Reliable and interactive protocol for short messages over the EDS service differentiation

Benjamin Gaidioz, Pascale Primet

INRIA

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

Reliable and interactive protocol for short messages over the EDS *service differentiation*

Benjamin Gaidioz, Pascale Primet

N° 5031

December 2003

THÈME 1

ISSN 0249-6399 ISRN INRIA/RR--5031--FR+ENG

Reliable and interactive protocol for short messages over the EDS service differentiation

Benjamin Gaidioz*, Pascale Primet†

Thème 1 — Réseaux et systèmes
Projet RESO

Rapport de recherche n° 5031 — December 2003 — 11 pages

Abstract: The TCP/IP stack has been mainly designed for elastic traffic (file transfers). It is nowadays recognized that it is not able to efficiently support traffic patterns with completely differing requirements (e.g. applications with delay requirements). Service differentiation at the flow agregate level (DiffServ) is a promising way to implement some form of IP QoS because it is robust and scalable. The EDS PHB is a Diffserv PHB based on both loss rate and delay proportional differentiation. In this article, we start from a network layer implementing the EDS service differentiation and present a specific transport protocol that realizes the reliable and interactive transfer of short messages.

Key-words: IP quality of service, service differentiation, Equivalent Differentiated Services, interactivity over IP

* Université Lyon 1, LIP RESO (UMR CNRS-INRIA-ENS-UCBL N°5668)
† LIP RESO (UMR CNRS-INRIA-ENS-UCBL N°5668), INRIA

Unité de recherche INRIA Rhône-Alpes
655, avenue de l'Europe, 38330 Montbonnot-St-Martin (France)
Téléphone : +33 4 76 61 52 00 — Télécopie +33 4 76 61 52 52

Protocole pour le transport fiable et interactif de messages courts sur la différenciation de service EDS

Résumé : La pile de protocoles TCP/IP a été conçue principalement pour transporter du trafic « élastique » (transferts de fichier). Aujourd'hui, il est admis qu'elle ne convient pas pour de nouvelles catégories de trafic aux besoins très différents (par exemple des flux temps-réel avec des contraintes de délais). La différenciation de services au niveau d'agrégats de flux (DiffServ) est une architecture prometteuse pour mettre en oeuvre une forme de qualité de service IP car elle est robuste et s'étend bien à un grand nombre de noeuds. EDS est un PHB Diffserv qui s'appuie sur la différenciation proportionnelle en délai et taux de perte. Dans cet article, en nous appuyant sur le modèle de différenciation de service EDS, nous présentons un protocole de transport qui effectue le transport fiable et interactif de messages courts.

Mots-clés : qualité de service sur IP, différenciation de services, Equivalent Differentiated Services, interactivité sur IP

1 Introduction

The network layer of Internet (IP) provides a simple, robust and effective packet forwarding service. The TCP/IP stack has been mainly designed for elastic traffic (e.g. FTP). It is nowadays recognized that it is not able to efficiently support new traffic patterns with different requirements in terms of speed, latency and reliability (e.g. real-time applications, interactive applications, bulk transfers, etc.). It is commonly accepted that IP needs to be extended with a form of quality of service (QoS). In the case of IP, adding QoS has to be done carefully in order to maintain the efficiency and robustness of the network.

The DiffServ architecture [1] is convincing as a QoS approach on a large scaled network. Its principles keep the network layer simple and robust. In the core network, IP packets are expected to be marked with a specific *class identifier*. This identifier selects a forwarding treatment met by the packet each time it crosses a router. From the core network point of view, there is a very small number of classes, there is no resource reservation between hops. Marking is done at the edge routers, where the number of flows is sufficiently low to apply marking rules without losing much performance.

A characteristic that makes TCP/IP so appealing is its ease of use: a host can plug into the network and use it immediately. According to its design philosophy [2], IP attempted to provide a basic *building block* out of which a variety of types of service could be built. The decision was an extremely successful one, which allowed the Internet to meet its most important goals.

In this report, we rely on the implementation of the EDS service differentiation [8, 7] at the IP level (see in the technical report [7] for details). This architecture provides a best-effort service differentiation which is not sufficient to satisfy most of the applications needs. Thus, it needs to be extended with end-to-end protocols which use the services it provides in order to ensure some higher level guarantees to applications. Then we present an adaptive transport protocol based on TCP which aims at providing a shorter transfer time than the plain TCP for short file transfers in sect. 3. In sect. 4, we present related work. Finally, in sect. 5, we give conclusions and future work.

