Interaction transport protocols / MAC fiabilisation for satellite mobiles services

Bastien Tauran[1]

[1]University of Toulouse, TéSA, ISAE/DISC, Toulouse, France

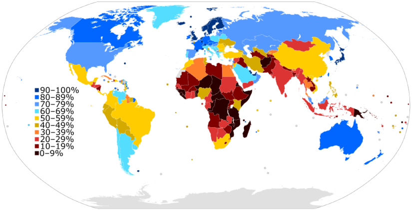
December 6th, 2018

Directeur:	Emmanuel Lochin
Co-directeur:	Jérôme Lacan
Rapporteurs:	Pascal Anelli
	Eugen Dedu
Examinateurs:	Caroline Bès
	Thierry Gayraud

Outline

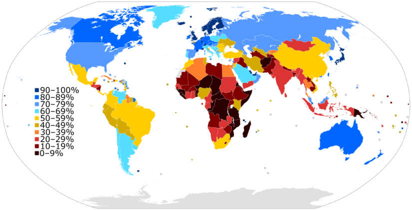
Satellite context

Why a satellite context



Internet users in 2015 as a percentage of a country's population

Why a satellite context



Internet users in 2015 as a percentage of a country's population

Low Earth Orbit satellite constellation

- LEO satellite constellations $\Rightarrow$ connect isolated areas or moving devices

Low Earth Orbit satellite constellation

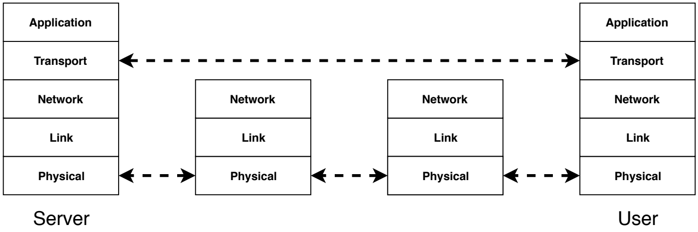
- LEO satellite constellations $\Rightarrow$ connect isolated areas or moving devices
- Advantages: LEO constellations allow the use of standard TCP/IP stack
  - Low delays in a LEO constellation

Low Earth Orbit satellite constellation

- LEO satellite constellations $\Rightarrow$ connect isolated areas or moving devices
- Advantages: LEO constellations allow the use of standard TCP/IP stack
  - Low delays in a LEO constellation
- Drawback: losses due to interferences, mobility, handover
  - TCP badly reacts to random losses
  - Need to use reliability schemes

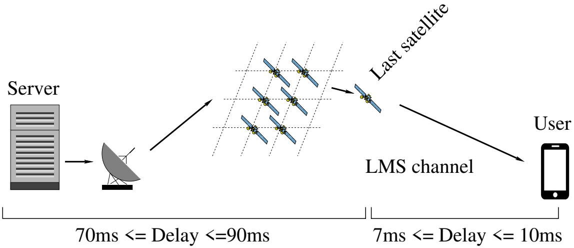
Low Earth Orbit satellite constellation

- LEO satellite constellations ⇒ connect isolated areas or moving devices
- Advantages: LEO constellations allow the use of standard TCP/IP stack
  - Low delays in a LEO constellation
- Drawback: losses due to interferences, mobility, handover
  - TCP badly reacts to random losses
  - Need to use reliability schemes
- Problem: interaction between the transport layer and the low layer reliability schemes



Background and state of the art

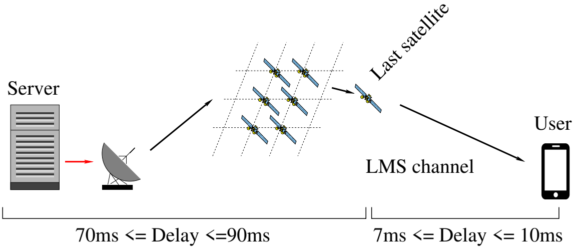
Scenario used



Satellite environment

- Changing transmission delay, between 70 ms and 100 ms
- Transmission errors only on the LMS channels
- No errors on the other links and the return path
- LMS channel
  - Low throughput
  - Fast variations

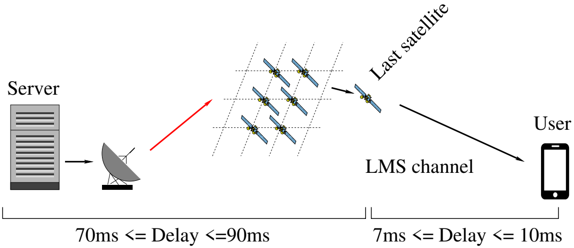
Scenario used



Satellite environment

- Changing transmission delay, between 70 ms and 100 ms
- Transmission errors only on the LMS channels
- No errors on the other links and the return path
- LMS channel
  - Low throughput
  - Fast variations

