LABORATOIRE D'INFORMATIQUE DE L'UNIVERSITE DE FRANCHE-COMTE

EA 4269

Distance-Dependent RED Policy (DDRED)

Eugen Dedu — Sébastien Linck — François Spies

Rapport de Recherche no RR 2011–????

THÈME 4 – Avril 2007

Distance-Dependent RED Policy (DDRED)

Eugen Dedu, Sébastien Linck, François Spies

Thème 4
OMNI

Avril 2007

Abstract: The network quality of service (QoS) and the congestion control of the transport protocol are important parameters for the performance of a network data transfer. To this end, routers use various queue policies for packet dispatching, and all of them must deal with packet drop. We propose a new algorithm for packet drop in routers. Given that a packet drop *wastes* all the network resources it has already used, we propose a new policy which favors packets with higher distance from source. It can be simply integrated on top of tail drop or RED (with or without ECN) queue policies. Simulations with NS2 show that long flows are indeed favored compared to short flows, and lead to higher overall resource utilisation without sacrificing TCP fairness.

Key-words:

Laboratoire d'Informatique de l'Université de Franche-Comté,
Antenne de Montbéliard — UFR STGI,
Pôle universitaire du Pays de Montbéliard,
25200 Montbéliard Cedex (France)
Téléphone : +33 (0)3 81 99 47 43 — Télécopie +33 (0)3 81 99 47 91

Politique RED dépéndante de la distance (DDRED)

Résumé :

Mots-clés :

Laboratoire d'Informatique de l'Université de Franche-Comté,
Antenne de Montbéliard — UFR STGI,
Pôle universitaire du Pays de Montbéliard,
25200 Montbéliard Cedex (France)
Téléphone : +33 (0)3 81 99 47 43 — Télécopie +33 (0)3 81 99 47 91

I. Introduction

TCP is a reliable protocol [1]. This reliability is achieved through retransmission of lost packets. Additional traffic is thus generated, leading to sender-side congestion window reduction, hence throughput reduction and greater transmission times.

TCP congestion control is crucial to avoid the saturation of various types of equipment between the source and the destination. If a distant TCP flow makes a router enter congestion, there will be an ejection of packets. When a packet is ejected, it requires retransmission, hence it has unnecessarily used physical resources.

Our proposition, DDRED (for Distance-Dependent RED), consists in eliminating packets having consumed fewer physical resources and thus coming from sources nearer to the congested router. By eliminating the packets from this source, retransmission is faster.

The TTL (*Time To Live*) field in the IP header [2] already contains information about how far the source is from the actual router. But this field may be initialized at wish by the sender [2], hence the initial TTL value cannot be known by routers. Therefore a method to measure the distance is simply to add two fields in IP header giving the initial TTL and its value when the packet reaches the destination (particularly useful in multi-path routing). Packet management is performed by queue policies in routers. DDRED is the implementation of our mechanism in RED policy [3] (Random Early Detection), but it can be integrated into other policies, such as tail drop. In this article, we present NS2 simulations[1] done on RED.

II. Related Work

There are several techniques to achieve higher overall resource utilization and/or to adapt the application requirements to the dynamic network conditions. Some techniques act on the host side, such as the congestion control of TCP [1], others on the router side. In the following we focus on techniques on the router side.

Furthermore, some techniques are content-based [4]. For example, in an MPEG video streaming flow, routers recognize the type of frames (I, P or B) and try to prevent I frame dropping (which are critical, since P and B depend on I).

Several techniques which are not content-based exist. Some of them use various scheduling algorithms, such as FQ (Fair Queue) and WRR [5] (Weighted Round Robin).

Others use various queue management policies, such as RED and its derivatives. Cisco's WRED [6] (Weighted RED) includes IP preference in RED by providing separate thresholds and weights for different IP addresses. DSRED [7] (Double Slope RED) improves the performance of RED by dynamically changing the slope of the RED probability curve as a function of the congestion level.

[8] and [9] use NS2 to validate their proposition. In [8] several queue sizes with different parameters are used when the queue is scarcely filled, leading to a faster RED processing. MRED (Modified RED) [9] optimises RED for bursty traffic.

