# Chapter 4

# A Rate-Based Framework for Congestion Control

The previous chapters show that, despite the advances made in terms of congestion control, traditional protocols and router mechanisms are deficient in terms of minimising congestion. Rate-based congestion control techniques may be useful in overcoming some of these problems. In this chapter, I give a brief discussion of the deficiencies in traditional congestion control schemes, and describe possible remedies. Given the remedies proposed, I sketch a framework for congestion control which is constructed on a 'sustainable rate' basis, without use of such paradigms as sliding windows and round-trip timers. I then examine the components of the framework in detail, and look at the parameters inherent in the framework.

## 4.1 Deficiencies and Remedies

### 4.1.1 End-to-End Control Mechanisms

End-to-end congestion control mechanisms do not perform well because they must infer network congestion from indirect evidence such as packet loss and round-trip delay increases. In many cases, these inferences are wrong: packets are lost due to other reasons such as packet corruption. Mechanisms which feed congestion information from routers back to traffic sources seem to work more effectively than purely end-to-end mechanisms.

The 'sliding window' mechanism is often overused. In protocols such as TCP, sliding windows performs error recovery, end-to-end flow control, and a primitive congestion control. The solution to these three problems is thus not orthogonal. Fast changes in window size can lead to bursty source transmissions: This can cause both short-term and long-term congestion in routers and destinations.

Most end-to-end mechanisms require a round-trip timer to detect packet loss: this is a problem in itself. Due to the often high variance in round trip time, the timer is usually set too low or too high, causing unneeded packet retransmission or poor throughput, respectively. Its inaccuracy leads to incorrect end-to-end judgements of current levels of congestion, based on packet loss. A round-trip timer is also computationally expensive to implement.

To avoid these shortcomings, transport protocols should ensure that error recovery, flow control and congestion control are completely orthogonal. Sliding windows certainly should not be used for congestion control. True rate control should be used for both flow control and congestion control, and packet admission into the network should be as evenly-spaced as possible. This minimises the burstiness of traffic, and helps to minimise short-term router congestion.

A protocol using true rate control should obtain information on the preferred rate from any congestion control scheme in the underlying Network Layer. Error detection should be performed by Selective Acknowledgments and Selective Retransmission. This retransmits only those packets that have been lost and need retransmission.

Instead of a sliding window to limit in-transit traffic for error recovery, a transport protocol should be allowed to have as much in-transit traffic as it desires. This is limited only by its transmission rate, available memory, and the amount of application data ready to transmit. A round-trip timer is still required, but only when a source exhausts its supply of traffic to transmit and must wait for acknowledgments. Until the protocol runs out of data, it should be transmitting in a rate-controlled regime that ensures good throughput. As the timer's value is not critical, its estimation will not be computationally expensive. Furthermore, it will be rarely used, and will thus cause little overhead.

### 4.1.2 Congestion Detection Schemes

The ultimate aim of any congestion scheme is to reduce the load on the network to that which is sustainable. Simplistic schemes such as Source Quench only inform the source of the existence of congestion within the network: no other information can be deduced.

Protocols such as Dec Bit are an improvement over Source Quench in terms of their detection of congestion and the return of this information to sources. However, again, only the existence of congestion is returned to the sources.

Schemes such as RED, while admirable in attempting to address the problems of end-to-end congestion control, can only indicate congestion by *dropping* packets. Packets should not be discarded unless absolutely necessary.

If transport protocols choose to employ rate control as a congestion control method, a congestion scheme should attempt to determine the ideal rate of flow for each source. If these ideal rates can be returned to all sources, the adjustment of their rates should alleviate any congestion.

### 4.1.3 Router Procedures

The aim of any router is to forward traffic and maximise power, operating at the knee of the power diagram in Figure 2 (pg. 4). Power is maximised when response time is low and throughput is high.