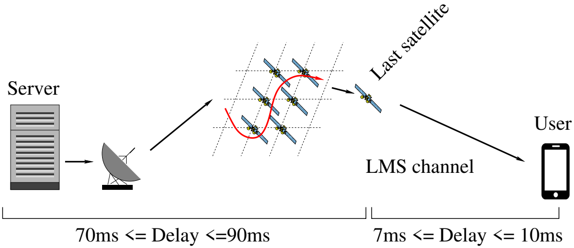
Scenario used



Satellite environment

- Changing transmission delay, between 70 ms and 100 ms
- Transmission errors only on the LMS channels
- No errors on the other links and the return path
- LMS channel
  - Low throughput
  - Fast variations

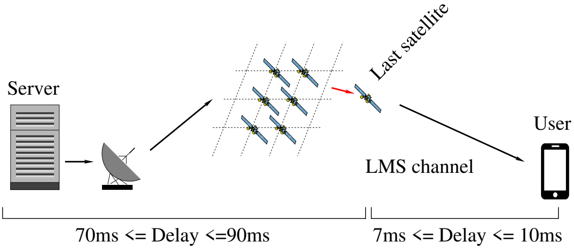
Scenario used



Satellite environment

- Changing transmission delay, between 70 ms and 100 ms
- Transmission errors only on the LMS channels
- No errors on the other links and the return path
- LMS channel
  - Low throughput
  - Fast variations

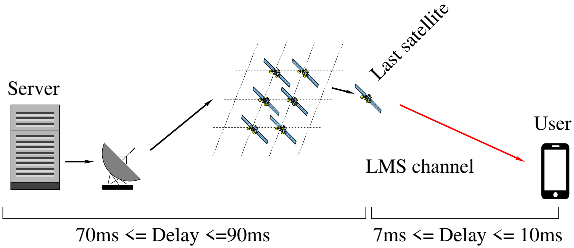
Scenario used



Satellite environment

- Changing transmission delay, between 70 ms and 100 ms
- Transmission errors only on the LMS channels
- No errors on the other links and the return path
- LMS channel
  - Low throughput
  - Fast variations

Scenario used



Satellite environment

- Changing transmission delay, between 70 ms and 100 ms
- Transmission errors only on the LMS channels
- No errors on the other links and the return path
- LMS channel
  - Low throughput
  - Fast variations

How to recover missing data

High error rate $\Rightarrow$ Reliability schemes

- Automatic Repeat reQuest (ARQ)
- Error Correction Code

How to recover missing data

High error rate $\Rightarrow$ Reliability schemes

- Automatic Repeat reQuest (ARQ)
- Error Correction Code
- Hybrid Automatic Repeat reQuest (HARQ)
  - ARQ + Error Correction Code
  - Best solution for satellite constellations
  - Drawback: delay and jitter increased

Services considered

Two services:

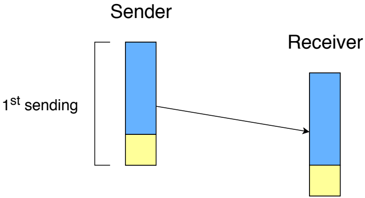
- Best-effort
  - TCP
  - Study of transmission performance
- VoIP
  - UDP
  - Low rates compatible with satellite transmissions

TCP performance analysis and tuning

Reliability scheme used

Use of Hybrid Automatic Repeat reQuest (HARQ)

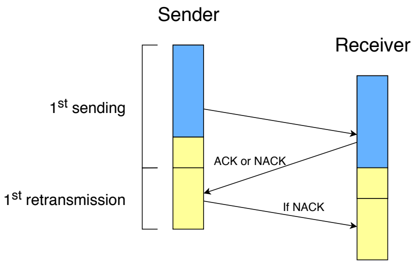
Description of HARQ



Reliability scheme used

Use of Hybrid Automatic Repeat reQuest (HARQ)

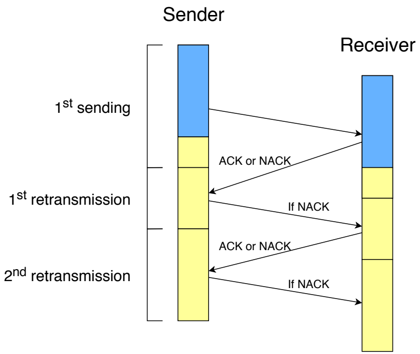
Description of HARQ



Reliability scheme used

Use of Hybrid Automatic Repeat reQuest (HARQ)

