Metrics for Packet Reordering – A Comparative Analysis *

Nischal M. Piratla[1] Anura P. Jayasumana[2]
[1]Deutsche Telekom Laboratories, Ernst-Reuter-Platz 7, D-10587 Berlin, Germany
[2]Department of Electrical and Computer Engineering, Colorado State University, Fort Collins, CO 80523, USA

Abstract – Packet reordering is an inevitable phenomenon on the Internet. An ideal metric for packet reordering should capture reordering accurately, provide insight into nature of reordering, and help in evaluation and analysis of reordering leading to mitigation of its adverse effects. Proposed metrics for packet reordering, namely, Reorder Density, Reorder Buffer-occupancy Density, Reordering Extent, and n-Reordering, overcome to various degrees the deficiencies of percentage reordering in capturing the nature and extent of reordering. These metrics vary widely in areas such as evaluation complexity, measurement technique, usage, and in the definition used for packet reordering itself. Metrics for reordering are evaluated using a framework consisting of a set of both essential and desirable attributes of reorder metrics. The attributes include the ability to capture reordering, sensitivity to lost and duplicate packets in reorder measurements, usefulness, simplicity, and evaluation complexity. Finally, the characterization of packet reordering using these metrics is discussed, using sets of measurements carried over the Internet.

Keywords: *Packet Reordering, Network Measurements, Reorder Metrics, Traffic Monitoring, TCP/IP, Performance Evaluation*

1. INTRODUCTION

Traffic trends over the Internet indicate a significant increase in the number of simultaneous flows and the number of routing entries within routers, calling for architectures with increased parallelism. The growth of complexity of networks, and the introduction of QoS features increase the likelihood of reordering. In addition, ad hoc routing and heterogeneous hand-off technologies in wireless domain introduce additional parallelism. All these point to an increasing trend of packet reordering. Applications based on both UDP and TCP are affected adversely due to reordering. In TCP, reordering leads to unnecessary retransmissions thereby creating a sense of congestion and an effective loss in throughput [1]. Moreover, reverse-path reordering affects the self-clocking property of TCP, resulting in bursty transmissions that may lead to congestion. In delay sensitive applications based on UDP, for example VoIP, an out-of-order packet that arrives after the elapse of playback time is treated as lost, thereby decreasing the perceived quality of voice.

The causes of packet reordering include [2], but are not limited to following:

- Packet reordering inside switches and routers: When an earlier packet is placed in a longer queue and a later packet in a shorter queue, the packets may leave the node out of order. The CPU processing speeds approximately double every 18 months, while network link speeds double every eight months [17]. Routers thus have to increasingly rely on parallel processors and queues, making this phenomenon more and more likely.
- Retransmissions: When a packet is lost, the retransmitted packet may arrive out of sequence. Examples include layer-2 retransmissions on wireless links and retransmissions in TCP.
- Diffserv scheduling: If a flow exceeds the negotiated service level constraints, the non-conformant packets of the flow are either dropped, or given a lower priority and placed in different queues, resulting in out-of-order delivery.
- Load splitting: To balance the load among the multiple paths, different packets of the same stream take different routes leading to different delays causing reordering.

Problems caused by reordering are handled at different levels in TCP/IP suite. TCP allows adjustment of 'dupthresh' parameter, i.e., the number of duplicate ACKs to be allowed before classifying a following non-acknowledged packet as lost [3]. This parameter (known as 'tcp_reordering' in Linux implementations) allows the reordering to occur to a certain extent without affecting the throughput. At application level, the out-of-sequence packets are buffered until they can be played back in sequence. An increase in out-of-order delivery however consumes additional resources and also affects the end-to-end performance. Consequently, certain techniques attempt to reduce reordering at intermediate nodes, i.e., at IP level. Many routers attempt to eliminate the reordering caused by the scheduling schemes within these nodes by either a) input sorting, i.e., identifying the individual streams and forwarding the packets of the same stream to the same queue thus preventing reordering, or b) output re-sequencing, i.e., buffering packets at the output of the router to ensure that the packets belonging to the same stream are released in the same order as their arrival into the router [4]. Network processors have built-in hardware to track flows. While these approaches reduce the reordering that occurs inside a router, they cannot eliminate reordering due to load splitting among multiple paths. Furthermore, the complexity of these approaches will increase significantly as the number of parallel flows over a path increases (due to the need to process and keep information on a large number of flows), and as the ratio of packet time to routing latency decreases.

Packet reordering is a problem that needs to be addressed proactively. Recent studies of packet reordering based on measurements over the Internet validate reordering as an increasingly common phenomenon [5, 6]. It is important to understand the nature of reordering that occurs in networks.

* This research was supported in part by NSF ITR Grant No. 0121546, Ixia University Partners Program and Agilent Technologies.
Nischal.Piratla@telekom.de, Anura.Jayasumana@colostate.edu.

Such information may be used to enhance the ways that the protocols deal with this problem. Measurement and characterization of reordering in packet sequences and models that provide insight into this problem can lead to the development of scalable techniques to deal with reordering and to develop tools that diagnose network problems.

Percentage reordering has often been used to report the amount of packets reordered. However, as explained in Section 2, it has numerous shortcomings. IP Performance Metrics group of IETF has been working on development of metrics to capture the nature and extent of packet reordering, and consistent measurement techniques. Several metrics have been proposed for packet reordering [2, 7, 8]. These metrics face multiple challenges that include spatial and time complexities, real-time evaluation, robustness, etc. Moreover, a packet arriving early can be classified as reordered only if the preceding packets are not lost, and similarly a late arriving packet may not be reordered if there are earlier copies of the same packet. Thus, reorder measurements may involve buffering or recording sequence numbers and latencies for detection of lost and duplicate packets; these affect the spatial and time complexities of the measurement algorithms. In addition, a TCP sequence rollover may falsely indicate certain packets as reordered in a TCP stream, unless the algorithms are robust to such peculiarities. Proposed metrics vary widely in their definitions of reordering, applicability, usefulness and complexities

This paper reviews the metrics for packet reordering, and compares them based on a framework consisting of a set of essential and desired attributes. The essential attributes such as not treating a duplicate packet as a reordered packet and not treating the packets following a lost packet as reordered, are vital for the validity of metrics. A framework for metrics for the Internet [9] states: "The metrics must aid users and providers in understanding the performance they experience or provide." A metric for packet reordering has to go beyond reporting a mere measure that varies with reordering, to a measure that is useful to characterize reordering. Desirable attributes would make a metric more relevant to Internet applications and protocols. These include but are not limited to simplicity, computation complexity, memory requirement for computation, and robustness of the measures. This discussion naturally leads to a comparison of all the metrics.

