6.962: Rate-based Congestion Control and Window-based Congestion Control

Navid Sabbaghi

May 17, 2001

1 Introduction

There are many forms of internet traffic, but for the most part they can be classified as [7]

1. traditional data traffic such as remote terminal (telnet), file transfer (ftp), name service (dns), and electronic mail (smtp). Following Shenker let us call this traffic *elastic* traffic because they can tolerate packet delays and packet losses gracefully in the current "best-effort" admit-all-flows architecture of the internet.

2. "real-time" traffic such as voice and video which are less elastic and hence less tolerant of delay variations and packet losses.

With the increase in traffic, congestion is a growing problem for network applications. Network applications such as streaming audio and streaming video can not live up to their functional specifications in a network where unpredictability in loss (due to buffer overflows at the resources) is rampant. In order to stabilize the network, the internet and its various senders and receivers need a distributed congestion control algorithm.

However just any congestion control algorithm that keeps internet traffic under control will not suffice. Imagine a congestion control policy where traffic is managed but where the policy is unfair to some users (network applications). Such a possibility induces the need for a notion of fairness in any devised distributed congestion control algorithm.

In this summary we will discuss two main types of congestion control algorithms for internet traffic: rate-based congestion control algorithms and window-based congestion control algorithms. We will discuss various attributes of fairness that are and are not present in these algorithms. Also we will discuss the realizability and complexity of these algorithms on top of the current internet infrastructure.

2 Rate-based vs. Window-based

[4] draws a distinction between rate-based congestion control and window-based congestion control algorithms. A rate-based congestion control algorithm directly controls the transmission rate of the connection, based on either feedback from the network [3] or on measurements taken at the end host. Whereas a window-based algorithm adjusts the congestion window size, which is an upper-bound on the number of packets that are "in the pipe", in order to directly control the backlog in the "pipes" and indirectly control the transmission rate.

So if we imagine a network of roads (with stop-lights) in Boston, and we zoom in on the city's popular caterer sending out a sequence of cars (of which it has an unlimited supply) to a particular location, rate-based congestion control would define the rate at which the caterer would send out delivery cars whereas window-based congestion control would define the maximum number of delivery cars that the caterer is willing to have outstanding (en-route/on the roads). Given that there are other cars from other entites on the roads, the aim is to minimize the commute time for all cars (fairness) while making sure the roads are being used to the fullest capacity (efficiency) without having congestion.

3 TCP: a window-based congestion control algorithm

TCP is currently the most widely used congestion control algorithm in use. TCP is a window-based congestion control algorithm. For a particular sender-receiver connection, the sender will break up the message into packets (of possibly varying sizes but let us assume fixed sizes for our purpose) and send those packets individually. For every packet the sender sends, the sender will receive an acknowledgement that is sent by the receiver upon receipt. The sender's window is defined as the number of packets the sender has sent but not received an acknowledgement for (the number of packets in the "pipe"). There are many versions of TCP but the unifying design philosophy is rooted in the idea of achieving network stability at "equilibrium" by obeying a "packet conservation" principle in it's data transport connection: a new packet is not put into the "pipe" until an old packet has been acknowledged and the "pipe" has as many of the connection's packets as it can hold [1]. In order to obtain "packet conservation" three things must happen:

- the dynamics of the connection's window size must reach an equilibrium,
- a sender must inject a new packet after an old packet has exited,
- and the resource limits along the path must not be saturated.

In order to have the dynamics of the connection's window reach an equilibrium the sender gradually increases the number of packets in-transit. The sender starts by having the congestion-window (the packets in-transit) set to one. If it receives an ack for a packet it increases the congestion-window by one packet. And then when it sends again it will send the minimum of the receiver's advertised window and its current congestion-window. If a loss occurs, the congestion window gets set to one. This algorithm is called slow but the window opens quickly enough because to reach a window size of W takes $R\log(W)$, if R is the round-trip-time.