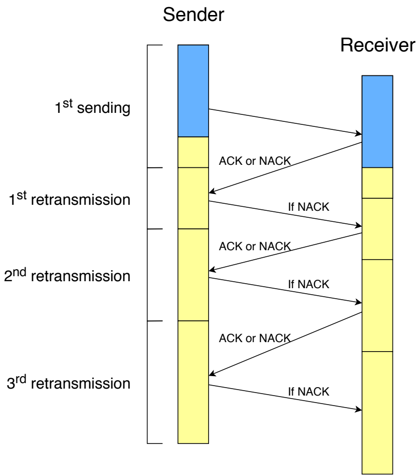
Description of HARQ



Reliability scheme used

Use of Hybrid Automatic Repeat reQuest (HARQ)

Description of HARQ



Transmission Control Protocol (TCP)

Variants used:

- TCP NewReno
- CUBIC
- TCP Hybla

Transmission Control Protocol (TCP)

Variants used:

- TCP NewReno
- CUBIC
- TCP Hybla

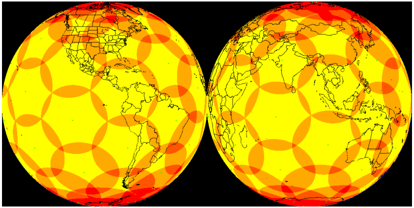
Useful TCP metrics

- TCP congestion window (*cwnd*)
- TCP Duplicate Acknowledgment (DUPACK)
- TCP Retransmission TimeOut (RTO)
- Spurious retransmissions

Implementation in *ns*-2

Simulations made with *ns*-2

- HARQ implemented in *ns*-2 ($\approx$ 1500 lines of code)
  - Uses real loss traces
- Uses delay traces simulating satellite movements and route changes
  - Satellite topology obtained from SaVi
  - Delay traces extracted using *ns*-2
  - Traces played in the simulations



- Other implementations: $\approx$ 750 lines of code

TCP performance

Lossless network $\Rightarrow$ expected goodput of 40 Mb/s

- Bandwidth of the LMS link: 50 Mb/s
- End-to-end delay: 70 ms to 100 ms

TCP performance

Lossless network $\Rightarrow$ expected goodput of 40 Mb/s

- Bandwidth of the LMS link: 50 Mb/s
- End-to-end delay: 70 ms to 100 ms

Measured goodput: 500 kb/s max

TCP performance

Lossless network ⇒ expected goodput of 40 Mb/s

- Bandwidth of the LMS link: 50 Mb/s
- End-to-end delay: 70 ms to 100 ms

Measured goodput: 500 kb/s max

- Explained by HARQ:
  - longer transmission delay
  - out-of-order packets
- ⇒ High number of spurious retransmissions

TCP performance

Lossless network $\Rightarrow$ expected goodput of 40 Mb/s

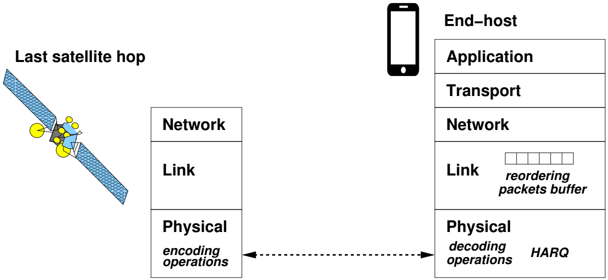
- Bandwidth of the LMS link: 50 Mb/s
- End-to-end delay: 70 ms to 100 ms

Measured goodput: 500 kb/s max

- Explained by HARQ:
  - longer transmission delay
  - out-of-order packets
- $\Rightarrow$ High number of spurious retransmissions

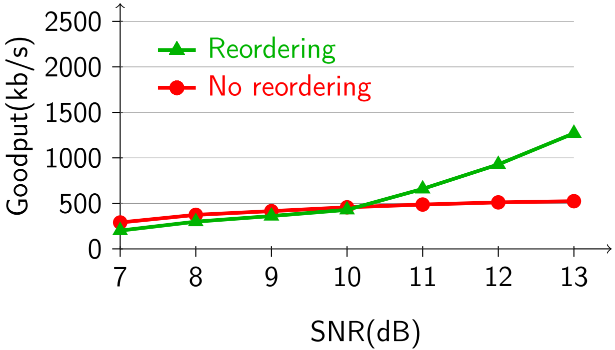
There is a need to mitigate out-of-order packets

Solution proposed:
adding a reordering mechanism between HARQ and the transport layer



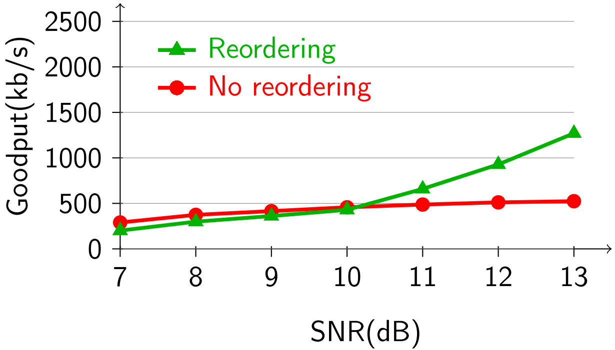
Description of HARQ with reordering mechanism

