timeout expiration at the sender without receiving the related ACK (or the receipt of duplicated ACKs) indicates the loss of the corresponding segment, thus triggering one of the implemented error recovery schemes. In any case, TCP considers the segment loss as an evidence of network congestion, and then reduces its transmission window. Such assumption causes dramatic effects on the TCP performance over wireless links. TCP is unable to discriminate the different loss events (congestion or corruption), performing always a reduction of the transmission window, thus lowering its throughput.

In the following analysis, we considered the standard TCP NewReno version. The congestion control is based on four algorithms, named *Slow Start* (SS), *Congestion Avoidance* (CA), *Fast Retransmit* and *Fast Recovery* [10]. The SS phase runs either at the start of the connection or after timeout expirations, performing an exponential increase in the congestion window (*cwnd*) every time the TCP sender receives an ACK for the sent data. On the other hand, TCP triggers the CA mechanism when the congestion window gets through a threshold value, called *slow start threshold* (*ssthresh*), or after the *Fast Retransmit/Fast Recovery* phase, allowing a gentler probing of the available capacity by a linear increase of *cwnd*. However, in both mechanisms the TCP response time is on a *Round-Trip Time* (RTT) basis.

IV. TCP-Based Resource Allocation

In the case of a typical GEO satellite network with a large bandwidth-delay product, a generic TCP source should spend a very long time to fully exploit the available capacity. As a consequence, a static resource allocation scheme is not optimal in the presence of a dynamically varying TCP transmission window, leading to an inefficient and unfair sharing of the total capacity [11],[12].

For what concerns the MF-TDMA air interface of the DVB-RCS standard, we propose a BoD scheme operated by the NCC in order to allocate resources on the basis of both an estimation of the *cwnd* behavior for the next RTT for each RCST and the ISN congestion.

V. Cross-Layer Design

In this paper, we present a novel approach to assign/remove resources as a function of the TCP status of each active connection (i.e., RCST). Then, we move from a terminal-based to connection-based allocation scheme. Our idea is to synchronize the TCP transmission window trend to the allocated resources, and vice versa. In fact, we want to provide TCP information (i.e., CA or SS phase) as input to allocation algorithm in order to assign/remove resources on the basis of the transfer status at the transport level. As a consequence, when many competing connections share the channel, the scheduler unit at the NCC can perform more fair

decisions, guaranteeing at the same time an optimal channel utilization [13]. To propagate the cross-layer signaling between layer 2 and layer 4, an ICMP message (as mentioned in Section II) is generated each time TCP parameters change beyond a threshold in order to communicate a new estimate of the TCP future state (from layer 4 to layer 2), or when the scheduler unit (at the NCC) notifies congestion signaling to RCSTs at layer 4, as described in the next sub-section. Since ICMP messages are carried through selected "holes", the cross-layer communications do not require a layer-by-layer processing. Then, such an approach appears flexible and efficient.

A. RCST Side

In each RCST, we need a double cross-layer interaction [5],[6]: from layer 4 to layer 2, to reserve resources in advance for the uplink; from layer 2 to layer 4, to inform the upper layers about the actual congestion conditions of the network (feedback message from the NCC).

In particular, the TCP layer communicates to the MAC layer the current values of both *cwnd* and *ssthresh*. After an appropriate elaboration, the MAC, considering the current phase of TCP (SS or CA), estimates the amount of data that will be transmitted in the next RTT (*RR_next_RTT*). Then, MAC inserts such information in two new field of its header.

After an RTT, MAC receives a TBTP message from the NCC. This message contains the amount of allocated resources (*RA_RCST*) and a *cwnd** value (that will be described in the next sub-section). MAC uses the *RA_RCST* value to establish how many packets (waiting for transmission in the MAC buffer) can be transmitted in the next RTT. Then, it passes the *cwnd** value to the TCP layer to limit (if needed) the growth of *cwnd* (*cwnd* ← *cwnd**). This value is lower than the actual value of *cwnd* if *RA_RCST* is lower than *RR_next_RTT*, in the case of unavailable radio resources (i.e., congestion).

B. NCC Side

This Section describes the admission rules and the scheduling policy that the NCC uses to manage the RCST resource requests based on a prediction of the *cwnd* behaviour, taking into account: (*i*) system available capacity; (*ii*) number and type of simultaneous connections.

