

Algorithms for Inter-domain Dynamic Bandwidth Allocation

Giorgio Calarco, Carla Raffaelli

D.E.I.S. – University of Bologna - ITALY

{ gcalarco, craffaelli } @deis.unibo.it

IPS 2003, Salzburg

February 2003

Targets

- Dynamic bandwidth allocation to guarantee the quality of service for the aggregated flows of traffic
- System parameters optimization
- Congestion management to avoid quality degradation
- Performance evaluation with particular reference to system efficiency and delay

The two-tier Model

The two-tier architecture splits resources' management in:

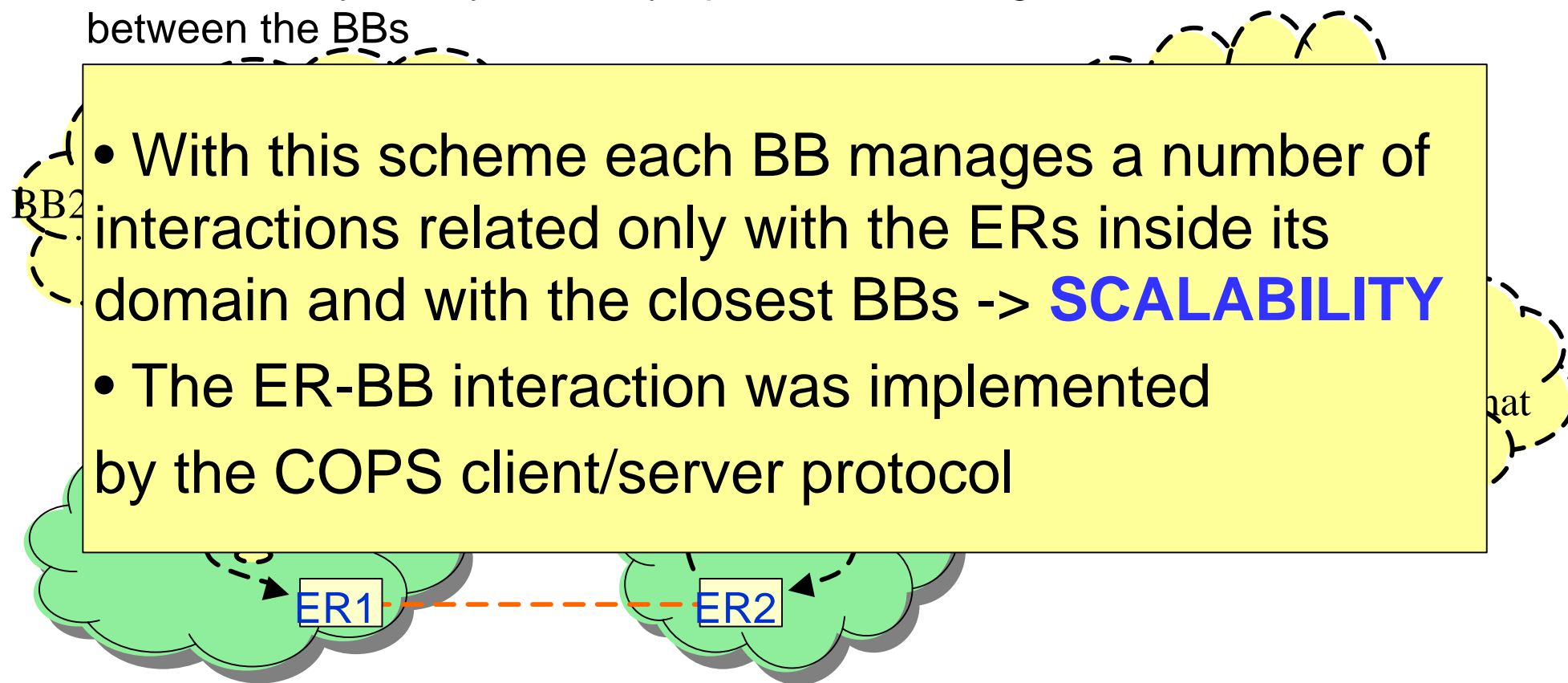
- Intra-domain allocation of resources
 - each domain can have its own independent policies
 - based on the single application flow
- **Inter-domain** allocation of resources
 - agreements between adjacent domains are needed
 - based on the aggregated flow of traffic belonging to a class of service