In best-effort networks, equipments do their best to deliver packets, but there is no guarantee that they will arrive at destination, neither can one be sure of the time they will take to arrive. DiffServ [10] is a method which tries to guarantee a QoS on such networks, by dividing traffic into groups with different priorities on routers.

AECN [11] (Adaptive ECN) adds to the TCP header a field containing information about the *RTT* of a flow. The field is set by senders and read by routers. Routers have a set of RTT ranges and corresponding flow sub-queues. Each packet is put in the appropriate sub-queue, based on its RTT. Unfortunately, the RTT gives the return time and is *independent* of the location of the packet in its trip, while in DDRED the distance gives the number of routers (resources) involved *up to* the router which would drop the packet.

III. Background

We here present a few congestion control mechanisms concerning our study: DropTail, RED and ECN.

A. DropTail and RED Router Policies

Routers may become congested. In such cases, some packets must be dropped from the input queue(s). Several policies exist in order to decide which packets will be rejected. Two very common such policies are tail drop and RED [3].

1) Tail drop policy: In tail drop, a new packet is rejected if and only if the queue is full. It is a very fast decision and hence it is suited to backbone routers.

2) RED policy: The aim of the RED (Random Early Detection) [3] policy is twofold:

- to *prevent* congestion, as a long-term filling of queues;
- at the same time to allow the usual TCP's bursts of traffic to pass, as a short-term filling of queues.

For *each* packet, RED measures the average queue size in the last N laps of time (e.g. last 2 seconds), hence it is an AQM-based (Active Queue Management) policy. The use of the average, and not of the instantaneous queue length, allows the router to absorb bursts of traffic.

RED uses two variables: $\mathrm{th}_{\min}$ and $\mathrm{th}_{\max}$, with $0 < \mathrm{th}_{\min} < \mathrm{th}_{\max} <$ full, where full is the maximum queue size. As shown in figure 1, when the average queue size q_{ave} is smaller than $\mathrm{th}_{\min}$, the new packet is added to the queue. When the average queue size is between $\mathrm{th}_{\min}$ and $\mathrm{th}_{\max}$, the new packet is marked with a linear probability p:

$$p = p_{\max} \times \frac{q_{\text{ave}} - \mathrm{th}_{\min}}{\mathrm{th}_{\max} - \mathrm{th}_{\min}} \quad (1)$$

where $p_{\max} < 1$. When the average queue size is greater than $\mathrm{th}_{\max}$, the packet is always marked. In the gentle variant of RED [12], used for our experiments, the probability increases linearly from $p_{\max}$ to 1 when the queue filling varies from $\mathrm{th}_{\max}$ to twice $\mathrm{th}_{\max}$, as shown in the figure.

[1]The simulation scripts can be found at `http://lifc.univ-fcomte.fr/~linck/ddred`.

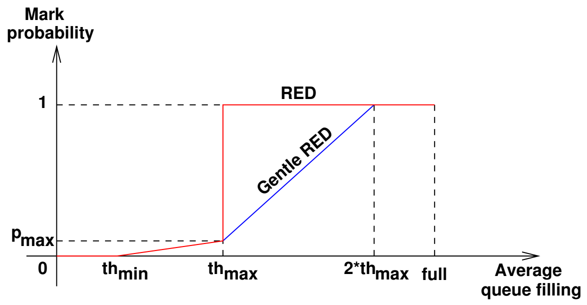

Fig. 1. The probability of marking a packet in the RED policy.

In pure RED, the marking means deletion. See the next section for the meaning of the marking when the ECN mechanism is used.

The RED policy has the advantages of reducing congestion and maintaining a reasonable queue size. On the other hand, as it uses CPU resources, it slows down the router and increases the latency, hence it is suited to edge routers.

B. *ECN Mechanism*

ECN (Explicit Congestion Notification) [13] is a mechanism allowing TCP senders to be notified when a congestion appears. It involves the sender, the receiver and RED routers. In short, routers, when congested, mark some bits in the IP header of packets instead of dropping them, and forward them to the network. The destination marks its packets for the source, which thus becomes aware of the congestion.

More precisely, it works as follows: the decision as to whether to use or not ECN is initiated by the sender during connection initialization. If all the involved network components (source, routers etc.) agree, the transfer is ECN-based.