Performance gain



Impact of reordering mechanism on end-to-end goodput (CUBIC)

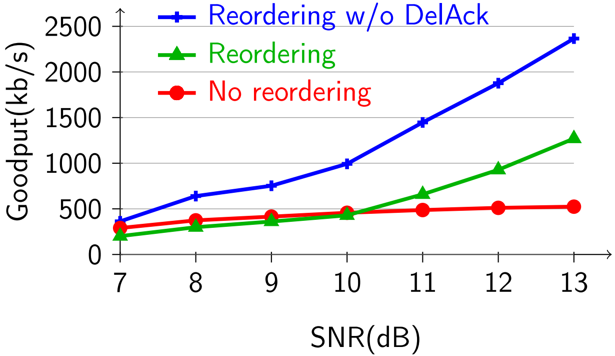
Performance gain



Impact of reordering mechanism on end-to-end goodput (CUBIC)

- Huge diminution of the number of DUPACK
- Increase of number of RTO

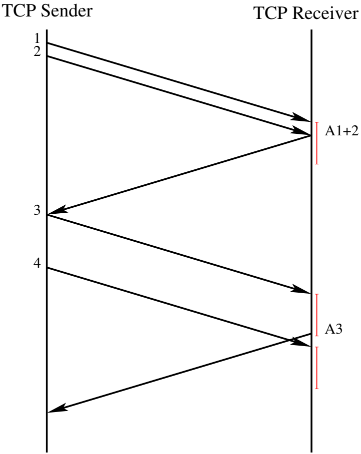
Performance gain



Impact of reordering mechanism on end-to-end goodput (CUBIC)

- Huge diminution of the number of DUPACK
- Increase of number of RTO

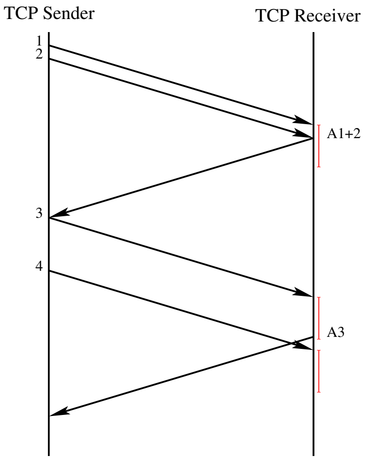
Delayed Acknowledgment - definition



Delayed Acknowledgment (DelAck):

- DelAck allows TCP to send only one Ack if 2 in-order packets are received within a fixed time window

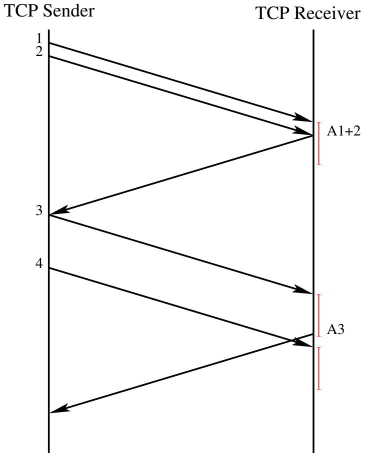
Delayed Acknowledgment - definition



Delayed Acknowledgment (DelAck):

- DelAck allows TCP to send only one Ack if 2 in-order packets are received within a fixed time window
- If an out-of-order packet is received, an single Ack is instantly sent

Delayed Acknowledgment - definition



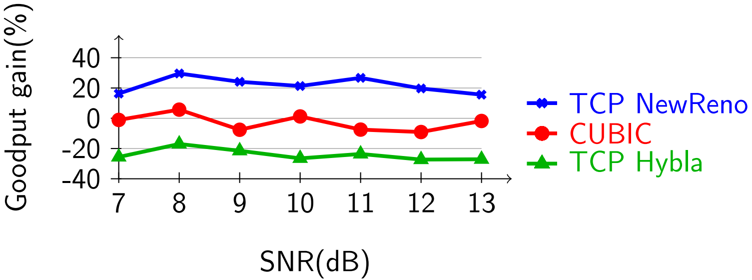
Delayed Acknowledgment (DelAck):

- DelAck allows TCP to send only one Ack if 2 in-order packets are received within a fixed time window
- If an out-of-order packet is received, an single Ack is instantly sent

Main advantage:

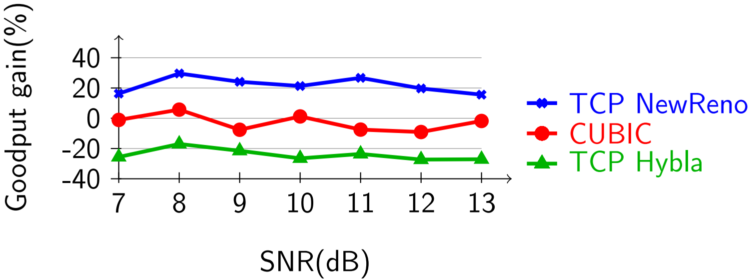
- Decrease number of Acks

Impact of DelAck - No reordering



Impact of DelAck activation on end to end goodput

Impact of DelAck - No reordering



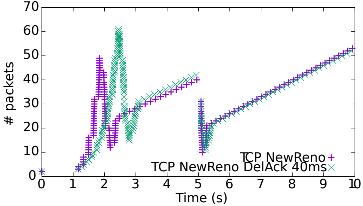
Impact of DelAck activation on end to end goodput

Impact of DelAck:

- Improvement with TCP NewReno
- Same performance with CUBIC
- Performance decrease with TCP Hybla

Study of the cause of this impact - No reordering

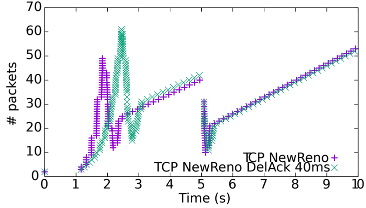
Enabling DelAck slows down the congestion window growth



Evolution of TCP congestion window with and without DelAck

Study of the cause of this impact - No reordering

Enabling DelAck slows down the congestion window growth



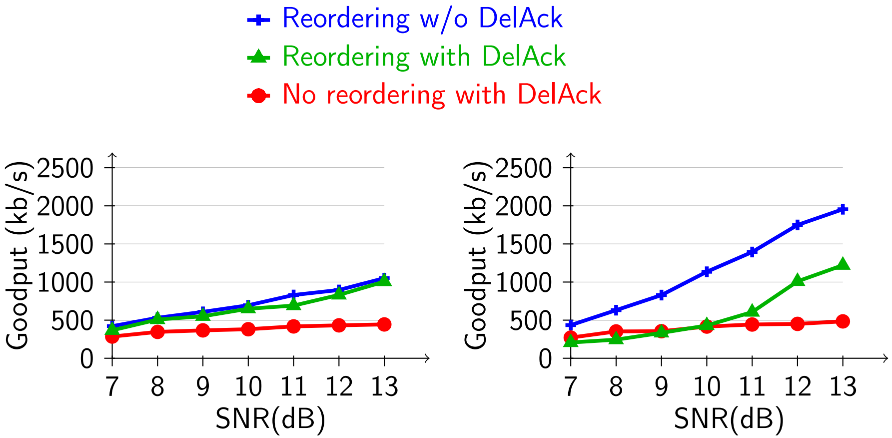
Evolution of TCP congestion window with and without DelAck

- TCP versions more aggressive ⇒ more retransmissions

SNR (dB)	Number of retransmissions		
	NR	CUBIC	Hybla
7	1068	1333	6969
8	932	1100	16914
9	925	1105	23073
10	789	949	16782
11	748	923	22399
12	742	780	23852
13	706	789	20688

Number of packets retransmitted, when DelAck is activated

Impact of DelAck - Reordering



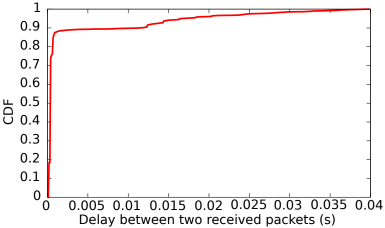
TCP NewReno

CUBIC

Whatever the variant $\Rightarrow$ better results without DelAck

Study of the cause of this impact - Reordering

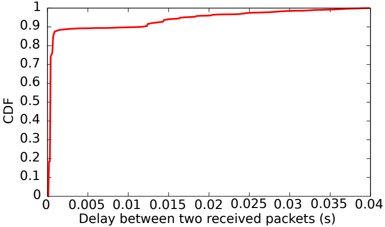
- HARQ + Reordering + DelAck ⇒ high transmission delay



Delay between the reception of two in-order packets forming a DelAck

Study of the cause of this impact - Reordering

- HARQ + Reordering + DelAck $\Rightarrow$ high transmission delay



Delay between the reception of two in-order packets forming a DelAck

- Transmission delay is smoothed $\Rightarrow$ RTO timer value decreases

$$RTO = SRTT + 4 * RTTVAR$$

TCP performance evaluation - conclusion

- TCP suffers from long and varying delays caused by HARQ

TCP performance evaluation - conclusion

- TCP suffers from long and varying delays caused by HARQ
- A reordering mechanism mitigates the impact of out-of-order packets