Each domain hosts one resource manager called **Bandwidth Broker**

Interaction Model for Bandwidth Updates

- Inter-domain QoS is guaranteed by bilateral agreements between the BBs of adjacent domains
- When the allocated resources (for a single QoS class) are no more sufficient, they are dynamically updated: a new agreement is needed between the BBs

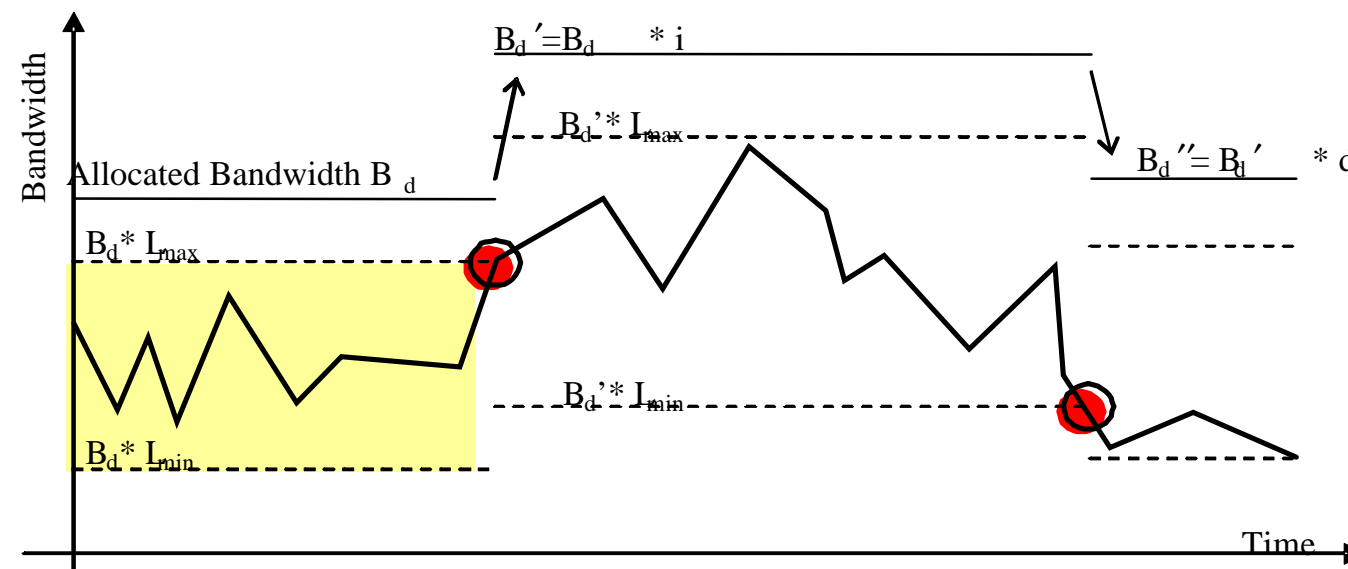
- With this scheme each BB manages a number of interactions related only with the ERs inside its domain and with the closest BBs -> **SCALABILITY**
- The ER-BB interaction was implemented by the COPS client/server protocol



Threshold-based Algorithm for bandwidth updates

- Variations of bandwidth are decided by the mean of a threshold-based mechanism
- Bandwidth updates are needed when the **used bandwidth**
 - $B_0 > t_u = B_d L_{max}$ (upper threshold) -> increase
 - $B_0 < t_l = B_d L_{min}$ (lower threshold) -> decrease
- If there's no congestion, the new **allocated bandwidth** b_d' is calculated using the **i e d** factors, so that:
 - $B_d' = i B_d$ (increase)
 - $B_d' = d B_d$ (decrease)