During data transmission, a router may process an ECN packet and a non-ECN packet differently. This difference appears only for RED routers, when the queue size is between $\mathrm{th}_{\mathrm{min}}$ and $\mathrm{th}_{\mathrm{max}}$. In this case, a non-ECN packet is rejected based on a probability (see previous section), while an ECN packet is marked and *added* to the queue (based on the same probability). When the destination receives a marked packet, it marks *each* of its packets to the sender. Finally, when the sender receives a marked packet, it enters the congestion phase by reducing its flow rate and informs the receiver to stop sending marked packets.

IV. Adaptive Priority Mechanism

In a point of the network, we define the distance d_r of a packet as being the number of routers between this point and the source of the packet. If a packet is to be eliminated, it is preferable to eliminate a packet with a small d_r value and to keep the packet with a large d_r value. We use this distance to develop new types of queue management policies.

A. *Principle*

Bearing this in mind, we can use the TTL field of 8 bits of the IP header [2]. The TTL is the lifespan of a packet in a network, a kind of expiry date. Each time the packet enters a router, its TTL is decremented, and when it reaches the critical value of zero, the packet is destroyed, even if it has not yet reached its destination. This field is initialized at wish by senders [2], and its initial value cannot be known by routers. Therefore, we need our own IP fields, implemented as IP options for example. We present two methods to implement these fields.

The first method is to take into account the distance d_r. This can be done using one additional field in the IP header: the initial TTL (TTL_i). TTL_i is set in each packet with the initial value of the classical TTL field of the *source* machine. By computing the difference between this new field and the TTL read on the router, we can determine that the distance $d_r = TTL_i - TTL$ covered by the packet.

The second method is to take into account the percentage of the route covered. It involves two fields: TTL_i and TTL_f, the latter (final TTL) being set with the TTL read on the destination of the previous packet. In case of multi-path routing strategy over the network the TTL_f is the distance of the longest path used between the source and the destination. (TTL_f of the very first packet, SYN, is not set.)

As the second method requires more resources (CPU time and one more byte in the IP header), we choose here the first method in our simulations.

B. *Implementation*

The router uses the covered distance to choose the packet to be dropped. We implement our mechanism in two common queue management policies used in routers:

Tail Drop queue type: The characteristic d_r of an arriving packet and of the last N packets of the queue can be compared. The one whose distance is smaller, therefore pertaining to the flow having the nearer source, is rejected, whereas the other is put at the tail of the queue.

RED queue type: In this policy the decision of packet ejection (leading the router to a congestion state) is based on a probability computing as presented in section III. We do not change the formula but only the packet to which this probability applies. When the probability requires dropping the packet, the router searches the queue for the nearest packet (smallest d_r) and drops it instead of the incoming packet. For ECN flows the same mechanism applies, by exchanging a dropped packet for a marked packet.

V. Case Study

For our simulations, we use Network Simulator version 2.29. We create a patch which changes the ejection policy in the RED algorithm as presented before. For more reliability, each case study (named Sr_n for Simulation run n) has been simulated with different initial random seeds (from 0 to 10), giving different scenarios. Each scenario is simulated twice with the same initial conditions (transfer size and transfer starting time): in the first one, all routers implement the RED algorithm and in the second one the DDRED algorithm.

All the routers use either the RED policy or the DDRED one and we compare the results based on the following criteria:

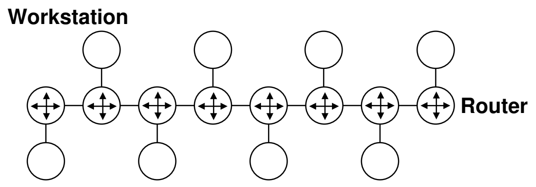

Fig. 2. Bus network topology.