Low response time occurs when network delays are low, such as when router buffer occupancy is low. Throughput is high when packets are not lost and links are fully utilised. Power is maximised when an incoming packet can be immediately retransmitted without being delayed due to buffering. Router buffer occupancies should therefore never exceed one packet.

Congestion mechanisms which depend on high buffer occupancies, such as packet reordering schemes and packet dropping schemes, can only be useful in exceptional circumstances when congestion recovery is required. A good congestion control scheme will keep the network operating at maximum power, with low buffer occupancies.

### 4.1.4 Packet Admission Mechanisms

Bursty packet admission (from a traffic source or from an intermediate router) will make low router buffer occupancies difficult to achieve, and will also cause short-term packet delays due to buffering. Smooth packet admission techniques such as Leaky Bucket [Turner 86] should be used by traffic sources and routers to help reduce short-term congestion [Sidi *et al* 93]. If congestion recovery is required, router packet reordering schemes which smooth traffic admission may also be employed.

## 4.2 A Rate-Based Framework: Design Considerations

The essential thrust of the remedies just given is to obtain detailed congestion information from the network, and to use this to adjust transmission rates which will optimise overall network power. The obvious paradigm for such a congestion control system is to be rate-based.

During the development of my rate-based congestion control framework, I was guided by the following design decisions. These address the deficiencies that I have outlined in existing congestion control schemes.

**Transport congestion control should use congestion information from routers.** Bit-setting techniques help a source identify the presence of congestion, but do not indicate the extent of congestion. Routers should feed back more than one bit of congestion information to transport protocols. Ideally, the routers should provide transport protocols with a *sustainable* rate for each traffic flow which allows optimum throughput without causing network congestion.

**Dropping packets is highly undesirable.** A router which is dropping packets is operating in congestion recovery mode, well past the optimum point of operation [Jain 90]. A congestion control scheme should emphasise congestion avoidance, but also provide congestion recovery. If routers are providing sources with congestion information, it is even more important that packets should not be dropped if at all possible.

**Router queue sizes $>$1 are highly undesirable.** Jain [Jain 90] shows that an average router queue length of 1 packet is the optimum value. Queue lengths higher than 1 cause increasing end-to-end delays and round-trip times. Variable queue lengths also add variance to these times. Increasing queue lengths indicate that the router is congested: more packets are arriving than can be retransmitted. A congestion control scheme should strive to keep average queue lengths at length 1.

**Use of round-trip timers is undesirable.** As we have seen, round-trip time estimation is complex and difficult to get right. The proposed congestion control framework should avoid the use of round-trip timers if at all possible.

**Use of sliding windows for flow control is undesirable.** Window update schemes are complex and difficult to get right. Window updates can also lead to bursty traffic, undesirable both from the point of view of congestion and time-sensitive network traffic. Limited window sizes can also lead to poor utilisation in high-speed networks with large latencies. Therefore, window-based flow control techniques should be avoided.

**Packet retransmission should not be done unnecessarily.** Traditional packet retransmission methods such as Go-Back-N retransmit data that have been successfully received by the destination. Newer retransmission schemes such as Selective Retransmission only retransmit data if it is known to have been lost. It is preferable to retransmit data only if required, as this helps to reduce the overall load on the network.

**Separation of error control and flow control is desirable.** Packet loss should not be associated with congestion or flow control in a transport protocol at all. Packets are lost for many reasons other than congestion. A transport protocol should use packet loss to prompt packet retransmission and nothing else.

**Packet admission and readmission should be smooth.** Bursty traffic (such as is seen during window updates) not only causes short-term congestion but increases end-to-end and round-trip variance. Transport protocols should admit packets into the network as evenly spaced as possible, and routers should attempt to preserve this spacing. Techniques such as Leaky Bucket should be used here, and any packet reordering should be discouraged. In fact, queue reordering is essentially impossible if router queue sizes are kept around one packet.

These design decisions form the requirements for the components in the congestion framework.