2 Quick overview of EDS

Fig. 1 gives the intuition of the concept of Equivalent Differentiated Services. It shows both the service provided by a best-effort router and an EDS router. Roughly, the EDS router provides a set of N classes, each one experiencing a different level of performance in both queuing delay and loss rate. Considering a single performance criteria (delay or loss rate), the range of performance is centered around an average performance, which is the performance all packets would have obtained through a plain best-effort router. Thus, the performance is directly linked to the router load. Moreover, there is an asymmetry between delay and loss rate performance: The class which obtains the i^{th} best performance in delay obtains the i^{th} worst performance in loss rate.

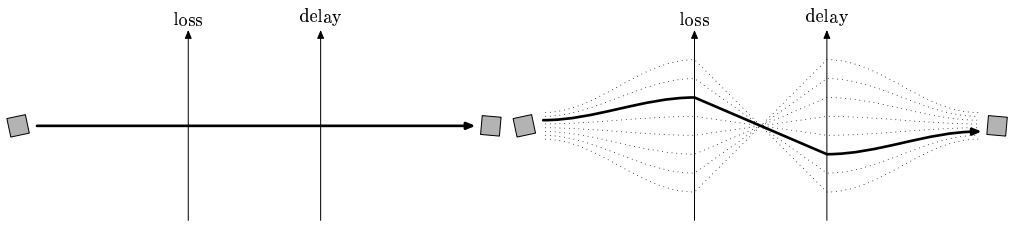

Figure 1: A packet (gray square) is crossing a best-effort router (left) or an EDS router (right). Through the best-effort router, the packet experiences a given queuing delay and a given loss probability. Through an EDS router, the packet is member of a service class among N classes, where it experiences a specific queuing delay and a specific loss probability, both being relatively better or worst than the performance through a best-effort router.

The EDS service differentiation proposal has been designed by starting from the Internet protocol design principles which gave to IP its robustness, ease of deployment and ease of use. Thus, EDS provides *best-effort* service differentiation.

Stronger guarantees on the plain best-effort IP are implemented in end-to-end protocols. For example, TCP guarantees reliability. These protocols use the building block provided by IP to provide a specific service matching the need of specific applications. Since we have defined a best effort service differentiation system from the same design rules, the natural way to implement stronger QoS guarantees has to be done in the protocol layer.

3 Reliable and interactive short messages

This section is related to reliable and interactive short messages. The application has no hard delay requirements and can handle a delayed file transfer. However, the need is to maximize the probability that the transfer ends in a short time. The message length is assumed to be four packets long. Using the typical TCP slow-start, the transfer of four IP packets takes at least three round-trip times (one packet is sent, then two, then one). If a packet is lost, the connection transmits the packet again and it takes some round-trip times more to complete the transfer.

3.1 Simulation of the protocol

The protocol has been implemented in NS [9] and runs on a topology shown on fig. 2 where it meets concurrent random traffic generated according to a Pareto distribution. Random traffic and application packets come from two different links and goes through a router (plain RED or EDS).

- The output link has a bandwidth of 10 Mb/s. Links latency is set to 50 ms everywhere (thus, there is a 100 ms end-to-end delay from e to r). The RED mechanism starts

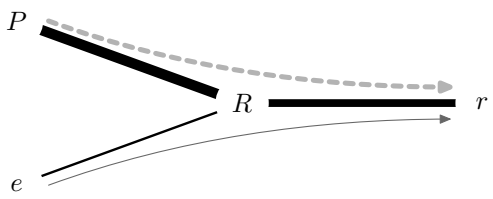

Figure 2: The simulated topology. The router is node R, receiver is node r, random Pareto sources are locate on node P and the application sending four packets is located on host e.

dropping packets when queue length is 20 packets long with a maximum threshold of 300 where the maximum average drop probability is 25%.

- There are eight EDS classes. Delay and loss rate are differentiated so that classes obtain a relative delay ranging from 1 to 8 and a loss rate ranging from 8 to 1. Thus, class 1 is to slowest and class 8 is the quickest. Class 8 has the highest loss probability and class 1 the lowest. The router queue has a capacity of 300 packets.
- There are eight Pareto sources (one per class). The average load generated by Pareto sources is a bit higher than what the output link can handle, so that packets experiences various queuing delays and losses probability. The random sources link has a highest bandwidth so that they can send bursts without experiencing congestion in P. Their average rate is equal to the output link bandwidth.