The remainder of the paper is organized as follows: Section 2 discusses the proposed packet reordering metrics. In Section 3, the essential and desirable attributes of a reorder metric are identified and a comparison of different metrics is carried out using these attributes. Section 4 presents the measurements of packet reordering over the Internet using the described metrics. Conclusions are in Section 5.

2. PACKET REORDERING METRICS

Percentage of reordered packets has often been used as a measure to evaluate reordering [4, 10, 11, 12]. However, its definition is vague, and no uniform definition exists. Consider two packet sequences (1, 2, 4, 5, 3) and (1, 2, 5, 4, 3). It is possible to interpret the out-of-orderliness of the packets differently: Ex. a) packets 3, 4 and 5 are out-of-order in both cases, or b) only packet 3 is out-of-order in the first sequence, while packets 4 and 5 are out-of-order in the second, c) packets 3 and 4 are out-of-order in both cases, etc. [13]. Need for a precise definition is evident from this example. Even if a convention such as 'only the late packets are out-of-order' is agreed upon, the measure still remains vague as it fails to capture the displacement of an out-of-order packet.

The amount of displacement of a packet directly influences the complexity of hardware/software to recover from reordering. In the sequence (1, 3, 4, 2, 5), if buffers are available to store packets 3 and 4 while waiting for packet 2, it is possible to recover from the reordering. However, there may be cases where the application requirement is such that arrival of packet 2 after this delay renders it useless. While one can argue that a good packet reordering measurement scheme should capture such effects, a counter argument can also be made that packet reordering should be measured strictly with respect to the order of delivery and be not application dependent.

Next we present several metrics that have been proposed for measurement of reordering.

2.1 Reorder Density

Reorder Density (RD) captures both the amount and extent of reordering of packets in an arrival sequence using a discrete density that represents the fractions of the sequence size with respect to packet displacements [2, 7].

Without loss of generality, consider a sequence of packets (1, 2,...N) transmitted over a network. A receive_index (1, 2…) is assigned to each non-duplicate packet as it arrives at the point of measurement, which we refer to as the destination. Lost packets are detected as described later, and the receive_index value skips the sequence numbers of these lost packets. For an arrival sequence in which no losses or duplication of packets occur in the network, in the absence of reordering, the sequence number of the packet and the receive_index values are the same for each packet. If the receive_index assigned to packet *m* is $(m + d_m)$, with $d_m \neq 0$, then a 'reorder event' has occurred, and this event is denoted by $r(m, d_m)$. Packet *m* is late if this offset $d_m > 0$, early if $d_m < 0$, and in order if $d_m = 0$. Thus, packet reordering in a sequence of packets is completely represented by the union of reorder events, R, referred to as the 'reorder set':

$$R = \bigcup_m \{r(m, d_m) \mid d_m \neq 0\} \quad (1)$$

RD is defined as the histogram of d_m values, normalized with respect to the total number of packets, which has been adjusted for losses and duplicates. Fig. 1(a) illustrates the sequence, assigned receive_index values, and displacements, as well as the corresponding reorder set and RD for a sequence. In this example, packets 5 and 6 are displaced by one unit from their positions, and packet 7 is early by two positions. Thus, we have a density component 2/8 at one 1/8 at –2 and 5/8 at zero.

When to consider a packet as lost, so that receive_index values can be assigned correctly, is a vital question in a reorder measurement process. If the sequence is small, one can wait till the arrival process is complete to assign receive_index values. However, certain network monitoring scenarios may require reordering to be measured for long sequences of packets. Another possibility would be to consider a packet as lost if it does not arrive when it is expected, but if it arrives later, go back to the previous arrivals and make appropriate corrections. However this approach requires keeping track of all received packets, as well as applying corrections to computations performed so far. Worst case memory consumption with both these approaches is O(N), and real-time evaluation of the metrics is not possible either. Similarly, to declare a packet as a duplicate, its sequence number has to be compared against those of received packets, which requires keeping track of the packets that have arrived earlier, or alternatively those that have not arrived. RD uses a threshold, D_T to declare a packet as lost. In case of an application, such as VoIP, a packet that gets reordered beyond a certain displacement is useless, i.e., as good as lost. Similarly, in a TCP stream if a packet is reordered beyond certain displacement then there is a retransmission with the assumption that the packet is lost. Thus, the application and/or the transport protocol constraints often define a displacement threshold (D_T) beyond which a packet can be considered lost. The same threshold D_T is used to maintain an early-arrival buffer to allow identification of duplicates. If a packet is not received within D_T arrivals from where it is expected, it is considered lost. Real-time RD evaluation may be done in one of two ways:

1. Go-back D_T: In this method, the rules are applied at each arrival. If a packet that was supposed to arrive D_T places ago does not arrive, then this sequence number is removed from the receive_index, and RD is recomputed for the previous D_T steps. Consider a received sequence (1, 2, 4, 5, 6, 7, 8, 3) and D_T = 3. As soon as packet 5 arrives, 3 is classified as lost and we go back and correct the previous D_T receive_index values and displacements. When packet 3 actually arrives later, we do not assign receive_index to this arrival, i.e., consider it as lost and discard the packet. This method requires recording only the previous D_T packet numbers, and involves additional processing to re-compute offsets when a packet is lost.
2. Stay-back D_T: Here the computation lags an arrival by D_T packets, i.e., the packet with receive_index *i* is not used in the evaluation until D_T more packets have arrived after that. Thus, we do not correct or adjust any displacements, rather wait until a missing packet can be declared lost. This method requires buffering of the next D_T arrivals.