The update requests are always satisfied in a time limited by T_{BB}

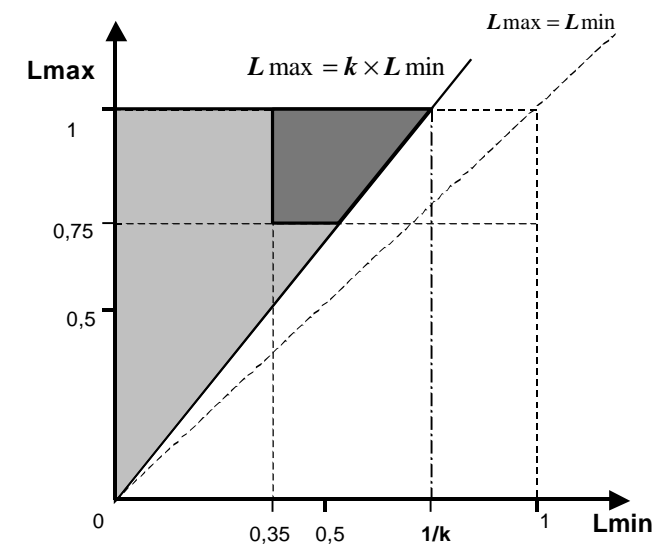


Constraints on system parameters

Parameters configuration is crucial to achieve bandwidth stability, scalability and efficiency

Oscillations of B_d can arise. **Stability** is guaranteed (with slow traffic variations) if

$$L_{max} > k L_{min} , \quad k = \max(i, 1/d)$$



Scalability: number N of interactions with the BB

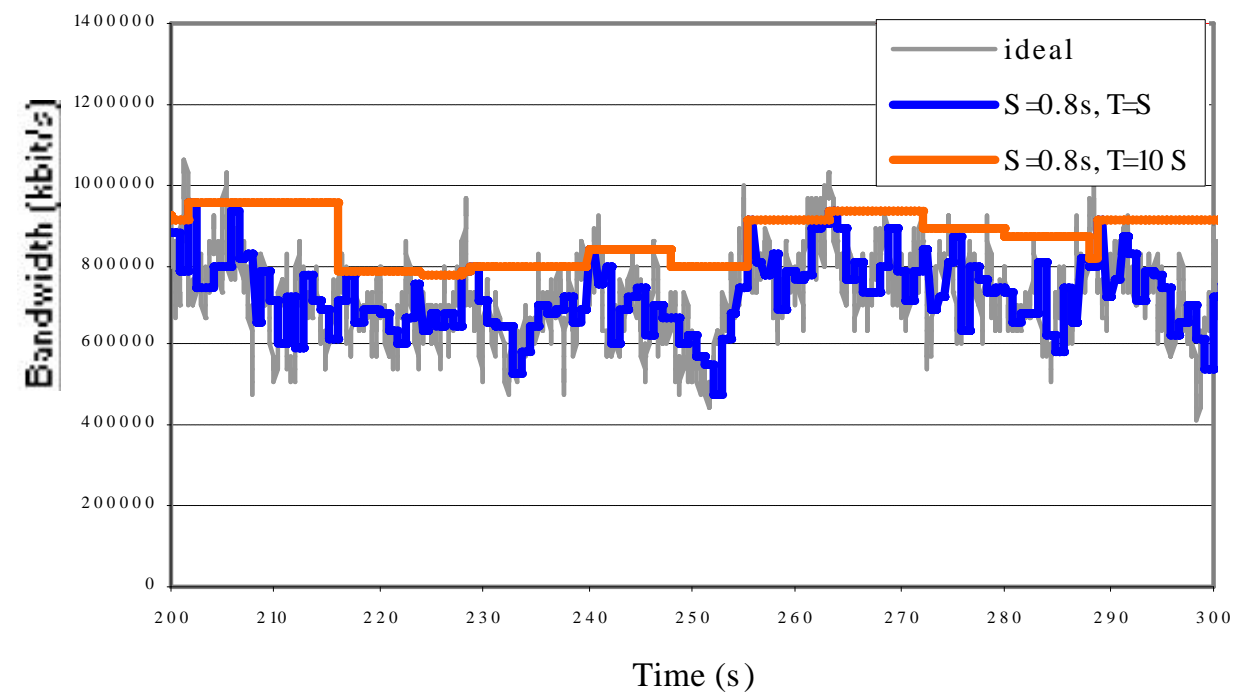
Efficiency

$$r_b = \frac{1}{T} \int_T \frac{b_0(t)}{b_d(t)} dt \cong \frac{b_{0m}}{b_{dm}}$$