In the next four experiments, we vary the differentiation settings and protocol behavior. We show the completion time distribution graph. For each file transfer, we compute the completion time. From these statistics, we draw a graph showing the share of transfers which completed in a delay shorter than t, for all t between zero and a maximum delay of seven seconds. In the document, we will consider it as a *probability* for a connection to end in a time shorter than t. We usually run the experiment twice in order to vary the network load.

3.1.1 Impact of delay differentiation on the TCP completion time

In this experiment, we set a visible delay differentiation. Loss rate differentiation is not "off". Coefficients are set to equal values so that the scheduler ensures an equal loss rate in all classes. Thus, packets are not dropped like in a tail-drop network. We show the evolution of the probability on fig. 3.

Since loss rate is equal in all classes, all probabilities tend to the same values with the difference that quick connections grow more rapidly to these values. This is due to the fact that quick connections meet so lower delay and jitter that their end takes place in a narrower range. That is why the probability grows quickly for a quick class and slowly for a slow class.

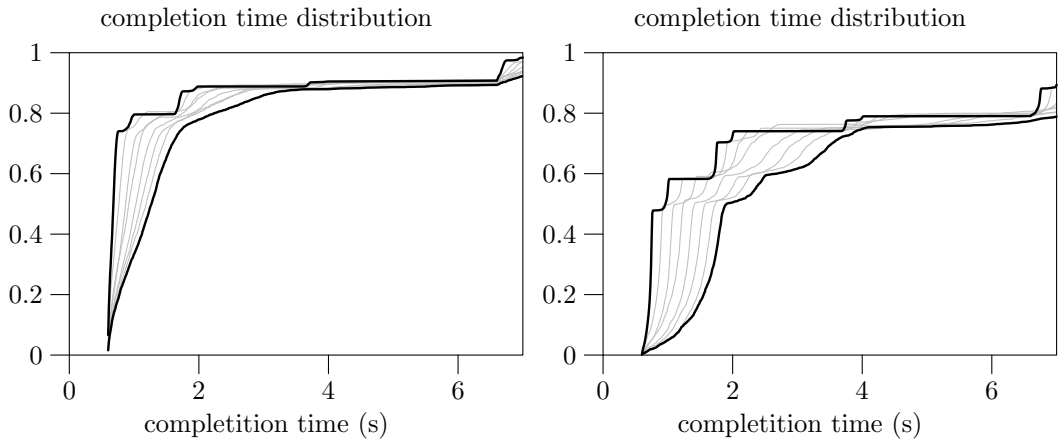

Figure 3: Completion time distribution when only delay differentiation is practiced.

However, they both reach the same level. Similar results are obtained with the experience running with a high load. The probabilities are simply lower because of the high loss rate.

3.1.2 Impact of loss rate differentiation on the TCP completion time

In this experiment, we set a visible loss rate differentiation and no delay differentiation. When all classes have the same delay differentiation parameter, they are not differentiated in delay and this results in a FIFO scheduling. We show the evolution of the probability on fig. 4.

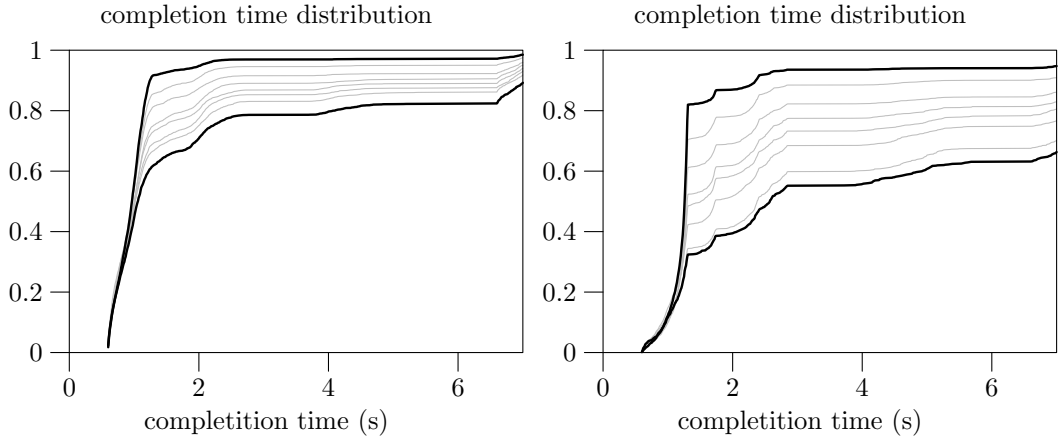

Figure 4: Completion time distribution when only loss rate differentiation is practiced.