A packet is classified, as a duplicate on its arrival, if its sequence number exists in early-arrival buffer or is less than the current receive_index.

2.2 Reorder Buffer-occupancy Density

Reorder Buffer-occupancy Density (RBD), is the normalized form of a histogram of the occupancy of a hypothetical buffer used for the recovery from out-of-order arrival of packets [7]. When a packet arrives with a sequence number greater than that expected, it is considered to be stored in a hypothetical buffer until it can be released in order. Buffer occupancy is tracked at each arrival instant, assuming one buffer for each packet. This concept can easily be extended to keep track of occupancy in bytes as well. The occupancy density of this buffer, RBD is a measure of reordering. For the sequence (1, 2, 3, 4, 7, 5, 6, 8), shown in Fig. 1(b), the buffer-occupancy when the packet with the sequence number 7 arrives is 1 because it arrived when 5 was being expected. The buffer occupancy remains 1, when the packet 5 arrives.

Arrived Sequence	1	2	3	4	7	5	6	8
Receive_index	1	2	3	4	5	6	7	8
Displacement	0	0	0	0	-2	1	1	0

R = {(5, 1), (6, 1), (7, -2)}

RD[-2] = 1/8, RD[0] = 5/8 , RD[1] = 2/8

(a) RD

s[i]	1	2	3	4	7	5	6	8
i	1	2	3	4	5	6	7	8
Extent ‘e’	-	-	-	-	-	1	2	-

Number of packets with extent 1 = 1

Number of packets with extent 2 = 1

(c) Reorder Extent

Arrived Sequence	1	2	3	4	7	5	6	8
Expected	1	2	3	4	5	5	6	8
Buffer-occupancy	0	0	0	0	1	1	0	0

RBD[0] = 6/8, RBD[1] = 2/8

(b) RBD

s[i]	1	2	3	4	7	5	6	8
i	1	2	3	4	5	6	7	8
n-reordering	0	0	0	0	0	1	0	0

Degree of 1-reordering = 1/8

(d) n-reordering

Figure 1. Example of reordered sequence (1, 2, 3, 4, 7, 5, 6, 8) with corresponding (a) RD, (b) RBD, (c) Reordering Extent and (d) n-Reordering

When packet 6 arrives the contents of the buffer can be used in order along with 7, so the buffer occupancy becomes zero. Thus, for two arrivals, i.e., 7 and 5 the buffer-occupancy is one, and for all other arrivals it is zero. Therefore, RBD for this sequence is given as: RBD[0] = 6/8, RBD[1] = 2/8.

As an arrival may be rendered useless due to an application constraint or a transport layer constraint, a threshold on buffer-occupancy B_T is used. If an out-of-order packet needs to be stored in the hypothetical buffer that is already filled to B_T, the currently expected packet is considered to be delayed more than the tolerance and hence, is deemed lost. Reordering ByteOffset metric in [8] is similar to RBD in [13], with buffer occupancy in bytes instead of packets.

To detect duplicates in RBD computation, the sequence number of the arrived packet is compared with expected sequence number and the contents of the buffer. If a packet has a sequence number lower than the expected or has same sequence number as one of the packets in the buffer, then it is classified as a duplicate.

Algorithms and Perl scripts and Java Applets for RD and RBD approaches are provided in [14].

2.3 Reordering Extent

Reordering Extent and n-Reordering [8] are lateness-based metrics, i.e., a packet is not considered reordered unless it is late.

Reordering extent is the maximum distance, in packets, from a late packet to the earliest packet received that has a larger sequence number. If a packet is in-order, its reordering extent is undefined. The first packet to arrive is in-order by definition, and has an undefined reordering extent. Consider a stream of packets (1, 2,.., N), and let N' be the total number of packets received out of the N packets sent at source. Reordering extent assumes that duplicates are removed, so N' ≤ N, with the difference corresponding to losses. Let s[1], s[2],.. s[N'] represent the original sequence numbers associated with the packets in the order of arrival. Consider a reordered packet with arrival index i and source sequence number s[i]. Note that values of i is equivalent to the receive_index values defined in RD. There exists a set of indices j ($1 \le j < i$) such that s[j] > s[i]. The reordering extent, e, of packet s[i] is defined to be (i-j) for the smallest value of j such that s[j] > s[i].

Fig 1(c) illustrates reordering extent for arrival sequence (1, 2, 3, 4, 7, 5, 6, 8). Consider the arrival of the packet with sequence number 5, i.e., s[i] = 5 and i = 6. The value of j for which s[j] > s[i] is 5 as (s[5] = 7) and (s[6] = 5). Therefore, the extent e of reordering for this arrival = i-j = 6-5 = 1. Considering the arrival of the packet with sequence number 6, i.e., s[i] = 6 and i = 7. The value of j for which s[j] > s[i] is 5 as (s[5] = 7) and (s[7] = 6). Therefore, the extent e of reordering for this arrival = i-j = 7-5 = 2. For other packet, the extent is undefined.

For both the arrivals considered above there are no values of j less than 5, such that s[j] > s[i]. The sequence number s[j] for the lowest value of j is termed as the point of discontinuity. Thus, for both the arrivals the point of discontinuity is same and is equal to 7. Note that no reordering extent is defined for packet s[i] =7, as the metric implicitly assumes the early packets are not reordered. Moreover, the frequencies of the extents in a received sequence could be used to obtain a histogram of reordering extent e, though such a representation is not a part of the metric's definition, we observe that it is useful to interpret reordering.