## 4.3 Overview of the Framework

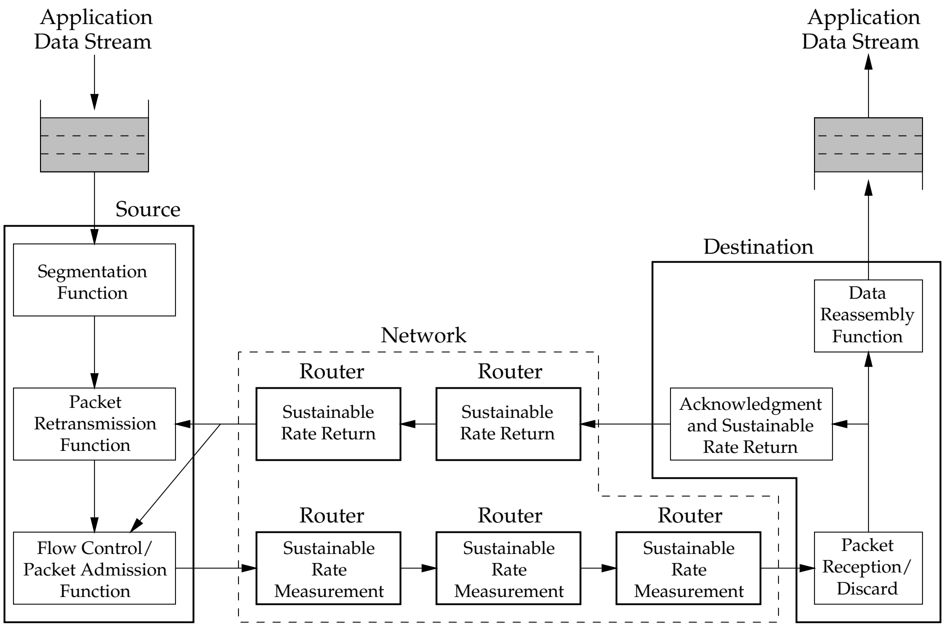


Figure 4: Overview of the Rate-Based Congestion Control Framework

The framework that I propose takes into account all of the requirements just discussed. In essence, the framework spreads the mechanisms of congestion control across both the Transport and Network Layers, measuring and using information on *sustainable rates of traffic flow* through the network. Figure 4 gives an overview of the framework, showing the main components. Routers consist of the functions shown in Figure 1 (pg. 3) but with an extra function, the Sustainable Rate Measurement function.

The essential elements of my proposed rate-based congestion control framework are as follows:

1. The Transport Layer uses a rate-based flow control scheme, using rate information provided by the Network Layer. The flow control and error control mechanisms are orthogonal, being performed by the Flow Control and Packet Retransmission functions, respectively.
2. A source admits packets for each traffic flow (i.e a single data stream from an application) into the network uniformly spaced in time. This minimises short-term congestion due to bursty traffic: this is performed by the Packet Admission function. Routers should attempt to preserve the time spacing of packets in a traffic flow.
3. The routers in the Network Layer implement a congestion control scheme, part of which measures the sustainable traffic rate across the network for each traffic flow; this is performed by the Sustainable Rate Measurement function. This sustainable rate measurement is passed to the destination machine and then via acknowledgment packets to the source machine. The idea of measuring the sustainable rate is the core idea to the framework, and the specific algorithm that I propose is named RBCC[1]. It is described in detail in Chapter 6.
4. Routers implement both congestion avoidance and congestion recovery mechanisms as the rest of the Network Layer congestion control scheme. Routers partition the available resources fairly amongst traffic flows, in both uncongested and congested operating regimes. Routers drop packets when in the operating region of the congestion cliff. These operations are performed by the Packet Selection, Packet Queueing and Packet Dropping functions.