A smaller part of connections running over a high loss rate class manages to end in a specific time t. That is why curves are similar with a scaling factor. However, because of dropping, a larger part has to retransmit packets. Thus, a larger fraction ends in a larger time. Thus, curves grow but classes with a higher loss rate grow more rapidly (the slope is a bit greater). In the case of a high load, all probabilities are lowered. Since absolute loss rate is much higher for high loss rate classes, the distance between extremes classes grows.

3.1.3 Impact of EDS on the TCP completion time

When setting both differentiations at the same time, the result is a combination of both. The quicker classes are also those which obtain the higher loss rate probability. Since they are quick, their probability grows quickly. However, because of the loss rate probability, it grows less than slow classes. Delay differentiation has no impact on the level the probability reaches, it has just impact on the slope of the curve. Thus, for high values of t, the probability is lower for quick classes although for low values of t, it is higher (because of the slope). See fig. 5

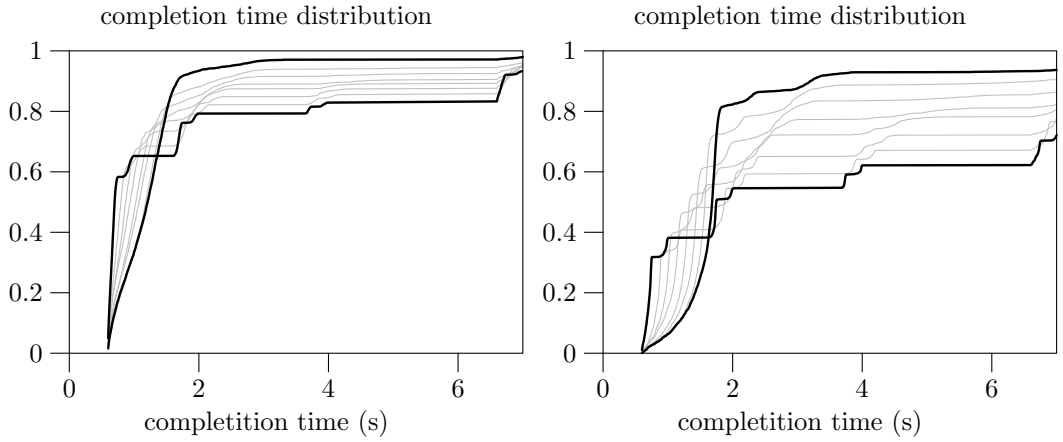

Figure 5: Completion time distribution when both delay and loss rate differentiation is practiced.

Starting from this first experiment, we can conclude that whatever the load, there is a duration t before which the probability is higher with a quick class and lower for a slow class. Beyond this duration limit, the probability is higher using a slow class than a quick class. Thus, setting a class identifier on a plain TCP connection sending a few packets is choosing a probability distribution.

- In the case where the application has a need for interactivity, it can try using a quick class in order to have more chances to end quickly.

- In the case where it has no need for interactivity, it can ensure an acceptable transmission time using a slow class.

3.1.4 Performance of a more aggressive TCP protocol

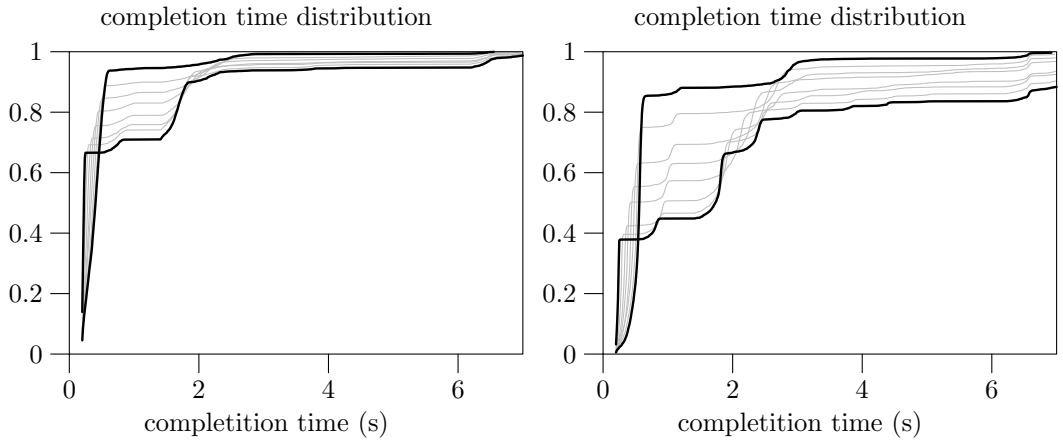

Figure 6: Completion time distribution with both delay differentiation and loss rate differentiation and a more aggressiv protocol. The completion time is shorter