2.4 n-Reordering

n-Reordering defines the extent as the maximum distance, in packets, from reordered packet to the earliest packet received that has a larger sequence number. However, unlike reordering extent, the definition of reordering is skewed. For example, if packets are late in sets, i.e., two or more consecutive packets are late but maintain their positions with respect to each other then only the first packet is considered as reordered [8].

Using the same conventions as in reordering extent, a packet number i, with source sequence number s[i], is n-reordered ($n < i \le N$) iff s[j] > s[i] for all j such that i-$n \le j < i$. The degree of n-reordering of the sample is m/N', where m is the number of n-reordered packets in the sample and N' is the size of the arrival sequence after discarding duplicate packets. A sample is said to have no reordering if its degree of n-reordering is 0 for all n ($1 \le n < N'$). Fig. 1(d) illustrates n-reordering for sequence (1, 2, 3, 4, 7, 5, 6, 8). Consider the received packet number i = 7, and its source sequence number s[7] = 6. As per the definition, a packet is late if one or a consecutive set of its immediate preceding packets have higher sequence numbers. However, in the case of packet number 6, since its arrival occurs right after packet number 5, which is a lower sequence number, it has no 'n' parameter associated with it.

3. ATTRIBUTES OF REORDER METRICS

Previous Section presented different metrics for measurement of packet reordering. RD takes into account the earliness and lateness of packets in a sequence. RBD evaluates reordering from the point of view of resources required to recover from reordering, while reordering extent and n-reordering identify only the late packets as reordered packets. In this section, we discuss three essential attributes and four desirable attributes for reorder metrics.

3.1 Essential Attributes

3.1.1 Capture Reordering

The first and foremost requirement of a packet reorder metric is its ability to capture the amount and extent of reordering in a sequence of packets. Fulfillment of other attributes is of no use if this property is not satisfied. Consider the sequence (1, 15, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14). If a metric is based only on early arrivals, then packet 15 is identified as early, i.e., out of order. However, a lateness-based metric would indicate all the packets from 2 to 14 to be out of order, even though it is more likely that a single packet arrived early. A metric based only on earliness or only on lateness captures only a part of information associated with

reordering. A metric capturing both early and late arrivals in contrast provides a complete picture of reordering in a sequence. n-Reordering captures only a part of late arrival based reordering, i.e., it treats a subset of late arrivals as out of order. For example, in the sequence illustrated in Fig. 1(d), packet 6 is considered in-order though it came after packet 7 in the sequence.

RD captures both late and early packets along with their extent. As observed in Fig. 1(a), the early packet 7 and late packets 5 and 6 have non-zero displacements associated with the arrivals. Thus, RD satisfies this fundamental requirement to full extent. In the case of RBD, the early packets are buffered, until they can be used, which happens when late packets arrive. Non-zero occupancy components of these metrics represent early arrivals. However, the zero-occupancy component may contain both in-order and late arrivals. Thus, metrics RBD and Reordering Extent only partially satisfy the reorder capture requirement.

3.1.2 Low sensitivity to packet loss and duplication

A reorder metric must treat only an out-of-order packet as reordered, i.e., if a packet is lost during transit then this must not result in the following packets that arrive in order to be classified as out of order. Consider the sequence (1, 3, 4, 5, 6). If packet 2 has been lost, the sequence should not be considered to contain any out-of-order packets. Similarly, if multiple copies of a packet (duplicates) are delivered, this must not result in a packet being classified as out of order, as long as one copy arrived in the proper position. For example, sequence (1, 2, 3, 2, 4, 5) has no reordering. The lost and duplicate packet counts may be tracked using metrics specifically to measure those, e.g., percentage of lost packets, and percentage of duplicate packets. In RD, a packet is declared lost if it does not arrive within a threshold. In RBD, a packet is considered lost if the number of buffered packets exceeds the buffer threshold. The use of a threshold allows these metrics to tradeoff the accuracy and complexity.

Other metrics that do not rely on a threshold assume either that a packet is lost (by default) until it arrives, or that the packet is not lost until the sequence is complete, and these metrics expect a sequence with duplicates removed as input for the computation of reorder measure.

3.1.3 Usefulness

Rather than being a mere value or a set of values that changes with the reordering of packets in a stream, a reorder metric should provide useful information, that can serve some purpose. An application or a transport protocol implementation, for example, may be able to use the reordering information to allocate resources to recover from reordering. A network diagnosis tool may use the metric to rule in or rule out certain causes of reordering. In this paper, we characterize the usefulness with respect to three features:

- Usefulness for TCP Flow Control: For example, lateness displacements from RD may be used to adjust the "dupthresh" parameter in TCP. Conversely, if "dupthresh" value is fixed then RD can provide an upper and a lower bound to the number of unnecessary TCP retransmits [5]. This is illustrated in the following example: Consider Fast Retransmit definition from RFC 2581 and delayed ACK criterion from RFC 1122. Assume no packet duplication within the network. In addition, if a packet is reordered then the following packets in the sequence are not reordered with same or higher displacement. For such cases, a TCP packet is retransmitted if four ACKs are received for the prior segment. This is illustrated in the following example, which also has duplicate ACKs for packet 3 (timeline from left to right for arrivals at the receiver and transmission of ACKs to the sender):

Duplicate ACKs for packet 3						
Arrivals:	1	2	4	5	6	3
ACKs:			3	3	3	3
Receive_index:	1	2	3	4	5	6
Displacement:	0	0	-1	-1	-1	3

Thus, the displacement (according to RD definitions) is greater than 2 for retransmitted packets. Therefore, the fraction of packets (F) that is unnecessarily retransmitted is given as:

$$F = \sum_{k>2} RD[k] \quad (2)$$

From [8], knowledge of n-reordering is particularly useful for determining the portion of reordered packets that can or cannot be restored to order in a typical TCP receiver buffer based on their arrival order alone (and without the aid of retransmissions). For example, if n=3 component exists in an n-reordering measure, a New Reno TCP sender would retransmit 1 packet in response to an instance of 3-reordering and therefore consider this packet lost for the purposes of congestion control (the sender will halve its congestion window). Detecting instances of 3-reordering is useful for determining the portion of reordered packets that are in fact as good as lost.