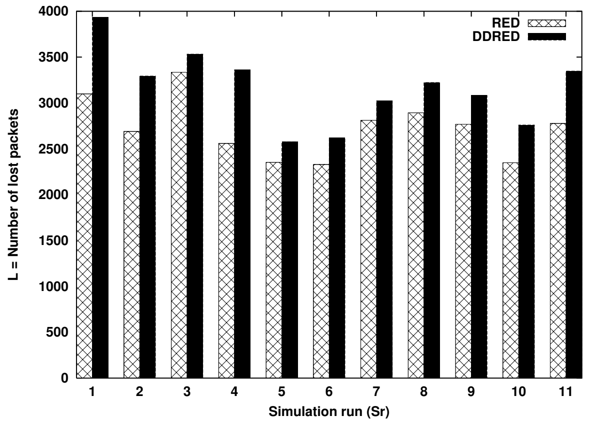

Fig. 3. Bus topology, number of lost packets.

- packet losses, which give information about network resource utilization,
- transfer times, because they are visible to all users. We suppose each connection independent.

A. Bus network

1) Network topology: The network is shown in figure 2. All the links between workstations and routers and between routers have a bandwidth of 10 Mbits/s. 100 FTP transfers, based on TCP New Reno, are created in a period of 60 seconds and the simulation stops when all the transfers are completed. The source and the destination of each of these flows are randomly selected among the workstations. The size of the data transferred is randomly selected (but reproducible) between 100 KB and 6 MB.

2) Results: Figure 3 presents the number of lost packets during the entire simulation. The average number of lost packets is $\overline{L_{RED}} = 2724$ and $\overline{L_{DDRED}} = 3160$. It shows that there are more losses with DDRED. Having more losses is not a bad thing. What is important is not that a packet is lost but that it has consumed resources (router processors and bandwidth), for example a packet lost at the 10th router compared to 3 packets lost at the second router.

To measure the resources consumed, for each distance covered d_r the number of packets lost L_d is totalled and then the number of losses is weighted by their respective covered distance:

$$S = \sum_{d_r=1}^{d_{rmax}} (d_r \times L_d) \tag{2}$$

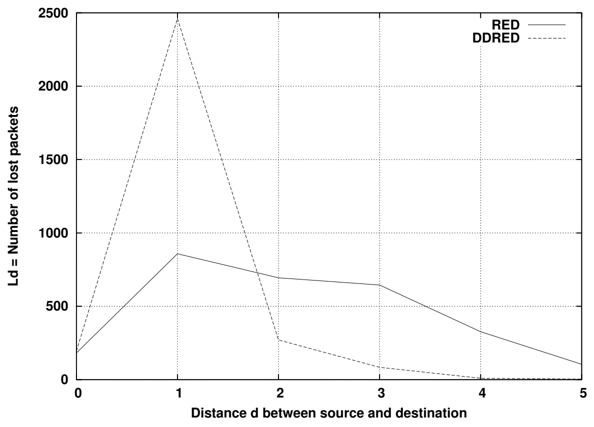

Fig. 4. Bus topology, loss repartition for a simulation run (Sr_6).

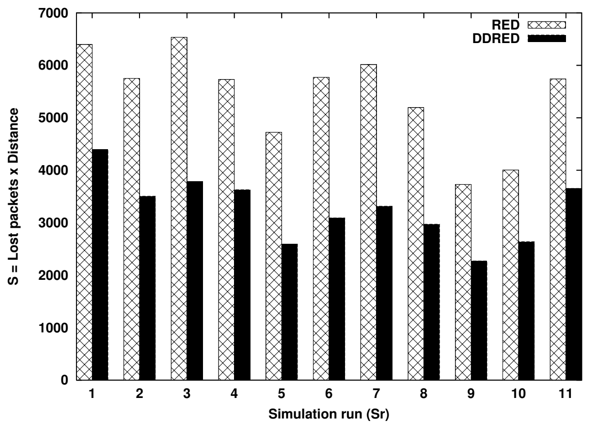

Fig. 5. Bus topology, number of losses weighted by their covered distance.

Figure 4 shows an example of loss repartition $L_d(d)$ for one of the scenarios. As seen, DDRED lost packets are drawn nearer to the source of the transfer. Figure 5 shows the sum for each simulation run Sr_n. We thus save space in the router queue or ”slots” (a slot is the space of one packet with average size in the queue). The average consumed slot number is reduced from $\overline{S_{RED}} = 5418$ to $\overline{S_{DDRED}} = 3259$.