In this section, we vary the behavior of the protocol by making it more aggressiv. Instead of slow-starting, the protocol sends all four packets in one burst. This increase the probability to end before a specific time t because the protocol has now a non null probability to end in one round-trip time. See fig. 6

3.2 Interactive reliable protocol for short messages

In order to minimize the completion time, the protocol starts with a larger window of four packets, where the entire message can fit into. Thus, it is possible that the message is received in one round-trip time. The protocol uses the class with the highest drop probability while sending this burst. If load is high, it will lose packets because of loss rate differentiation, that will results in a smaller burst. If load is low, a four packets burst does not increase the overall load too much if the network is sufficiently well provisioned. Moreover, the connection does not continue in slow-start since all packets have been sent.

Retransmitted packets use the slowest class, where drop probability is low. Thus, in the case of low load, the file transfer has a chance to end in a very short time. In the case of high load, it should lose packets. The protocol uses a class where the loss rate is lower and takes a chance to end in an acceptable time.

When executing the algorithm over an empty network with a fixed latency, it is obvious that this protocol has a better performance than the plain TCP for short files. However,

when using it for large file, it becomes very bad because of the window limitation. This is shown on fig. 7 where the theorical tranmission time of both protocol is computed for an increasing file size.

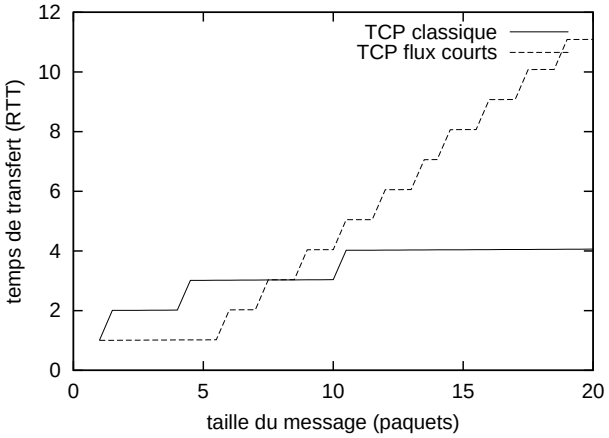

Figure 7: Transmission time for both TCP and the modified version.

3.3 Simulation of the protocol

Fig. 8 shows the probability graph of completion times using the plain TCP on best-effort, the aggressive protocol on EDS and the aggressive protocol on best-effort. Thanks to the burst at the beginning of the connection, the EDS-aware protocol has a chance to end in a short time, what is impossible using TCP since it requires three round-trip times. The aggressiveness of the protocol is affected by the loss rate probability of the class it uses while in slow start. Indeed, it obtains a lower probability to end in a short time than the same protocol running on best-effort. Retransmissions occurs using a class with a lower loss rate. Thus, the probability to end in a longer time is acceptable compared to the plain TCP.

3.4 Conclusion

The optimized protocol let the user a relatively high probability to end in a short time thanks to the burst at the beginning of the connection. However, it uses the class with the highest loss probability in order to lose packets if the network load is high. Then, if some packets were lost, it continues using a slow class with a lower drop probability, in order to increase the probability to end while in a high network load.

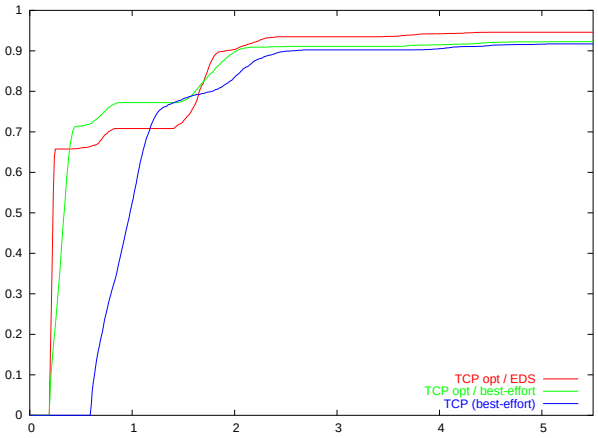

Figure 8: Probability of completion time. The x axis is in seconds.