- Usefulness in Buffer/Resource Allocation in Reorder Recovery: RBD emulates sequencing packets back in order without any exceptions, and thus, may be used to determine the buffer allocation to recover from reordering. Reordering extent 'e' value is suggested in estimations of buffer requirement to recover from reordering, but it fails to do so when there is packet reordering that involves consecutive packets arriving late. For example, in the sequence (1, 2, 3, 4, 7, 5, 6, 8) (also shown in Fig. 1(c)), the buffer requirement before the arrival of 6 is one, but using the extent value the buffer requirement is estimated as two packets. However, from Fig. 1(a), the estimate of buffer size requirement, from RD, is still one for this arrival.
- Usefulness in Network Diagnosis: Anomalous conditions such as errors and faults may cause packet reordering. A metric useful for network diagnosis may be able to rule in or rule out certain causes based on measurements. Use of RD to relate the causes and effects of reordering has been illustrated in [5] under certain scenarios. As an example, if an intermediate node suffers from route fluttering then

RD is shown to have dominant non-zero components for displacements corresponding to the delays of multiple paths taken by packets during fluttering. The periodicity of fluttering may also be related to the component values at these displacements. No such results are yet available with other metrics.

3.2 Desirable Attributes

3.2.1 Simplicity/Informativeness

An ideal metric is the one that is simple to understand and evaluate, yet able to provide a complete picture of reordering. Percentage packets reordered is the simplest metric; But the ambiguity in its definition as discussed earlier, and its failure to carry the extent of reordering make it less informative. On the other hand a record of the complete arrival sequence will contain all the information about reordering; however due to the shear volume of data, may not be of direct use.

RD, being able to capture earliness and lateness, and both amount and extent of reordering, is the most comprehensive metric. It is also simple to understand and easy to evaluate. RBD, the occupancy of a buffer used to recover from reordering, is extremely useful for resource allocation. Both reordering extent and n-reordering are simple to evaluate, given a sequence without duplicates. However, the information carried in reordering extent is just lateness among the packets and is not particularly useful, as it fails to estimate any buffer requirements for de-reordering. One the other hand, n-reordering captures only some of the late packets, as explained earlier.

RD provides information such as mean delay of late packets and percentage of early packets which can be used as singleton metrics for reordering [18]. Reorder Entropy, based on the concept of entropy from information theory using the discrete density distributions can also be used as a singleton metric to characterize disorder in sequence [18]:

$$\text{Reorder Entropy} = (-1)\sum_{i}(\text{RD}[i]\times\log_{e}\text{RD}[i]) \qquad (3)$$

It accounts for both earliness and lateness. In physical terms, the reorder entropy translates to the disorder in the sequence or the amount of surprise in the displacement of a packet.

3.2.2 Low evaluation complexity

Memory and time complexities associated with evaluating a metric play a vital role in implementation and real-time measurements. Spatial/Memory complexity corresponds to the amount of memory or record keeping required for the overall measurement process, whereas time/computation complexity refers to the number of computation steps involved to effectively compute the amount of reordering in a sequence. On-the-fly evaluation of the metric for large streams of packets require the computational complexity to be O(N), where N is the number of packets. This allows the metric to be updated in constant-time as each packet arrives. In the absence of a threshold defining losses or the number of sequence numbers to buffer for detection of duplicates, the worst-case complexity of loss and duplication detection will increase faster than O(N). The rate of increase will depend among other things on the value of N and the implementation of duplicate detection scheme.

RD and RBD have a memory requirement of sizes D_T and B_T respectively. In addition, RD relies on an early-arrival buffer that is also of size D_T [2]. Application and transport requirements place an upper bound on D_T and B_T values. Thus, the buffer requirements for these metrics are always a constant. Apart from buffering packets, RD and RBD compare the current packet's sequence number with receive_index and expected sequence number respectively, leading to a time complexity of O(N) for a sequence, in both cases.

n-Reordering and reordering extent do not have thresholds or in-built schemes to detect duplicates. Thus the buffer requirements have an upper bound of N, as the previously received sequence numbers for each current sequence number have to be tracked. The tracking is required for the detection of point of discontinuity in reordering extent and n-parameter computation for n-reordering. The worst case for N^{th} packet could result in looking up all the previous N-1 arrivals for this tracking. Thus, n-Reordering and reordering extent metrics have a spatial complexity of O(N) and time complexity of $O(N^2)$. Note this complexity is true irrespective of detection of duplicate packets.

3.2.3 Robustness

Reorder measurement should be robust against different network phenomena and peculiarities in measurement or sequence. Peculiarities associated with sequence include such a very late arrival of a duplicate packet, a rogue packet due to undetected error, and sequence number wrap-around. The impact due to an event associated with a single or a small number of packets should have a sense of proportionality on the reorder measure. Consider the arrival sequence: (1, 5430, 2, 3, 4, 5, 6, ..) where packet 5430 appears to be very early; it may be either due to sequence roll over in test streams [7] or some unknown reason. For this case, reordering extent observes e = 1, 2, 3, etc. for all packets until that with sequence 5429, and the measure of reordering changes drastically. One packet out of a large number of packets can have a disproportional effect on the measurement. Thus, n-reordering is not robust against this peculiarity. For n-reordering, there is a change in degree of 1-reordering but due to partial measure of lateness, it can counter such peculiarities.

RBD is also affected by a very early arrival, as it is stored in buffer, until it can be used in order and thus, shows occupancy of at least 1 for all the intervening arrivals. Use of thresholds (on both early and late arrivals) in RD allows it to recover quickly (within D_T in all cases) from these anomalies, thus making RD robust against such peculiarities.