Measurement of bandwidth

- Exact and immediate knowledge of the current bandwidth usage is an abstraction !
- In practice, the threshold algorithm for updates can be based on measured values of bandwidth
- **Time window** measurement system with parameters T (window length) and S (sampling time) has been used

Time Window Measurement of bandwidth (2)



- The measured bandwidth value is available with delay
- Increasing the time window, memory increases

Extension to the basic algorithm: Congestion Management

- Congestion arises when $b_0(t) > b_d(t)$
- Packets are queued, delay grows up and real-time services quality becomes poor
- A congestion management algorithm, which monitors the queue state, was inserted to avoid the *short term congestions*

Congestion Management Algorithm

- The main constraint is T_s , the max acceptable time to empty the queue
- The queue threshold (in bytes): $S_q(t) = T_s b_d(t)/8$
- The current queue occupancy: $Q(t)$

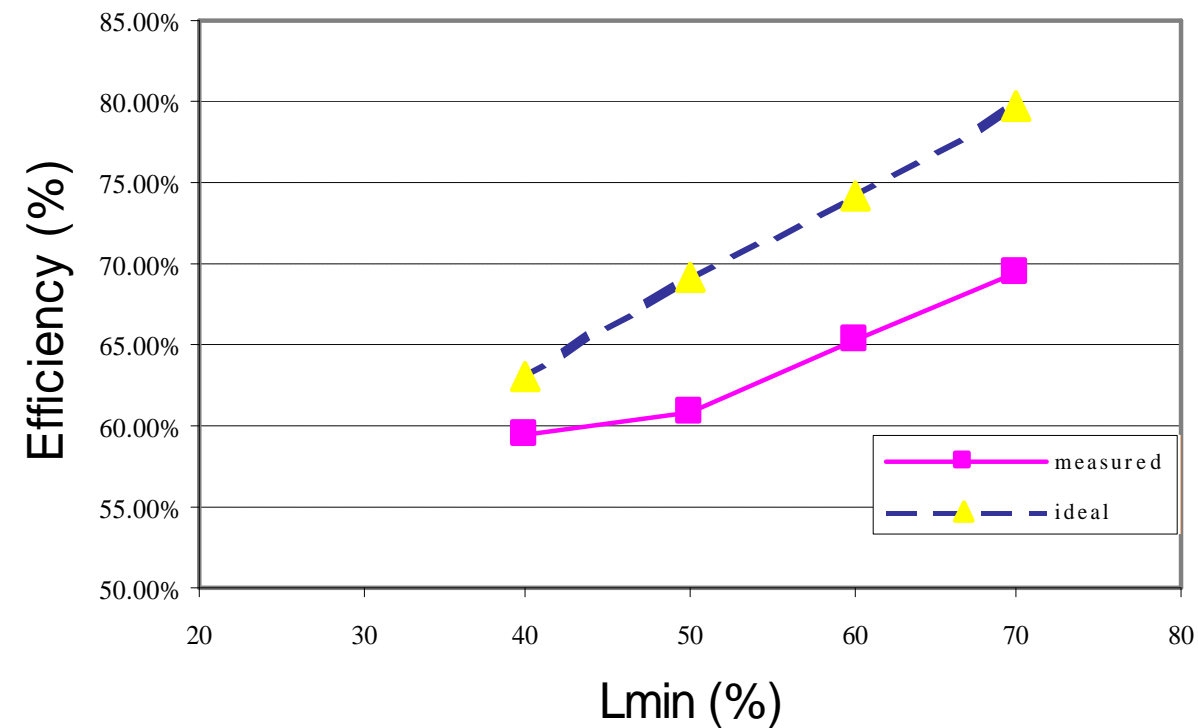
When $Q(t) > S_q(t)$ congestion appears, the measurement - based algorithm is excluded and some additional operations are executed:

- a bandwidth update, so that S'_q becomes greater Q
- if queue occupancy is still greater than the threshold due to further arrivals, a new update is performed
- when congestion finishes, after a guard time T_g , the normal algorithm is used again