4 Related Work

Similar work is conducted on top of different service differentiation systems.

- The authors of proportional differentiated services [3] have seen the need to match stronger QoS requirements from the end-to-end viewpoint since proportional differentiation does not provide absolute guarantees. An adaptive protocol [4] has been designed to dynamically select the best class and then provide an absolute guarantee in delay.
- The TCP-friendly differentiated services marking [5] is based on the same way to consider service classes as a building block protocols can use with a fine grained adaptation algorithm. The system is based on the standard *assured* service where classes get different drop level probability (no delay differentiation). Their objective is similar to us since they do not aim at providing strong guarantees but improving the overall performance (they prove that they obtain less timeouts).

In both cases, the underlying service differentiation layer provides privileged classes. The architectures must be deployed with an access control system (or marking has to be done by a trustworthy border entity) to ensure a cordial use of services.

5 Conclusion and future work

We have presented both the EDS best-effort service differentiation system and a transport protocol built on top of it.

The EDS system has been designed by mapping the IP design philosophy to service differentiation. The same way TCP has been designed to provides reliability on top of the unreliable network layer IP, we have presented a protocol which provides specific soft quality of service properties to applications. It ensures that the transfer time is shorter for short files than with TCP.

From a more practical point of view, since we implemented EDS with proportional differentiation schedulers in Linux [6], we are also going to start the implementation of the protocol presented in this article and see how it runs in real life.

References

[1] Steven Blake, David Black, Mark Carlson, Elwyn Davies, Zheng Wang, and Walter Weiss. An architecture for differentiated services. Internet Request For Comments RFC 2475, Internet Engineering Task Force, December 1998.

[2] David D. Clark. The design philosophy of the DARPA internet protocols. In *Proceedings of Special Interest Group on data Communication (*SIGCOMM*)*, number 4 in Computer Communication Review, pages 106–114, Stanford, CA USA, August 1988. ACM, ACM Press.

[3] Constantinos Dovrolis and Parameswaran Ramanathan. A case for relative differentiated services and the proportional differentiation model. IEEE *Network*, 13(5):26–34, September 1999.

[4] Constantinos Dovrolis and Parameswaran Ramanathan. Dynamic class selection: From relative differentiation to absolute qos. In *Proceedings of the International Conference on Network Protocols (*ICNP*)*. IEEE, November 2001.

[5] Azeem Feroz, Amit Rao, and Shivkumar Kalyanaraman. A TCP-friendly traffic marker for IP differentiated services. In *Proceedings of International Workshop on Quality of Service (IWQoS)*, Pittsburgh, PA USA, June 2000. IEEE/IFIP.

[6] Benjamin Gaidioz, Mathieu Goutelle, and Pascale Primet. Implementation of IP proportional differentiation with waiting-time priority and proportional loss rate dropper in linux. Technical Report RR-4511, INRIA, August 2002.

[7] Benjamin Gaidioz and Pascale Primet. The equivalent differentiated services. Technical Report RR-4387, INRIA, February 2002.

[8] Benjamin Gaidioz and Pascale Primet. EDS: A new scalable service differentiation architecture for internet. In *Proceedings of International Symposium on Computer Communication (*ISCC*)*, pages 777–782, Taormina, Italy, July 2002. IEEE.

[9] Network simulator NS-2 web site. `http://www.isi.edu/nsnam/ns/`.

Unité de recherche INRIA Rhône-Alpes
655, avenue de l'Europe - 38330 Montbonnot-St-Martin (France)

Unité de recherche INRIA Futurs : Domaine de Voluceau - Rocquencourt - BP 105 - 78153 Le Chesnay Cedex (France)
Unité de recherche INRIA Lorraine : LORIA, Technopôle de Nancy-Brabois - Campus scientifique
615, rue du Jardin Botanique - BP 101 - 54602 Villers-lès-Nancy Cedex (France)
Unité de recherche INRIA Rennes : IRISA, Campus universitaire de Beaulieu - 35042 Rennes Cedex (France)
Unité de recherche INRIA Rocquencourt : Domaine de Voluceau - Rocquencourt - BP 105 - 78153 Le Chesnay Cedex (France)
Unité de recherche INRIA Sophia Antipolis : 2004, route des Lucioles - BP 93 - 06902 Sophia Antipolis Cedex (France)

Éditeur
INRIA - Domaine de Voluceau - Rocquencourt, BP 105 - 78153 Le Chesnay Cedex (France)
http://www.inria.fr
ISSN 0249-6399