Figure 6 presents the sum of all transfer times for each Sr_n. The sum of the transfer times (difference between the last packet received time T_{end} and the first sent packet time T_{begin}) of the 100 flows is given by:

$$T_{Sr_j} = \sum_{i=1}^{100} (t_{end_i} - t_{begin_i}) \; where \; 0 < j < 12 \tag{3}$$

Generally, it is smaller and shows on average a profit for DDRED of 2.5% ($\overline{T_{RED}} = 3196$ s and $\overline{T_{DDRED}} = 3125$ s). The saved slots in the DDRED queue on router are free for other flows which advance faster.

B. ”Flower network”, realistic simulation

1) Network topology: In order to have a more realistic scenario, we decide to create a simple schematization of it, a

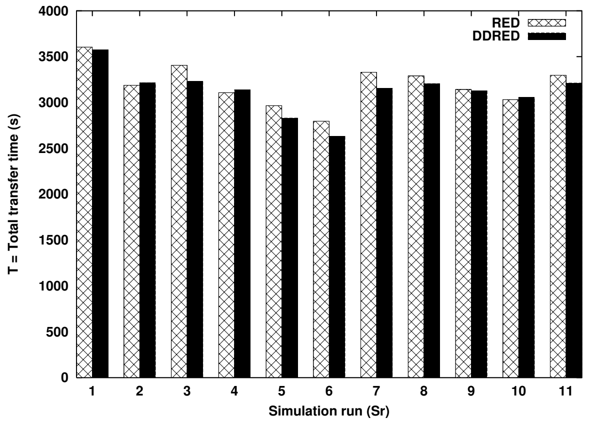

Fig. 6. Bus topology, sum of transfer times.

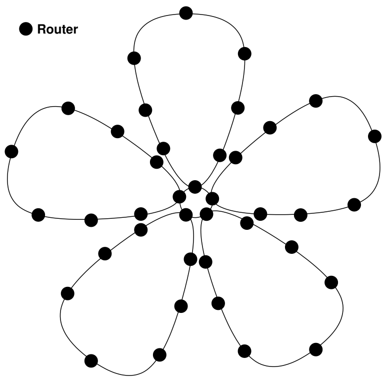

Fig. 7. "Flower" backbone topology.

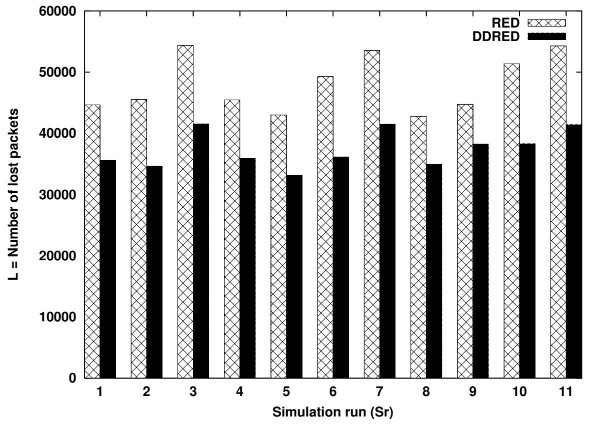

Fig. 8. "Flower" topology, number of lost packets.

"flower" network (see figure 7). According to a small study of the xDSL backbone of a provider[2], most of these networks are built around a center core where several loops are connected. These loops are composed of a small number of routers. The aim of the closed loop is to have a fault tolerance.

We considered a flower with five loops with eight routers on each of them. As in the bus network, only one workstation is connected to a router and connections have the same conditions and parameters (sizes and times of transfers). In this simulation 500 connections are initialized in the same first minute and the simulation ends at the end of the last transmission, as in the first one.

2) Results: Here only the results which differ from the previous simulation are presented. Figure 8 presents the number of packets lost. Contrary to the bus network, fewer packets are lost, approximately 22% ($\overline{L_{RED}} = 48083$ to $\overline{L_{DDRED}} = 37402$. This is due to the form of the network which leads to fewer congested routers.

In figure 9, the number of used "slots" with these queue policies is on average $\overline{S_{RED}} = 152595$ to $\overline{S_{DDRED}} = 78683$ packets. With DDRED, the saved "slots", therefore available for other flows, can grow up to 50% compared to RED.