The NCC elaborates the requests received from all active RCSTs on the basis of several priorities and through suitable resource allocation algorithms. In particular, the MAC layer of NCC receives the transmission requests from the RCSTs and processes the input information (*RR_next_RTT* and TCP phase). Through a fair sharing algorithm, the NCC assigns the resources to the terminals (*RA_RCST*). This algorithm is described in the next sub-section.

If the requested amount of resources is unavailable, the NCC allocates only a part of the requests and it will allocate the remaining part as soon as they will be available. Effectively, the NCC defines a *cwnd** value to stop further increases in *cwnd* that could create congestion in the ISN.

Every super-frame, the NCC assigns the resources according to a fair sharing approach and sends a broadcast TBTP message to the RCSTs with the allocated resources and the information about the position of the slots (time and frequency).

With our proposed dynamic resource allocation scheme, the NCC can assign the same resource to two different RCSTs into two subsequent super-frames.

C. Resource Allocation Algorithm

The allocation algorithm inputs are the *TCP phase flag* and the *next congestion window estimate* (*next_cwnd*) inserted in two new fields of the MAC header. The resource allocation algorithm can be divided into two phases depending on the fact that there are still free resources or not. In particular, when there are not assigned resources, the NCC filters any incoming segment in order to compare the *next_cwnd* value with the amount of resources already assigned (*a_res*) to the corresponding source (in terms of packets). Therefore, two cases are possible:

- *if next_cwnd* < *a_res*. The NCC considers that losses affect the performance, and then takes "*a_res* – *next_cwnd*" resources from the given source;
- *if next_cwnd* > *a_res*. NCC considers that TCP sender will use all the assigned resources in the next RTT, and its internal buffer will start to accommodate segments. Therefore, a resource request is issued.

In the last case, the requests will be inserted in either a *high-priority queue* or *low-priority queue* according to the TCP phase flag. The proposed algorithm aims to privilege connections in the SS phase with respect to those working in CA. The NCC first evaluates all the requests in the high priority queue. The *Maximum Legal Increment* (MLI) [14] is implemented to manage both queues. The NCC, estimating all the RCST requests (*RR_next_RTT*), assigns resources to them, respecting both their order of priority and guaranteeing a minimum number of allocated resources per request (i.e., one slot of the MF-TDMA super-frame). If the request of a given RCST cannot be fully satisfied, the NCC stops its window growth by setting a new TCP status variable (*cwnd**), until new resources are available. The *cwnd** value is notified to the RCST by the ACK packets.

When all the resources have been already assigned, every RTT the NCC takes resources from the RCSTs with more resources and assigns it to that one with fewer resources. The aim of this algorithm phase is to guarantee a fair sharing of the channel among several connections, starting at different instants.

D. Implementation Issues: the Simulator

To analyse the effectiveness of the improvements introduced by the proposed cross-layer approach, we implemented a simulator in the ns-2 environment (release 2.27) [15]. In particular, we considered the ns-2 extensions aiming to reproduce a traditional GEO satellite network. In the simulation, terminal nodes have been placed on the Earth's surface by assigning latitude and longitude values; for each

node a network interface stack has been defined. On the other hand, a "bent-pipe" GEO satellite has been configured to connect multiple users (uplink) and to support asymmetric links. Then, TCP agents have been attached to the nodes to generate the data traffic. Moreover, we modified the C++ code to simulate a centralized MF-TDMA scheme and the NCC node. We created two new specific classes: "*Cross-layer*" and "*BoD_algorithm*". The former (in the RCST) performs the functions for the exchange of information between TCP and MAC and the functions for the estimation of the requested resources for the next RTT. The latter (in the NCC) implements the algorithm to allocate/deallocate the resources to RCSTs with the evaluation of *cwnd**.

VI. Results

We performed a simulation campaign aiming at highlighting the impact of the proposed cross-layer-based BoD scheme in managing FTP transfers via the satellite return link. In particular, we have reproduced a DVB-RCS like communication scenario, where a group of terrestrial terminal share the return channel to transfer data files towards the NCC node. The main simulation parameter values are detailed below:

- Minimum RTT value: 508 ms;
- Return link bandwidth: 2 Mbit/s;
- TCP packet size: 1500 bytes;
- Packet Error Rate (PER): $[0, 5\times10^{-2}]$;
- Transport Protocol: TCP NewReno;
- Application Protocol: FTP;
- Maximum number of TCP sources: 32;
- File size: 5 Mbytes.