The final two requirements for "packet conservation" require that TCP has an "accurate" round-trip-time estimator. Because in order for TCP to inject a packet into the network for a packet that may have been lost (due to error or queue saturation), TCP needs to wait the approximate round-trip-transit time to be sure the packet actually got lost. In fact, the better versions of TCP estimate both the transit time R and the variation in the transit time [1]. If TCP finds that its window is too large after it has reached equilibrium (due to the dynamics of other senders and receivers), TCP will decrease its window size multiplicatively. Hence the slow-start in conjunction with this multiplicative decrease after equilibrium is why TCP is often called an additive-increase/multiplicative-decrease window-based congestion control algorithm.

TCP can be unfair and inefficient [4]. This is because if route a has a shorter round-trip-time then route b, and the route's share a substantial part of the same physical route, then route a will be able to reach its equilibrium faster then route b. Therefore route a, which may be elastic traffic, may snatch bandwidth from route b, which may be "real-time" traffic. The fact that the Internet is public domain and potentially noncooperative has led to work on pricing models in order to control internet traffic. Therefore for the rest of this summary we will consider rate-based and window-based congestion methods that are grounded in some pricing mechanism. We will forego any analysis of flat rate charging because it has been show that flat-rate charging leads to inefficiencies where a large number of low-usage users end up subsidizing a small number of high-usage users. [4] Instead we will consider usage-based pricing in the rest of this summary.

4 A model for the network

In order to be able to discuss rate-based congestion control algorithms and window-based congestion control algorithms on the same footing, we will compare the two sets of algorithms against the same simplified version of a network model for internet traffic. The network is described via:

- a set J of *resources*, where C_j is the finite capacity of resource j for $j \in J$.
- a *route* route r which is a non-empty subset of J, and let R be the set of possible routes. (note that a user and the path for their network data is represented by r because in this survey we do not deal with the issue of routing although as described later these models generalize to deal with routing)
- a $0-1$ matrix A that describes the allocation of resources to each of the routes. More precisely, $A_{jr} = 1$ if resource j lies on route r (ie $j \in r$), and $A_{jr} = 0$ otherwise.

This particular notation was borrowed from Kelly [3].

5 Pricing: Kelly's rate-based congestion control algorithm

In [3], Kelly is concerned with how available bandwidth within the network should be shared between competing streams of elastic traffic. He uses his mathematical model to conduct stability and fairness analysis of a class of rate control algorithms. Kelly's model directly models rates and not queue lengths. This is one of the reason's his model leads to rate-based congestion control algorithms. He cites work that reasons that queue build-up can be prevented through the bounding of these rates.

In his work Kelly devises a convex program that maximizes an overall utility function for the whole system by picking an optimal set of rates for each route. He splits this convex program into a convex program for the network and for each of the individual users (which he refers to directly as routes). The convex program for the users picks a price that is the user is willing to pay for his/her rate. The convex program for the network maximizes a function that encodes a notion of fairness termed as "proportional fairness" by picking the optimal prices. Kelly reasons that the network's convex program is computationally difficult for any centralized processor and hence he decentralizes a relaxation of the convex program by looking at the solution of a set of differential equations. And he analyzes stability by looking at relevant stochastic perturbations. He also analyzes the dual of relaxation and its set of differential equations. In the dual to the network convex program, the network chooses the best shadow prices of additional capacity at the resources given the users request for rates. These two convex programs lead to the two classes of rate control algorithms. However both classes of rate control algorithms require explicit feedback from the switches in the network. This may or may not be practical for wide deployment in the internet. This will lead us to analyze La's and Anantharam's window-based congestion control that follows up on Kelly's work. But first let us go through the details of Kelly's work.

So given that brief overview let us go through the details. Let us give some of the notation that is specialized for Kelly's model.