A reorder metric should be robust against phenomena such as bursty losses, and very early/ very late arrival of a packet(s). For example, the measure should continue to meet all the desirable requirements even if there are bursts of losses creating a significant increment in sequence numbers. The assignment of receive_index and thresholds provide such a mechanism for RD and RBD.

3.2.4 Extension to cascaded networks

Given the reordering introduced by two individual networks, the ability to predict the end-to-end reordering for a connection formed by cascading them is an extremely useful property. Even though we list this property under desirable attributes, we believe that this will prove to be an extremely important attribute for measuring and modeling of reordering in complex networks. In the absence of this property, no amount of individual network measurements short of measuring the reordering for the pair of endpoints of interest would be useful for predicting or characterizing end-to-end reordering. In addition, such a property will significantly reduce the number of measurements required to characterize complex networks.

Note that RD corresponding to a sequence of in-order packets is a unit impulse (at $k = 0$). Consider sending this sequence of packets through a network. RD of the sequence observed at the output corresponds to the reordering introduced by the network, and hence is termed the reorder response ($\boldsymbol{J}[k]$) of the network. Reorder response is defined with respect to an input and an output of a network. Thus each input/output port pair of the network may have a corresponding reorder response. Reorder response of a network in which there is no reordering corresponds to a unit impulse at k=0. $\boldsymbol{J}[k]$ can also be interpreted as the probability that a certain packet gets displaced by k. The reorder density $\boldsymbol{J}$[n] of a network formed by cascading two networks with reorder responses $\boldsymbol{J}_1$[n] and $\boldsymbol{J}_2$[n] respectively, has been shown in [2] to be equal to $\boldsymbol{J}_1[n] * \boldsymbol{J}_2[n]$, where * is the convolution operator. This result is valid under stationary network conditions and has been verified using measurements. A metric that does not capture both earliness and lateness cannot capture reordering in cascaded networks accurately, as it is unable to account for reordering caused by earliness in one subnet and lateness in the other. Since metrics other than RD fail to capture earliness of packets, such a relationship cannot be expected of them.

3.2.5 Reordering Models

Analytical models are often helpful in gaining insight into different aspects of networks. Consider for example a scenario of route fluttering, with a packets taking one of several paths with certain probabilities. The relationship of the end-to-end reorder metric to parameters such as probabilities and path delays in this case will prove useful for detecting and diagnosing such scenarios. While it is unrealistic at this point to expect models for complex scenarios, due to relatively short time the problem has been under investigation, a metric may not be considered to be useful in long run if models cannot be derived for at least simple networking scenarios.

Models for different scenarios of reordering using RD metric are derived in [5]. Due to its comprehensive nature, i.e., capture of earliness, lateness and in-order packets, RD accounts for the movement of each packet. This may be the reason why it is the only metric that has successfully modeled packet reordering for different scenarios. Models of RBD where the reordering patterns are limited to a maximum of two intertwined movements at a time are also provided in [5]. No such models are available yet with other metrics. The lack of ability to capture the packet displacement in both directions (early and late) by a metric makes it somewhat harder to deal with modeling complexity by reducing complex scenarios to a combination of simpler scenarios.

Table 1 summarizes the capabilities of the different metrics with respect to essential and desirable attributes. In lateness based percentage reordering, a packet arriving after one or more packets that have higher sequence numbers, is classified as late and is considered reordered. Using an in-built loss detection scheme, such as that in reordering extent, percentage reordering could achieve low sensitivity to losses. As n-reordering and reordering extent do not specify a mechanism for duplicate detection [8], we categorize 'sensitivity to loss and duplicates' as partially present, which is also true for lateness based percentage reordering.

Analysis and simulation based modeling of networks often rely on the ability to reproduce network phenomenon based on models. Regeneration of packet sequences satisfying a given reorder metric thus would play a useful role. Regeneration of sequences satisfying a given RD or RBD measure has been demonstrated in [14]. Cascading metrics for individual subnets to predict reordering in cascades of subnets, and generation of sequences satisfying a given measure, have yet to be demonstrated with other metrics.

4. MEASUREMENTS

Different reorder metrics are used in this section to illustrate packet reordering observed over selected network paths. Multiple files (> 2MB) were downloaded from various servers located at different places around the world to a host on subnet 129.82.x.x (Colorado, USA). The one-way delay value (D) and approximate standard deviation of the delay (SD) were obtained using ping results. Three sets of measurements are presented here, corresponding to a range of end-to-end delays. Servers, where the packet sequences originated were running TCP applications namely, HTTP and FTP at Net-1 in USA (D = 45.5 ms, SD = 0.81 ms), and Net-2 in Italy (D = 81.9 ms, SD = 0.19 ms) respectively. From host on subnet Net-3, which is 130.105.x.x located in New Zealand (D = 95.4 ms, SD = 0.17 ms), a media file was played with an application based on RTP/RTCP. 'tcpdump' was used to collect the information on received data. The number of packets and duplicates on these networks are: Net-1 - 391 packets and 16 duplicates, Net-2 - 6493 packets and 2 duplicates, Net-3 - 5805 packets and no duplicates.

Fig. 2 illustrates the reordering measures using RD, RBD, reordering extent and n-reordering. Reordering extent does not follow the normalization with respect to total number of non-duplicate packets received as other metrics do. However, the normalization is applied for comparing it with other metrics to illustrate the fraction of packets with corresponding extents. In the case of RD and RBD plots, i.e., Fig. 2(a) and 2(b), the vertical axes are in log10 scale to show the reordering components clearly. The threshold values for these metrics are $D_T = 15$ and $B_T = 15$ respectively.

Table 1. Summary of essential and desirable attributes with respect to corresponding metrics

X – the attribute is absent, ◖ - the attribute is partially present, √ - the attribute is present.