TCP performance evaluation - conclusion

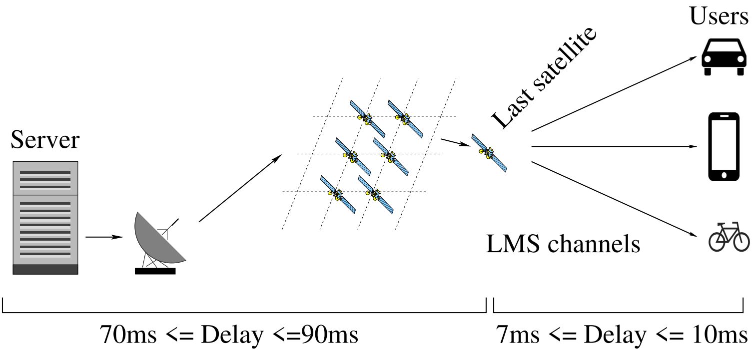
- TCP suffers from long and varying delays caused by HARQ
- A reordering mechanism mitigates the impact of out-of-order packets
- DelAck has a significant impact on TCP performance
  - Negative for the most aggressive TCP variants, without reordering
  - Always negative with the reordering mechanism

Solutions to deploy Internet services over LEO constellations

Improving the LMS channel capacity

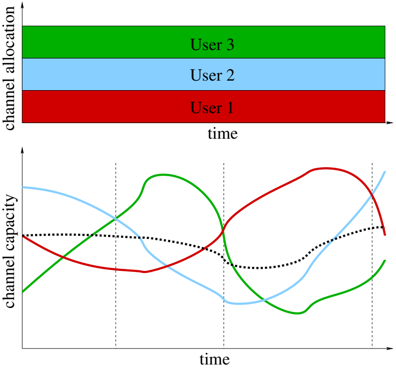
When several users share the LMS link:

- Same last satellite
- LMS channels evolution independant



Satellite environment

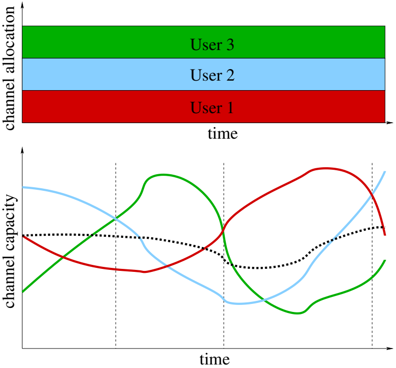
- The LMS channels are independent



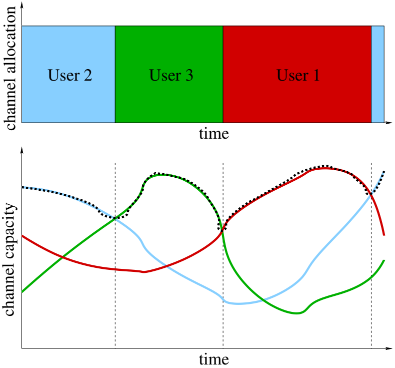
Before scheduling

Improving the LMS channel capacity

- The LMS channels are independent
- Idea: send only to the user with the best capacity



Before scheduling



After scheduling

Scheduling policies

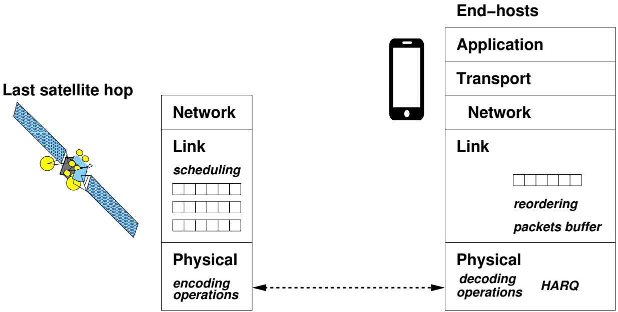
Need to find a good scheduler:

- Favoring users with a good channel capacity
- Ensuring fairness between the users

Algorithm	Metric	Traffic	Complexity
PF	Throughput	Best-effort	Very low
M-LWDF	Throughput and time in buffer	Real-time	Low
EXP-PF	Throughput and time in buffer	Real-time and BE	Low
UBMT	Throughput and time in buffer	All	Medium
BBS and BPS	Throughput or time in buffer	Depends on metric	High

Comparison of different schedulers

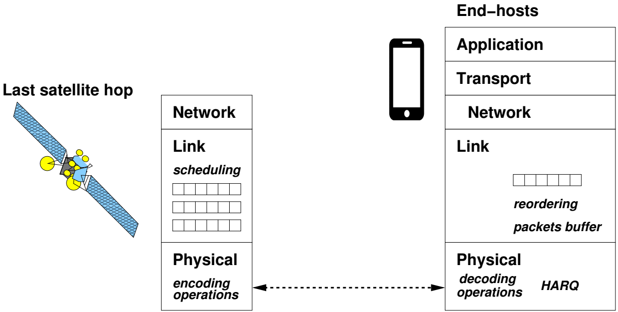
Scheduling policies



Proportional Fairness (PF)