The analysis is organized in two parts: an evaluation of the cross-layer-based allocation process, and an evaluation of the end-to-end performance, as presented in the following sub-Sections.

A. Cross-layer Resource Allocation Scheme

Fig. 2 shows the main phases of the proposed allocation scheme. First, it is clearly shown that the amount of assigned resources follows the TCP congestion window trend: an exponential resource assignment is performed when the SS phase is active, while a linear growth is obtained when TCP switches in CA phase. In particular, four distinguished allocation steps can be recognized:

1. Two connections start at different instant times. NCC allocates resources to both connections on the basis of their expected congestion window growth (notified by cross-layer signaling);
2. Available resources start to be exhausted. The NCC privileges connection 2, running the SS phase, with respect to connection 1 in the CA one;
3. All the return link resources have been allocated. To perform a fair resource sharing, the NCC moves a resource from connection 1 to connection 2, every RTT;

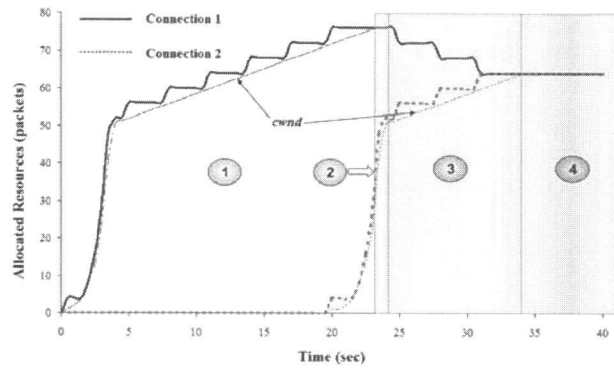

Fig. 2. Congestion window trend of two different data sources.

4. Finally, resources have been fairly split between the two active connections. The NCC stops the congestion window growth of both connections to avoid congestion in the internal buffer of the terminals.

Fig. 3 focuses on the resource redistribution process, when a single loss affects the transmission rate of one of the active connections. In fact, since connection 2 halves its congestion

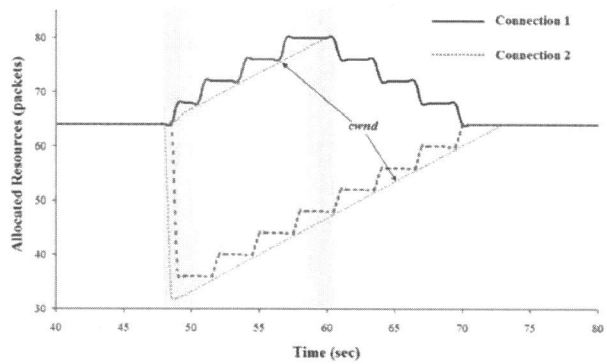

Fig. 3. Allocation process after a loss event.

window in response to a packet loss detection, the NCC temporally assigns unused resources to connection 1, thus allowing its window growth (i.e., multiplexing effect). Later, when all the resources have been assigned, the NCC operates to achieve a fair resource sharing between connections.

B. End-to-end Performance Evaluation

In order to probe the benefits of the proposed allocation scheme, we performed some preliminary end-to-end tests, considering a fixed allocation scheme (static repartition of the whole capacity among all the active connections) as a reference. Furthermore, we envisaged a simulation scenario, with 15 TCP connections starting after regular intervals (5 seconds), and transferring data files of 5 Mbytes. In particular, Fig. 4 evaluates the achieved average throughput for a large

range of PER values. The outcomes show an overall performance improvement (up to 56%), when the cross-layer allocation scheme is employed.

Finally, we also measured the average transfer time of each TCP connection. Fig. 5 plots the mean time that each

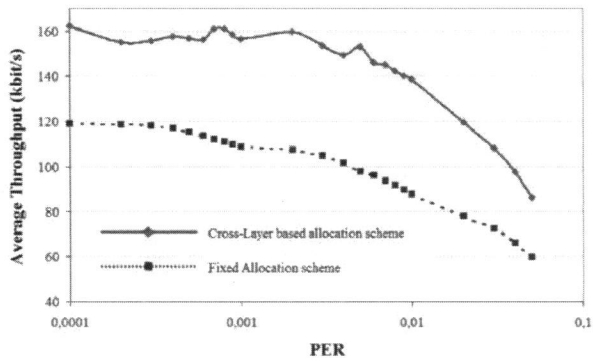

Fig. 4. Average throughput vs. PER.