## 4.4 Network Layer Rate Measurement

Before discussing the individual components in the framework, we need to look at the data flows that allow the components to intercommunicate. In the framework, the network routers must measure the sustainable traffic rate across the network for each traffic flow, and return the measurement to the source. This is best achieved with a number of fields in the Network headers of each packet sent from and to the source.

In the direction of flow from source to destination (e.g in data packets), I propose that three fields be used:

**Desired_Rate:** The bit rate which the source of this traffic desires: it can be $\infty$.

**Rate:** The bit rate which has been allocated to this flow of traffic by a router.

**Bottle_Node:** The router which set the value of the **Rate** field in the packet.

When a source transmits a packet, it initialises the **Desired_Rate** field to the bit rate which it desires for the flow of traffic; The **Rate** field is also initialised to this value, and the **Bottle_Node** field is set to the identity of the source. A finite **Desired_Rate** indicates that the source knows in advance the maximum data rate required for its transmission. A **Desired_Rate** of $\infty$ indicates that the source will use as much bandwidth as is made available to it.

[1]Rate-Based Congestion Control.

$$delay = \frac{packet\ size\ in\ bits}{Return\ Rate\ in\ bits\ per\ second} \tag{4.1}$$

Note that if packets are not constant in size, the delay will vary in value, even when the **Return_Rate** value is constant. The delay between packets can be greater than the value determined by Equation 4.1 if there is no available data from the application to transmit. However, once data becomes available, the source must not transmit packets faster than the set rate in order to "make up the average".

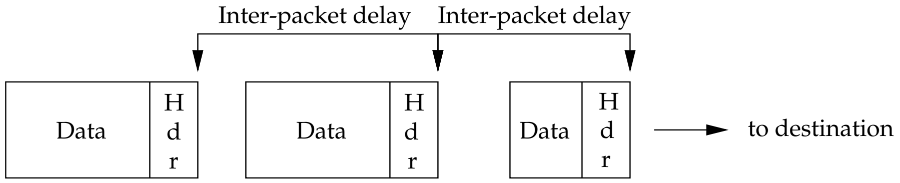


Figure 6: Inter-packet Delay in the Framework

**Return_Rate** values arrive in every acknowledgment packet from the destination. Immediate use of every value is liable to cause short-term rate fluctuations which could result in short-term congestion. To avoid this, the source must leave 1 round-trip gaps between delay changes, in order to allow the **Return_Rate** value used to take effect. Note that sequence numbers can be used to determine that a round-trip has occurred: a round-trip timer or round-trip time estimation is not required.

Although packets are initially admitted at intervals determined by Equation 4.1, this spacing will be destroyed if routers have queue sizes bigger than one packet. Routers should attempt to preserve spacing if possible, and refrain from reordering packets. In the simulations presented in Chapters 8 and 9, First-Come-First-Served queueing was used, with no attempt to preserve packet spacing. Section 10.10 shows that, despite this, packet spacings are preserved.

## 4.6 Source Error Control Function

Any function that provides the error control desired by the transport protocol can be used here, as long as it meets the design decisions. That is, it must be orthogonal to the protocol's flow control, it should not retransmit data unnecessarily, and it should make no reliance on round-trip timers or round-trip time estimation if possible. Chapter 5 discusses a transport protocol which meets these requirements using Selective Retransmission.

It should be noted that, as shown in Figure 4, any packet retransmissions will form part of the uniformly-spaced packet stream exiting the source's Packet Admission Function.

## 4.7 Destination Acknowledgment Function

As outlined in the previous section, a transport protocol should not retransmit data unnecessarily. The destination's Acknowledgment function must be designed with this requirement in mind. Thus, Go-Back-N is ruled out, and Acknowledgment functions such as Selective Acknowledgment should be used. The destination should return acknowledgments to the source as soon as possible, in order to provide timely feedback of the sustainable rate values. Where an acknowledgment acks several data packets, only the *most recent* sustainable rate value should be returned, as this holds the most timely sustainable rate information from the network.