Requirement/Metric	*(RD)*	*RBD*	*Lateness based Percentage Reordering*	*Reordering Extent*	*N - reordering*
Capture reordering	√	◖	◖	◖	X
Low sensitivity to loss and duplication	√	◖	◖	◖	◖
Usefulness – TCP Flow Control	√	X	X	X	√
Usefulness – Buffer/Resource Allocation	√	√	X	X	X
Usefulness – Causes and Effects	√	**X**	X	X	X
Simplicity/ Informativeness	√	√	X	◖	◖
Evaluation Complexity- Spatial	**Constant**	**Constant**	**O(N)**	**O(N)**	**O(N)**
Evaluation Complexity- Computation	**O(N)**	**O(N)**	**O(N)/O(N^2)***	**O(N^2)**	**O(N^2)**
Robustness	√	X	X	X	√
Extension to cascaded networks	√	X	X	X	X

* For detection of duplicate packets

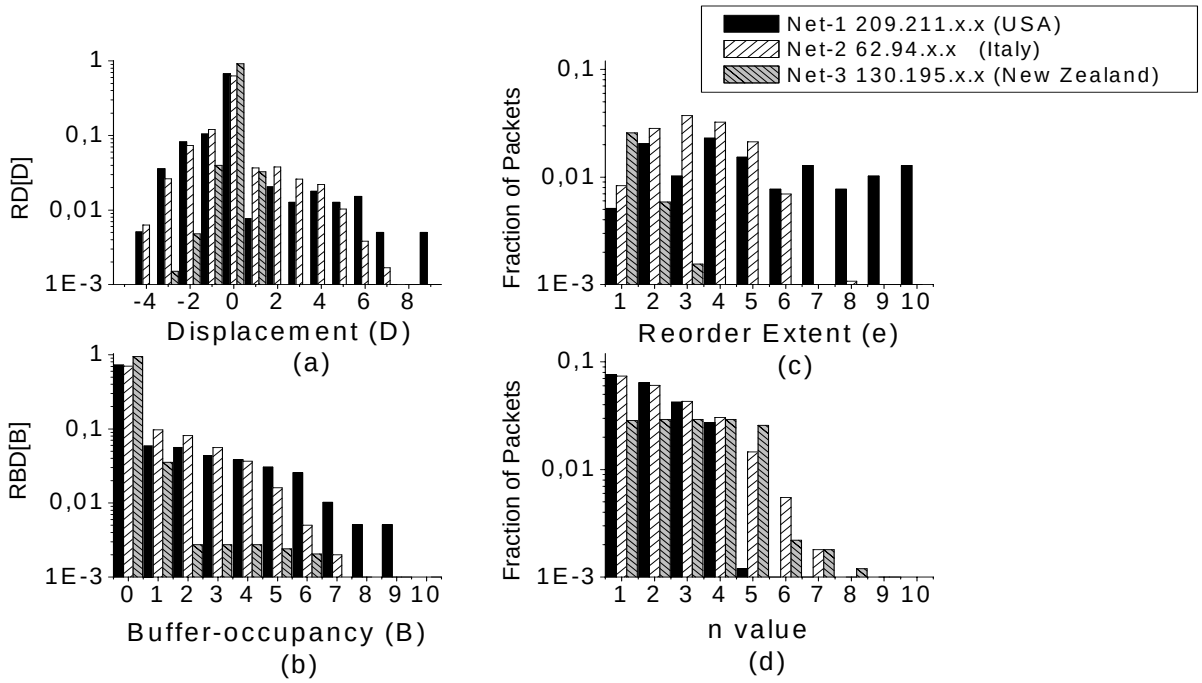

Figure 2. Reorder measurements using different metrics for Net-1, Net-2 and Net-3 (a) RD, (b) RBD, (c) Reordering extent and (d) n-Reordering

As an example, considering measures corresponding to Net-1, it can be observed that reordering extent has values until 10 whereas the buffer required to recover, from RBD or maximum lateness in RD, is nine. This clearly shows the overestimate of buffer requirement using *e* value. RBD also illustrates how often the buffer is filled. With n-reordering measure for Net-1, there is no substantial information that one can derive as the maximum value of 'n' in n-reordering is limited to 4, whereas packets are as late as 9 (from RD).

RD, with RD[0] = 0.63, shows that for Net-2, the number of packets that arrived in their actual positions is lower than that for other Nets (for Net-1 and Net-3 they are 0.68 and 0.92 respectively). The higher reordering can also be derived from the RBD plot where it shows more buffer usage for Net-2 compared to other Nets. Net-3 had the least number of packets out of their actual positions, as shown by RD. Though, there are few packets with displacements as high as 8, the fraction of such packets is less. Additional measurements on packet reordering over Internet, such as daily and weekly trends can be found in [18].

5. CONCLUSIONS

Packet reordering will become more prevalent as the parallelism within routers and switches increases to bridge the increasing gap between processing speeds and network link speeds. Increase of number of flows in links will make it much more difficult to track flows to send all the packets of a flow to the same processor, or to release the packets in the same order as they came in. Overlay networking, QoS provisioning, protocols exploiting multi-homing, etc., features that are being increasingly deployed, also contribute to reordering. In the wireless domain, the hand-off of connections among heterogeneous technologies or ad hoc routing encourages parallelism to achieve successful data transfer. Measurement and characterization of packet reordering is a vital step for dealing with the increase of packet reordering and mitigating its impact on end-to-end performance.

The metrics for reordering, Reorder Density (RD), Reorder Buffer-occupancy Density (RBD), Reordering extent and n-Reordering were compared with respect to relevant attributes. Attributes are classified into (i) essential that must be satisfied by any reorder metric, and (ii) desirable, whose presence in a metric makes it more useful for extensive analysis and characterization of networks. Essential attributes include capturing reordering in a sequence, low sensitivity to packet loss and duplication and metric's usefulness to evaluate behavior and performance of a network, etc. Due to its ability to capture both earliness and lateness of packet reordering, RD fares better than the other metrics in all categories.