- Let x_r be the rate allocated to user (route) r.
- Assume that if user r has rate x_r, then the utility is $U_r(x_r)$ where $U_r(x_r)$ is an increasing, strictly concave and continuously differentiable function of x_r over the range $x_r \geq 0$.

Under this model the optimal rates are the solution to the convex program:

$$SYSTEM(U, A, C):$$

$$max \sum_{r \in R} U_r(x_r)$$

subject to

$$Ax \leq C$$

over

$$x \geq 0.$$

5.1 Why are utilities the quantity to maximize in flow control

How do we evaluate a particular flow structure for a network? Well the Internet was designed to meet user needs and so any such evaluation should take into account how happy the flow structure makes the users and the service requirements of their applications [7]. As an interesting aside Scott Shenker classifies the general classes of utility functions that can be used to describe the performance and resulting value of an application under a network flow structure. The actual utility functions will not be important for Kelly's decentralized algorithm because the only person that will need to know the utility function is the sender of traffic (not any algorithm).

5.2 Decomposing the system optimization problem

The convex program can be decomposed into a user and a network optimization problem. There are two ways to decompose this convex program.

5.2.1 User picks price, Network picks rates

One way to do the decomposition is to have the user demand prices to pay per unit time, p_r, and to receive in return a flow x_r (determined by the network) that is proportional to p_r (say $x_r = \frac{p_r}{\lambda_r}$ where λ_r is the charge per unit flow for user r. This would correspond to the following decomposition:

$$USER_r(U_r; \lambda_r):$$

$$max U_r(\frac{p_r}{\lambda_r}) - p_r$$

over

$$p_r \geq 0.$$

$$NETWORK(A, C, p):$$

$$max \sum_{r \in R} p_r \log x_r$$

subject to

$$Ax \le C$$

over

$$x \ge 0.$$

5.2.2 Fairness

Why does the NETWORK's optimization problem have that particular form? That has to do with fairness. Let us discuss some different notions of fairness that will concern us throughout this summary.

The most commonly described criterion for fairness is max-min fairness. A set of rates is *max-min fair* if no rate may be increased without simultaenously decreasing another rate which is already smaller [3]. So in some sense, max-min fairness gives an absolute priority to users with smaller rates. However, in order to achieve such fairness network efficiency often must be sacrificed. This led Kelly to define a vector of rates $x = (x_r, r \in R)$ to be *proportionally fair* if it is feasible (with respect to his convex program such that $x \ge 0$ and $Ax \le C$), and if for any other feasible vector x^*, the aggregate of proportional changes is zero or negative:

$$\sum_{r \in R} \frac{x_r^* - x_r}{x_r} \le 0.$$

Kelly also defines *rates per unit charge* as being proportionally fair if x is feasible, and if for any other feasible vector x^*

$$\sum_{r \in R} p_r \frac{x_r^* - x_r}{x_r} \le 0.$$

This is simply proportional fairness where each user is able to represent him/herself by p_r identical subusers.

5.2.3 Interesting results about the decomposition

It turns out that the vector of rates that solve NETWORK(A,C;p) are rates per unit charge that are proportionally fair. Furthermore it turns out that the a price vector and rate vector can be found to optimize both SYSTEM(U,A,C) and the decomposition of SYSTEM(U,A,C) into the distributed problems of NETWORK(A,C;p) and $USER_r(U_r; \lambda_r), r \in R$. Also note that the utility function $U_r(x_r)$ is not required by the network, only the user needs to know his/her utility function.