- PF sends packets to the user with the best channel capacity
- $i^* = argmax(\frac{q_i(t)}{\overline{q_i}(t)})$

Scheduling policies



Proportional Fairness (PF)

- PF sends packets to the user with the best channel capacity
- $i^* = argmax(\frac{q_i(t)}{\overline{q_i}(t)})$

Comparing with:

- Round Robin (RR)
- DropTail
  - with small buffer capacity (DT_S)
  - with high buffer capacity (DT_H)

Simulation parameters

Case of Voice over IP (VoIP) flows

- From 10 to 200 parallel flows
- Each flow follows the G711 norm with a rate of 64 kb/s

Important metrics ensuring a good Quality of Experience (QoE)

- latency
- jitter
- loss rate

Simulation parameters

Case of Voice over IP (VoIP) flows

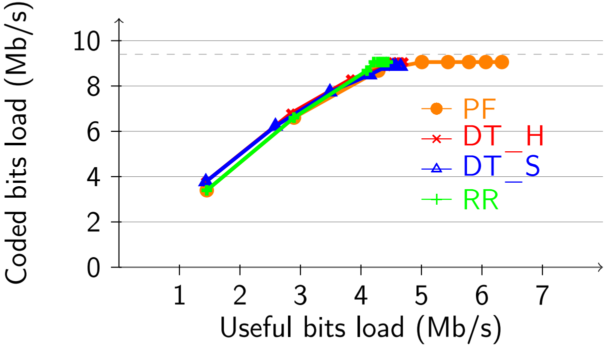
- From 10 to 200 parallel flows
- Each flow follows the G711 norm with a rate of 64 kb/s

Important metrics ensuring a good Quality of Experience (QoE)

- latency
- jitter
- loss rate

In our scenario, saturation reached between 75 and 100 users

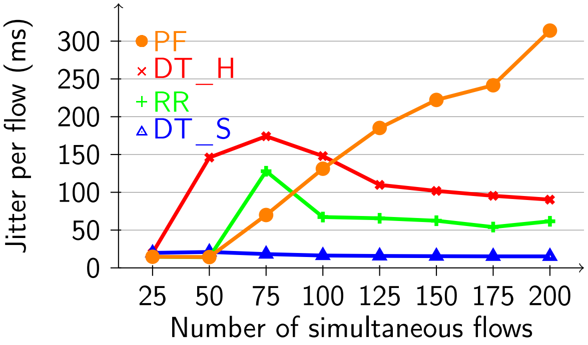
Performance per flow - Spectral efficiency



LMS channel spectral efficiency

PF improves the spectral efficiency of the LMS channel
$\Rightarrow$ more useful bits are transmitted for a same value of coded bits

Performance per flow - Jitter



Jitter obtained with different policies

However, PF causes a high jitter $\Rightarrow$ decreases the QoE of the users

Improving the scheduler efficiency

Drawbacks of the queues:

- packets are dropped due to buffer overflow
- packets can be expired when arriving at the receiver

Improving the scheduler efficiency

Drawbacks of the queues:

- packets are dropped due to buffer overflow
- packets can be expired when arriving at the receiver

Improvement: new queue management policy, working conjointly with PF: **Controlled Delay Scheduler (CoDeS)**

- packets are now dropped to timeout and not buffer overflow

Improving the scheduler efficiency

Drawbacks of the queues:

- packets are dropped due to buffer overflow
- packets can be expired when arriving at the receiver

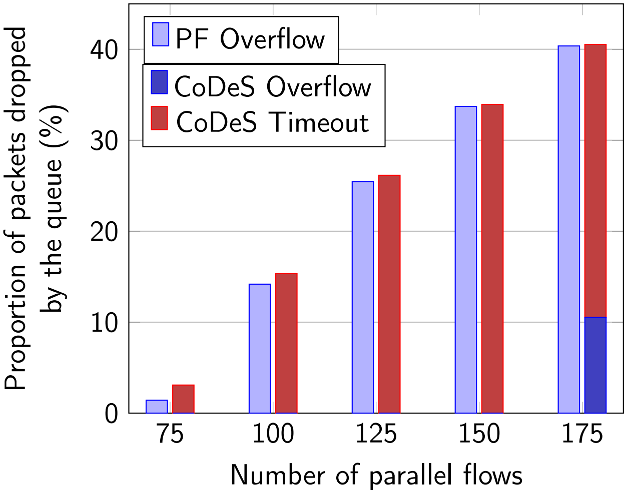
Improvement: new queue management policy, working conjointly with PF: **Controlled Delay Scheduler (CoDeS)**

- packets are now dropped to timeout and not buffer overflow

Finding optimal timeout value:

- Optimal value between 50 ms and 100 ms $\Rightarrow$ best compromise, ensuring the best QoE for the users