connection spends to transfer a 5 Mbytes file when the radio channel is affected by a PER of 10^{-3}. The curves show that there are no privileged connections. The slight differences in term of mean file transfer time are due to the variable network

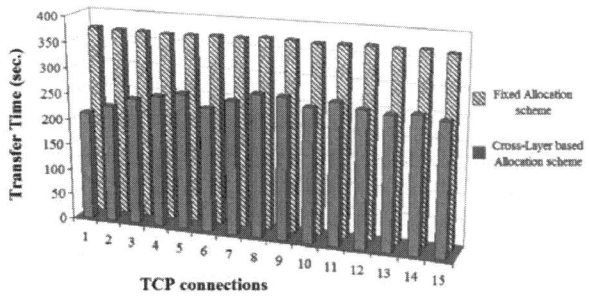

Fig. 5. Mean time needed to transfer 5 Mbytes files.

load.

The proposed allocation approach leads to two main advantages: (*i*) better overall performance with respect to a fixed allocation scheme, (*ii*) higher fairness among the competing connections.

VII. Conclusions

In this paper, we proposed a new approach to dynamically allocate resources, based on a cross-layer interaction between TCP and MAC layers operated by terminals. Referring to a DVB-RCS scenario, we considered that the NCC dynamically allocates resources to terminals on the basis of their expected behaviors of the TCP congestion window. Extensive simulation results have permitted to show that the envisaged cross-layer allocation scheme permits to improve the utilization of resources, achieving a higher throughput at the TCP level and reducing the occurrence of TCP time outs due to congestion in the ISN.

Acknowledgements

This work has been developed in the frame of the "SatNEx" NoE project (contract No. 507052) under the FP6 of the European Commission - joint activity 2430 (A&TCP).

References

[1] ETSI EN 301 790, "Digital Video Broadcasting (DVB); Interaction Channel for Satellite Distribution System," V1.3.1, 2003.

[2] ETSI TR 101 790, "Digital Video Broadcasting (DVB); Interaction Channel for Satellite Distribution System, Guidelines for the use of EN 301 790,", V1.2.1, 2003.

[3] Z.H. Haas, "Design methodologies for adaptive and multimedia networks," Guest Editorial, IEEE Communications Magazine, Vol.39, No. 11, pp. 106-107, November 2001.

[4] Qi Wang, M.A. Abu-Rgheff, "Cross-Layer Signalling for Next-Generation Wireless Systems," Wireless Communications and Networking, Vol. 2, pp. 1084-1089, March 2003.

[5] G. Wu, Y. Bai, J. Lai, A. Ogielski, "Interactions between TCP and RLP in wireless Internet," Proc. IEEE GLOBECOM'99, December 1999.

[6] J. Postel, "Internet Control Message Protocol," RFC 792, September 1981.

[7] A. Conta, S. Deering, "Internet control message protocol (ICMPv6) for the Internet Protocol version 6 (IPv6)," RFC 1885, December 1995.

[8] K. Chen, S.H. Shan, K. Nahrstedt, "Cross-Layer design for data accessibility in mobile ad-hoc networks," Wireless Personal Communications, Vol. 21, No. 1, pp. 49-76, April 2002.

[9] W. Stevens, "TCP/IP illustrated," vol. 1, Ed. Addison Wesley, 1994.

[10] W. Stevens, "TCP Slow Start, Congestion Avoidance, Fast retransmit and Fast Recovery Algorithms" Internet RFC 2001, 1997.

[11] C. Patridge, T.J. Shepard, "TCP Performance over Satellite Links," IEEE Network, vol. 11, No. 5, pp. 44-49, 1997.

[12] M. Luglio, C. Roseti, M. Gerla, "The Impact of Efficient Flow Control and OS Features on TCP Performance over Satellite Links," ASSI Satellite Communication Letter, vol. 3, No. 1, pp.1-9, 2004.

[13] E.Guainella, A.Pietrabissa, "TCP-Friendly Bandwidth-on-Demand for Satellite Networks," ASMS Conference, July 2003.

[14] G. Acar, and C. Rosenberg, "Algorithms to compute for Bandwidth on Demand Requests in a Satellite Access Unit", Proc. of 5th Ka-band Utilization Conference, 1999, pp. 353-360.

[15] NS-2 Network Simulator (Ver. 2) LBL, URL: http://www.isi.edu/nsnam/ns/ns-build.html