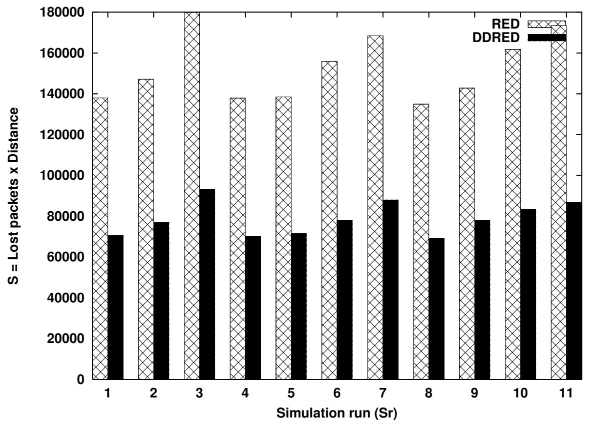

Fig. 9. "Flower" topology, number of losses weighted by their covered distance (slot saving).

[2]http://support.free.fr/reseau/province.png

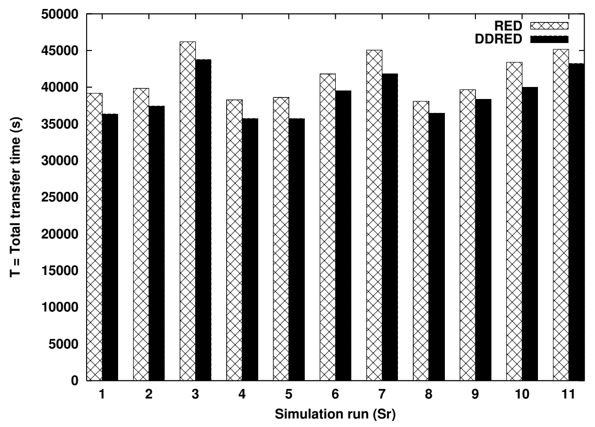

Fig. 10. "Flower" topology, sum of transfer times.

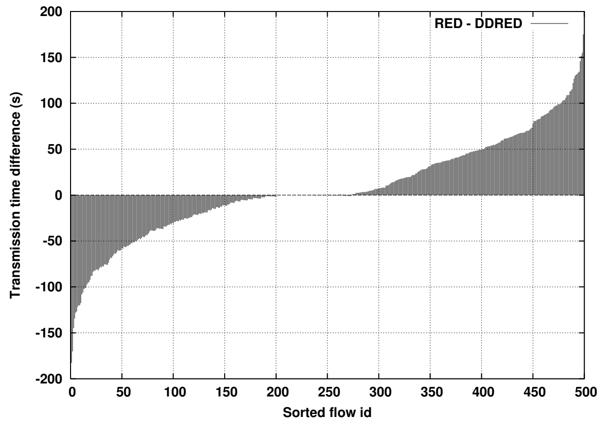

Fig. 11. "Flower" topology, transfer times of each flow for a simulation run (Sr_6).

The sum of the transfer times is on average 6% smaller (see figure 10, where $\overline{T_{RED}} = 41383$ s and $\overline{T_{DDRED}} = 38945$ s). Not all the transfers are faster, but on average time is saved, as shown in figure 11. 237 flows are better and 214 flows are worse compared to RED. The surface of the better flows is 11362 (mean difference time x number of better flows) and the surface of the worst is 8142 (mean difference time x number of worse flows), so DDRED gives a good distribution of the winning and losing flows and it has a better global profit.

VI. Discussion

This section presents a study on the fairness of competing TCP flows. Unlike RED, DDRED does correlate losses, since it always chooses the shortest connection to remove packets. However, this flow is the fastest in packet loss recovery, which reduces its disavantage. Moreover, unlike tail drop, bursts of losses are avoided, because of the use of RED probability. DDRED can be thought of as between tail drop and RED.