Simplicity, informativeness, evaluation complexity, robustness, and extensibility to cascaded networks form a set of desirable attributes. The evaluation complexities of RD and RBD are characterized by constant buffer requirement and computation complexities of O(N), where N is the size of the sequence under consideration. Reordering extent and n-reordering have a buffer requirements of O(N) and computation complexities higher than O(N).

ACKNOWLEDGEMNTS

Authors wish to acknowledge the following for helpful discussions: Abhijit Bare, Bin Ye, Tarun Banka, Rick Whitner, Henk Uijterwaal, Mark Allman, Al Morton and S. Shalunov.

REFERENCES

[1] Laor, M., and Gendel, L., "The Effect of Packet Reordering in a Backbone Link on Application Throughput," *IEEE Network*, Sep./Oct. 2002, 28-36.

[2] Piratla, N. M., Jayasumana, A. P., and Bare, A. A., ""RD: A Formal, Comprehensive Metric for Packet Reordering," *Proceedings Fourth IFIP-TC6 Networking Conference (Networking 2005), LNCS 3462,* May 2005, 78-89.

[3] Zhang, M., Karp, B., Floyd, S. and Peterson, L., "RR-TCP: A Reordering-Robust TCP with DSACK," *Proc. 11th IEEE Int. Conf. Networking Protocols (ICNP' 03)*, Nov. 2003, 95-106.

[4] Liu, H., "A Trace Driven Study of Packet Level Parallelism," *Proc. Int. Conf. Communications (ICC' 02)*, 2002, 2191-2195.

[5] Piratla, N. M., "A Theoretical Foundation, Metrics and Modeling of Packet Reordering and Methodology of Delay Modeling using Inter-packet Gaps," *Ph.D. Dissertation*, Dept. of Electrical and Computer Engineering, Colorado State University, Fall 2005.

[6] Gharai, L., Perkins C., and Lehman, T., "Packet Reordering, High Speed Networks and Transport Protocol Performance", *Proc. IEEE ICCCN*, Oct. 2004, pp. 73-78.

[7] Reorder Density and Reorder Buffer-occupancy Density - Metrics for Packet Reordering Measurements, *IETF Draft*, Last updated Sep. 2005 (work in progress).

[8] Packet Reordering Metric for IPPM, IETF draft, Last updated Jun. 2005 (work in progress).

[9] Paxson, V., Almes, G., Mahdavi, J. and Mathis, M., "Framework for IP Performance Metrics," *IETF RFC 2330*.

[10] Bennett, J. C. R., Partridge, C. and Shectman, N., "Packet Reordering is Not Pathological Network Behavior," *Transactions on Networking IEEE/ACM*, Dec. 1999, 789-798.

[11] Jaiswal, S., Iannaccone, G., Diot, C., Kurose, J. and Towsley, D., "Measurement and Classification of Out-of-sequence Packets in Tier-1 IP Backbone," *Proc. IEEE INFOCOM*, Mar. 2003, 1199-1209.

[12] Bellardo, J. and Savage, S., "Measuring Packet Reordering," *Proceedings Internet Measurement Workshop (IMW'02),* Nov. 2002, 97-105.

[13] Banka, T., Bare, A. A., and Jayasumana, A. P., "Metrics for Degree of Reordering in Packet Sequences," *Proc. 27th IEEE Conf. Local Computer Networks*, Nov. 2002, 333-342.

[14] http://www.cnrl.colostate.edu/Reorder_perl_scripts.html.

[15] Bare, A. A., "Measurement and Analysis of Packet Reordering," *Masters Thesis*, Dept. Computer Science, Colorado State University, 2004.

[16] Piratla, N. M., Jayasumana, A. P., and Banka, T., "On Reorder Density and its Application to Characterization of Packet Reordering," *Proc. 30th IEEE Local Computer Networks Conference*, Nov. 2005, 156-163.

[17] Roberts, L. G., "Beyond Moore's Law: Internet Growth Trends," *Computer*, vol. 33, no. 1, Jan. 2000, 117-119.

[18] Ye, B., Jayasumana, and Piratla, N. M., "On End-to-End Monitoring of Packet Reordering over the Internet," *Proc. International Conference on Networking and Services (ICNS 2006)*, Jul. 2006.

Nischal M. Piratla is a Senior Research Scientist at Deutsche Telekom Laboratories. He is also a co-founder and CTO of Qiro GmbH in Berlin, Germany. He received Ph.D. (2005) and MS (1999) in Electrical Engineering from Colorado State University, and B.Tech (with Distinction) from JNTU College of Engineering, Hyderabad, India. His industry terms in R&D were at Avaya Inc. (2000-2002), Seagate Technology (1999-2000) and CMC (India) Ltd. (1996-1997). His areas of interest include Context-Aware Services, Recommender Systems, HCI and Network Measurements.

Anura Jayasumana is a Professor of Electrical and Computer Engineering, and Computer Science at Colorado State University. He is a member of the NSF Engineering Research Center for Collaborative Adaptive Sensing of Atmosphere (CASA). He received his Ph.D. (1985) and M.S. (1982) in Electrical Engineering from Michigan State University, and B.Sc. in Electronics and Telecommunication Engineering, with first class honors, from University of Sri Lanka, Moratuwa (1978). Awards he has won include award for best student in electrical engineering at University of Sri Lanka, Moratuwa (1978), College of Engineering Outstanding Academic Achievement Awards at Michigan State University (1982,1983), Outstanding Faculty of the Year Award from Mountain States Council of the American Electronics Association (1990), and Engineering Deans Council Award for Academic Excellence from the Colorado State University in 1998. His research interests include high-speed network protocols, sensor networks, optical networks, performance evaluation, and VLSI testing. He is a frequent consultant to industry. Dr. Jayasumana is a member of Phi Kappa Phi.