Performance - efficiency

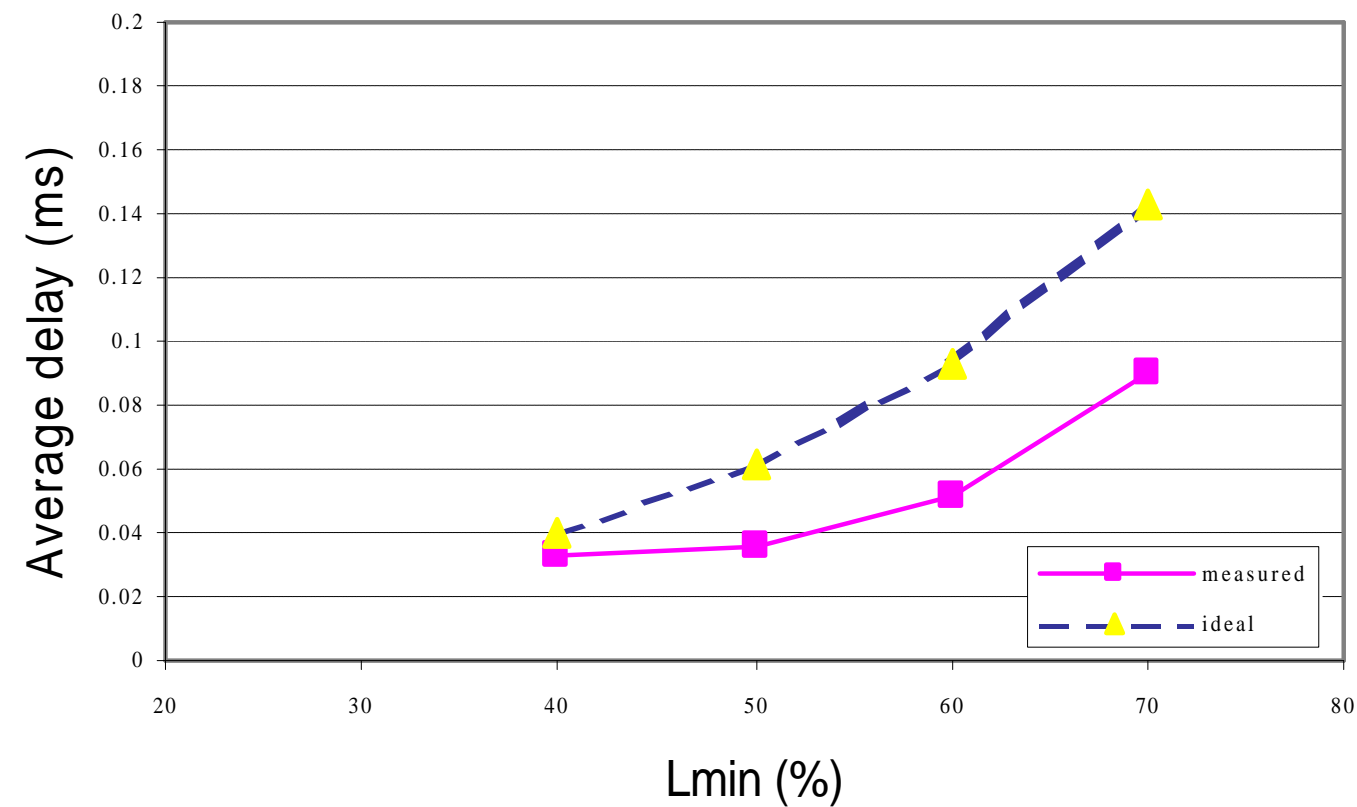
Type	# sources	T_{ON}	T_{OFF}	Peak transmission rate
1	60	Pareto a=1.2, k=0.4	Pareto a=1.2, k=0.6	32Kbit/s
2	60	Pareto a=1.2, k=0.04	Pareto a=1.2, k=0.6	128Kbit/s