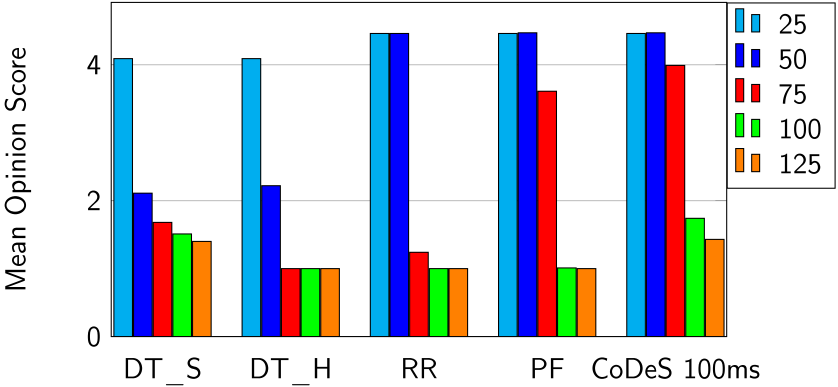
Drop cause with CoDeS



Cause of packets drop with PF and CoDeS

Performance per flow - MOS

$$MOS = f(codec, latency, jitter, losses)$$



Mean Opinion Score of scheduling policies

Scheduling TCP flows

Case of TCP flows and best-effort traffic, the objectives are now:

- a good delivery of every packets without any error;
- to maximize the transmission goodput for each user.

Scheduling TCP flows

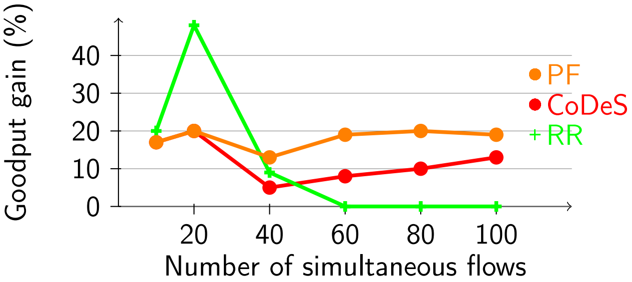
Case of TCP flows and best-effort traffic, the objectives are now:

- a good delivery of every packets without any error;
- to maximize the transmission goodput for each user.

First remarks:

- PF is totally suited of best-effort traffic
- Performance of CoDeS ?
  - What is its interest now ?

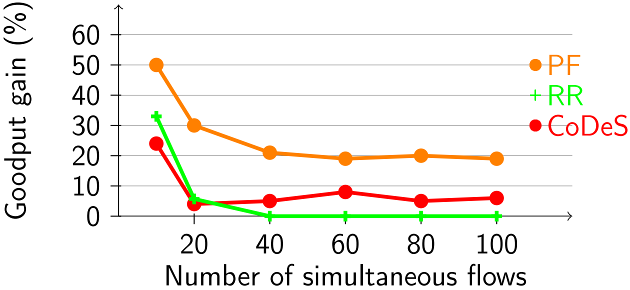
TCP flows - Without reordering



Goodput gain per user for different scheduling policies compared to DT_H (CUBIC), without reordering

- PF is the best policy
- CoDeS not competitive
- RR: good performance with low number of users

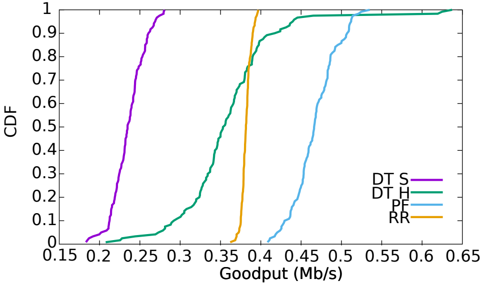
TCP flows - With reordering



Goodput gain per user for different scheduling policies compared to DT_H (CUBIC), with reordering

- PF still the best
- CoDeS and RR not competitive

Fairness between TCP users



Fairness between users with reordering

- PF has the best performance
- RR has the lowest variance
- DropTail queues are not competitive

Schedulers - summary

Criteria	DT_S	DT_H	RR	PF	CoDeS
VoIP (QoE)	✗✗	✗✗	✗✗	✓	✓✓
TCP (Goodput)	✗✗	✗	✗	✓✓	✓
Fairness between users	✗	✗✗	✓✓	✓	✓
Overall efficiency	✗✗	✗	✗	✓✓	✓✓

Performance of schedulers for different criteria

Conclusion and perspectives

Conclusion

- Reliability schemes such as HARQ have a high impact on TCP performance
- DelAck can also be counter-productive in this context
- We proposed solutions to deploy Internet services over LEO
  - VoIP service
  - Best-effort service

Perspectives

- Evaluation of other transport protocols
  - BBR
  - QUIC
- Study scheduling policies with other kinds of traffic
- Study TCP and scheduling policies performance with other topologies

Thank you for your attention

Questions ?