A. Network topology

Suppose the following test network, R2 using DDRED:

```
PCs1 --- R1 --- R2 --- PCd
                |
               PCs2
```

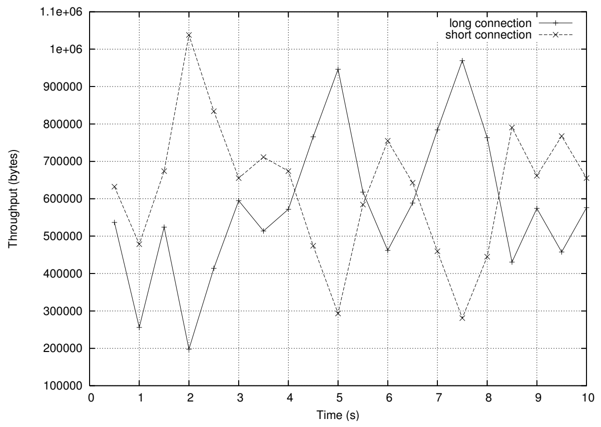

Fig. 12. The two flows have alternatively the highest throughput (DDRED, latency=20ms).

Each link has a latency of 1ms and a bandwidth of 10Mb/s. Two TCP connections, from PCs1 to PCd, and from PCs2 to PCd, start at 0s. The time interval taken into account is from second 0 to second 10 (arrival time), and data size is infinite.

B. Discussion

Given our TTL-based algorithm, PCs1 always has priority over PCs2 on R2.

Stage 1: both connections exist in the queue. When R2 becomes congested, PCs2 decreases its sending rate, but PCs1 continues to increase its rate regularly. This happens until the rate of PCs1 exceeds the R2 processing speed.

Stage 2: only PCs1 exists in the queue. Now, a packet of PCs1 will be rejected. So PCs1 reduces its sending rate, allowing PCs2 to increase its throughput, but PCs2 is faster. Go to stage *1*.

Result of stages: because of its higher RTT, PCs1 cannot always obtain most of the bandwidth.

When the difference of RTT is small, PCs1 is much less penalized and, when penalized, recovers almost as fast as PCs2. This situation is clearly favorable to PCs1, but it does not lead to PCs2 muting. Simulations below show that PCs2 can still gain 15% of bandwidth in such an unfavorable case.

When the difference of RTT is high, PCs1 needs more time to recover from the error of Stage 2, enough time to allow PCs2 to send many packets, even more than PCs1. (This is exacerbated by the fact that packets are marked from up to bottom, hence PCs2 is not informed immediately about the congestion and has time to send even more packets.) This situation, where intervals of PCs1 higher than PCs2 (stage 1) and PCs2 higher than PCs1 (stage 2) alternate, is shown in figure 12, obtained by the simulation presented below.

C. Simulations

Several simulations have been done, with latency of the link PCs1-R1 varying from 1ms to 200ms. Table I presents some results shown as a ratio throughput PCs1 / throughput PCs2.

Latency PCs1-R1 (ms)	1	20	50	100
RED throughput (KB/s)	599/641	235/998	171/1025	123/1079
DDRED throughput (KB/s)	1024/198	577/625	449/686	128/1076

TABLE I

COMPARISON BETWEEN RESULTS OF RED AND DDRED SIMULATIONS.

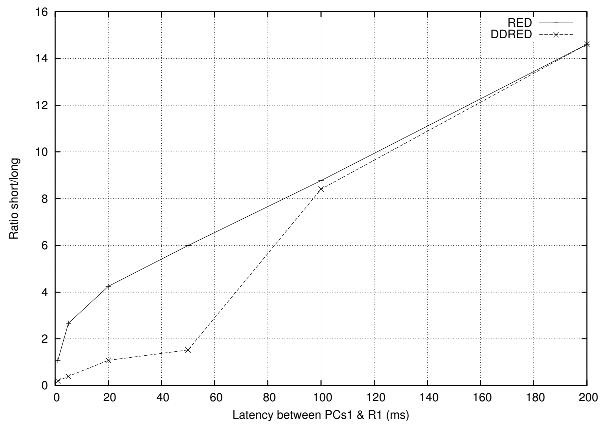

Fig. 13. Ratio between short and long flow bandwidth.

To compare the fairness of DDRED to RED, the ratio of bandwidth between the two flows is shown in figure 13. The nearer to 1 the ratio is, fairer the algorithm is. (The ratio between the bandwidths of two TCP flows is not exactly 1, but depends on their RTT, see for example [14]. The simulations show that for a difference of latency inferior to 10ms (difference of RTT inferior to 20ms), RED has a ratio nearer to 1, hence RED is fairer. However, between 10ms and 100ms, DDRED has a ratio nearer to 1, hence DDRED is fairer. When the latency is greater than 50ms, the fairness of both strategies is over, even if DDRED has a better value than RED. One of the conclusions is that DDRED is not adapted to networks where latencies are small, or "closed networks", but works better than RED for "open networks", where latencies are varied. All the autonomous systems in today's Internet can then be thought of as open networks.