Traffic configurations used in the simulations:
60 Pareto sources



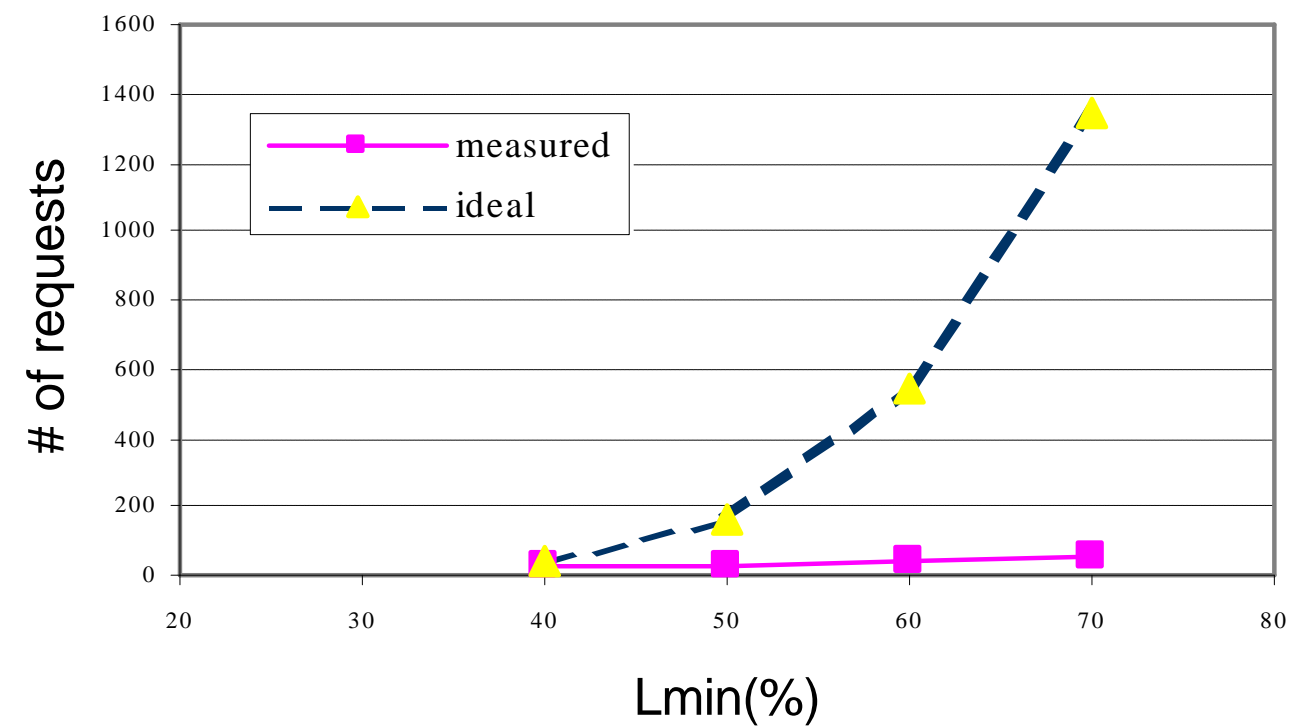
Comparison of **efficiency** between the measurement based algorithm and the ideal approach with Traffic Type 1, $S=0.8s$, $T=10$, $i=d=10\%$, $I_{max}\%=90\%$

Performance - average delay



Comparison of [average delay](#) for the measurement based algorithm and the ideal approach with Traffic Type 1, $S=0.8s$, $T=10$, $i=d=10\%$, $I_{\max}\%=90\%$

Performance - number of interactions



Comparison of the [number of interactions](#) for the measurement based algorithm and the ideal approach with Traffic Type 1, $S=0.8s$, $T=10$, $i=d=10\%$, $I_{\max}\%=90\%$