## 4.8 Router Sustainable Rate Measurement Function

The sustainable rate measurement function in each router is the cornerstone of the proposed congestion control framework. A discussion on the requirements and goals of the function, and an example instantiation, is presented in Chapter 6.

## 4.9 Router Packet Queueing Function

The router may use any packet queueing function which meets the design decisions. Queueing functions which attempt to preserve packet spacings for each source may be useful. First-Come-First-Served was used in the simulations described in Chapters 8 to 11.

## 4.10 Router Packet Dropping Function

Again, the router may use any packet dropping function which meets the design decisions. Dropping functions which attempt to preserve packet spacings for each source may be useful. If the rate-based framework keeps the network operating at maximum power, packet loss will occur rarely, and the choice of functions for both Packet Queueing and Packet Dropping will have little effect on the framework. This is borne out in the simulations described in Section 11.8. Drop Tail was used in the simulations described in Chapters 8 to 11.

## 4.11 Router Congestion Avoidance

As noted previously, congestion avoidance schemes attempt to keep the network operating at the knee of Figure 2 (pg. 4). As the main source of congestion information in the framework is the sustainable rate measurement, the router can lower the **Rate** fields in packets further than normal, if it believes it is operating above the knee of the curve. In doing this, it must set itself as the **Bottle_Node**. This extra lowering of the **Rate** field will cause the source to throttle its transmission, and help to bring the router back below the knee of the curve.

## 4.12 Router Congestion Recovery

Although the congestion framework has congestion avoidance, there may be times when a router reaches the point of congestion collapse. For example, a new traffic flow may learn about its rate allocation before other large round-trip flows learn that their allocation has been reduced. For a small period of time, more packets will be admitted into the network than can be transmitted.

The large round-trip flows will eventually learn about their new rates, but only after one round-trip. It should be possible to inform these flows about the problem in less than one round-trip. This control mechanism must also not overburden the network with congestion control information.

I propose as the congestion recovery mechanism a variant of Source Quench. When a router considers itself congested, it returns **Rate Quench** packets to the sources it believes are the cause of the congestion. These packets also have a **Return Rate** field, which the congested router sets to values which will alleviate congestion from each of the congesting sources. Upon receipt of a **Rate Quench** packet with a **Return Rate** field lower than the source's current rate, the source is obliged to immediately lower its rate to the value in the **Rate Quench** packet. **Rate Quench** packets can never raise a source's transmitting rate.

In order to ensure stale **Rate Quench** packets are not used, a source must discard **Rate Quench** packets which have a sequence number less than the last valid **Rate Quench** packet received.

A policy on **Rate Quench** packet generation is discussed in Section 8.4.

One problem with Rate Quench congestion control is that all routers along a flow's path may send Rate Quenches if a flow's rate is lowered. This is an excessive amount of control information admitted into the network. Section 11.3 describes how the use of Rate Quench packets affects the congestion control of the framework.

## 4.13 Summary

The congestion control framework outlined in this chapter is very different from traditional schemes used in connectionless packet-switched networks: sliding windows are not used, round-trip time estimation is not used, Go-Back-N acknowledgment is not used. The bulk of the work in the framework is now performed by the network routers, which are in an ideal position to observe network congestion and take measures to alleviate it.

The Transport Layer is reduced in complexity, with a Leaky Bucket-style flow mechanism controlled by the sustainable rate measurements returned (via acknowledgments) from the routers.

Any transport protocol which meets the requirements described in this chapter may be used in the framework: for example, NETBLT, VMTP and XTP could be used. As much of the functionality of these protocols was not required in the rate-based congestion control framework, I chose to develop a new rate-based transport protocol, TRUMP. The description of this protocol is given in the next chapter.

The sustainable rate measurement function in each router plays a central role in the framework. Its requirements and goals, and a specific algorithm implementing these, RBCC, is discussed in Chapter 6.