Finally, it is worthwhile to note that DDRED is not only fairer (i.e. the ratio between flows is nearer to 1) than RED, but also has a higher throughput.

VII. CONCLUSIONS AND PERSPECTIVES

During congestion, routers drop packets, independently of the queue policy. When a packet is dropped, all the network resources it has consumed are *wasted*. This article proposes a new algorithm of packet drop on routers which takes into account the path covered by a packet up to these routers. Packets far from their sources are favored compared to packets near their sources. The solution we propose is based on a modification of the IP packet management by the routers during congestion. It is based on an adaptation of the flow priority according to the position of its source in the network.

Simulations in NS2 of this algorithm in RED have been carried on various networks. They show that with the new algorithm the bandwidth of a favored flow does indeed increase compared to other flows. Moreover, as packets from long path flows are favored, we globally save network resources. This leads to higher global bandwidth, hence higher average flow bandwidth, without sacrificing the TCP fairness.

VIII. ACKNOWLEDGEMENTS

The work in this paper is funded by: the European Union, French Ministry of Research, DRIRE, Franche-Comté Council and CAPM.

REFERENCES

[1] J. Postel, "Transmission control protocol," Internet Engineering Task Force, RFC 0793, Sept. 1981.

[2] ——, "Internet Protocol," Internet Engineering Task Force, RFC 0791, Sept. 1981.

[3] S. Floyd and V. Jacobson, "Random early detection gateways for congestion avoidance," *IEEE/ACM Transactions on Networking*, vol. 1, no. 4, pp. 397–413, Aug. 1993.

[4] A. Carzaniga, D. Rosenblum, and A. Wolf, "Content-based addressing and routing: A general model and its application," University of Colorado, Tech. Rep., Jan. 2000.

[5] C. Semeria, *Supporting Differentiated Service Classes: Queue Scheduling Disciplines*, Juniper Networks, Dec. 2001, white paper.

[6] Cisco Systems, "Weighted random early detection on the Cisco 12000 series router," http://www.cisco.com/univercd/cc/td/doc/product/software/ios112/ios112p/gsr/wred_gs.pdf, Mar. 2002.

[7] B. Zeng and M. Atiquzzaman, "DSRED: an active queue management scheme for next generation networks," in *25th Annual IEEE Conference on Local Computer Networks LCN 2000*, 2000, pp. 242–251.

[8] A. Haider, H. Sirisena, and K. Pawlikowski, "Improved congestion control with hybrid RED," in *10th International Conference on Telecommunications ICT 2003*, vol. 2, 2003, pp. 923–928.

[9] A. G. F. Agarwal, A. Jayaraman, and C. K. Siew, "Modified RED gateways under bursty traffic," in *IEEE communications Letters*, vol. 8, no. 5, 2004, pp. 323–325.

[10] S. Blake, D. Blake, M. Carlson, E. Davies, Z. Wang, and W. Weiss, "An architecture for differentiated service," Internet Engineering Task Force, RFC 2475, Dec. 1998.

[11] Z. Zheng and R. Kinicki, "Adaptive explicit congestion notification," in *Seventh International Symposium on Computers and Communications (ISCC'02)*, Italy, July 2002, pp. 855–860.

[12] S. Floyd, "Recommmandation on using the gentle_ variant of RED," http://www.aciri.org/floyd/red/gentle.html, Mar. 2000.

[13] K. G. Ramakrishnan, S. Floyd, and D. L. Black, "The addition of explicit congestion notification (ECN) to IP," Internet Engineering Task Force, RFC 3168, Sept. 2001.

[14] M. Mathis, J. Semke, and J. Mahdavi, "The macroscopic behavior of the TCP congestion avoidance algorithm," *ACM SIGCOMM Computer Communication Review*, vol. 27, no. 3, pp. 67–82, July 1997.