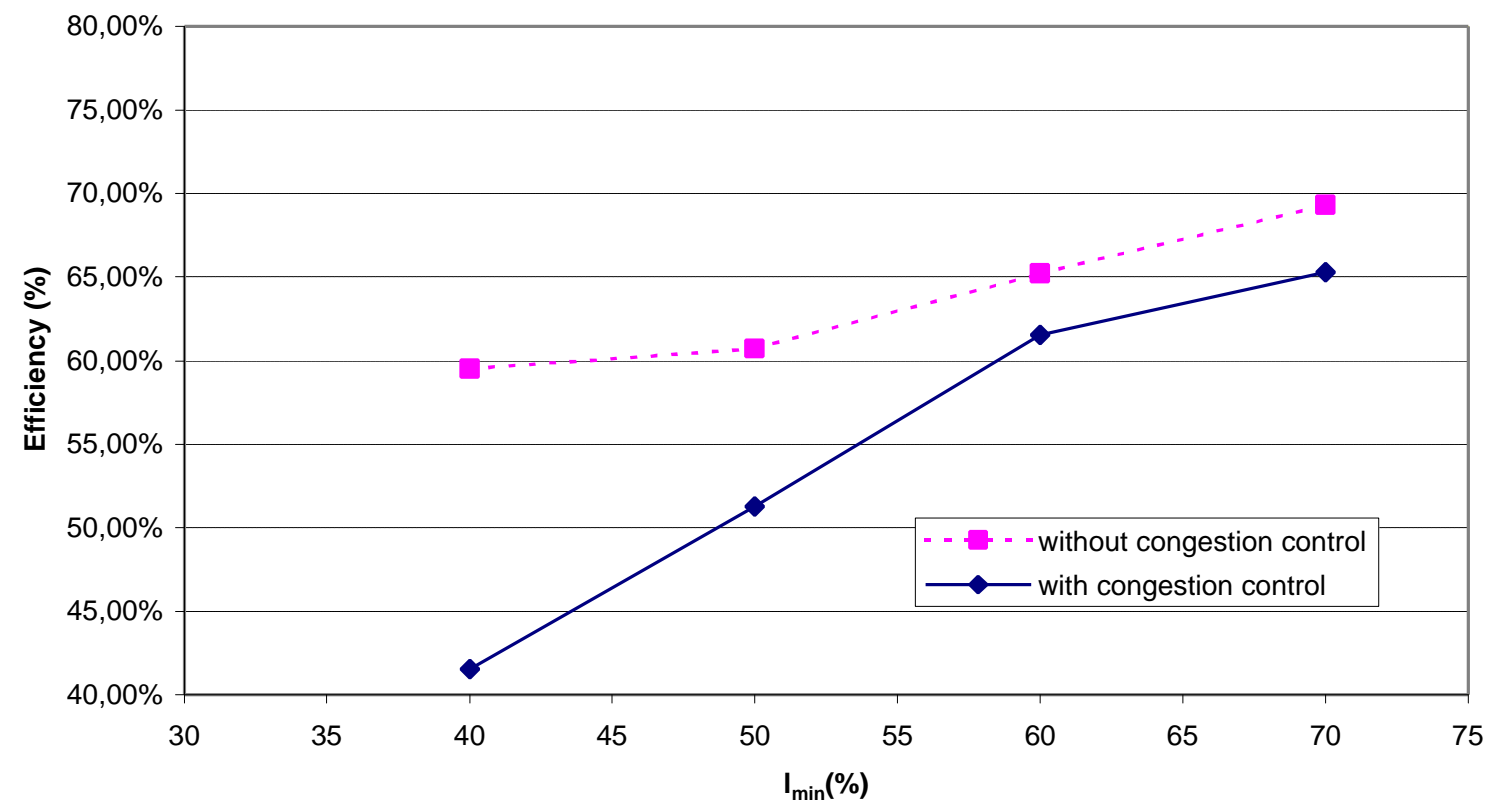
Congestion control & delay

The congestion management algorithm lowers the delay time

Type	(S,T)	L_{\min} %	L_{\max} %	without control [ms]	With control [ms]
1	(0.8s, 10)	70%	90%	61.8	34.5
2	(0.8s, 50)	60%	80%	37.6	22.1

Maximum delay: comparisons for the measurement based system with and without **congestion control** ($i=d=10\%$, $T_S=15\text{ms}$, $M_S=20\%$, $T_I=1.6\text{s}$)

Congestion control & efficiency



Comparison in terms of **efficiency** as a function of the lower threshold for type 1 traffic for the dynamic allocation algorithm with **congestion control** ($S=0.8s$, $T=10$, $I_{max}\%=90\%$, $i=d=10\%$, $T_S=15ms$, $M_S=20\%$, $T_I=1.6s$).

Conclusions

- A threshold-based algorithm for QoS was implemented to manage aggregated flows of traffic with efficiency and scalability in mind
- The optimization of the system is not easy, many parameters must be considered, also with serious constraints due to possible oscillations
- The congestion management algorithm decreases the delay time
- Future work: other algorithms for dynamic bandwidth allocation are to be investigated to lower the number of parameters and make system configuration easier