5.2.4 The Dual: User picks rate, Network picks price

Now we can switch the roles that users and networks have by looking at the Lagrangian for the NETWORK(A,C;p) problem:

$$L(x, z; \mu) = \sum_{r \in R} p_r \log x_r + \mu^T (C - Ax - z)$$

where $z \ge 0$ is a vector of slack variables and μ is a vector of Lagrange multipliers (shadow prices). Then we have

$$\frac{\partial L}{\partial x_r} = \frac{p_r}{x_r} - \sum_{j \in r} u_j$$

which leads to the optimum for the primal problem:

$$x_r = \frac{p_r}{\sum_{j \in r} u_j}.$$

The shadow prices for the capacity is the network will now choose given p_r and hence implicity x_r from the optimality relation we just derived. Hence the dual for the network problem becomes:

$$DUAL(A, C; p):$$

$$max \sum_{r \in R} p_r log(\sum_{j \in J} u_j) - \sum_{j \in J} u_j C_j$$

over

$$\mu \geq 0.$$

5.3 Decentralized algorithms that correspond to solutions of relaxation

We can find the solution to a relaxation of the primal problem by looking for the stable point of the system of differential equations:

$$\frac{d}{dt} x_r(t) = \kappa(p_r - x_r(t) \sum_{j \in r} u_j(t))$$

where

$$u_j(t) = p_j(\sum_{s: j \in s} x_s(t))$$

Notice that this algorithm resembles the steady-increase and multiplicative-decrease of TCP.

It turns out under mild regularity conditions on the functions p_j, $j \in J$, there is a Lyapunov function for the system of differential equations and hence a stable point to which all trajectories converge. And the functions p_j may be chosen so that the maximization of the Lyapunov function comes arbitraily close to approximating the optimization problem NETWORK(A,C;p).

Also we can find the solution to a relaxation of the dual problem by looking for the stable point of the system of differential equations:

$$\frac{d}{dt} u_j(t) = \kappa(\sum_{r: j \in r} x_r(t) - q_j(u_j(t))))$$

where

$$x_r(t) = \frac{p_r}{\sum_{k \in r} u_k(t)}$$

In this algorithm users pick their rate and networks pick their price.

Kelly goes ahead and analyzes stochastic perturbations and time lags for this primal and dual algorithm. Notice in the above that small feedback messages need to be sent from the network to the user or vice versa in order to reach an equilibrium eventually. As we mentioned before this may or may not be practical for wide deployment in the internet.

6 Pricing: Anatharam's window-based congestion control algorithm

La and Anantharam argue that the Internet needs a window-based congestion control algorithm over a rate-based algorithm. One of the main reasons for their claim is in regards to the stabililty of the network. If a user uses a rate-based algorithm and they have incorrect estimates of the available bandwidth, then they may release packets into the network at a much higher rate than they should. The network could then suffer from packet loss due to overflowing buffers. However, a window-based algorithm would not only control the rate but it would also control the maximum number of outstanding packets which seems important for network stability purposes. Since their algorithm is window-based they do explicitly model the queues in the network.

There are different versions of TCP that can be classified as loss-based TCP (such as TCP Tahoe and TCP Reno) and queue-based TCP (such TCP Vegas and [6]). Loss-based TCP uses packet losses as an indication of congestion in the network while queue-based TCP uses the estimated queue size to adjust the congestion window size [4]. Loss-based TCP exhibits delay bias and a high packet loss rate whereas Mo's and Walrand's queue-based TCP have illustrated the existence of a fair window-based end-to-end congestion control algorithm that updates the congestion window size in a smarter fashion. Anatharam takes the fairer versions of TCP that Walrand has demonstrated and considers the problem of maximizing the aggregate utility of users while maintaining that fairness.

6.1 The network model

The network model is represented by the following fluid model:

$$Ax - C \leq 0$$

which represents the capacity constraint,

$$Q(Ax - C) = 0$$

which says there is no backlogged queues if the rate does not equal the capacity,

$$X(A^T C' q + d) = w$$

which says the window size of connection i should equal the sum of the amount of fluid in transit and the backlogged fluid in the buffers ($w_i = x_i * \bar{d}_i = q^i$ where q^i denotes connection i's total backlog in the buffers). Window size and rate are related and by adjusting window size we are essentially adjusting the rate.

6.2 A rough overview of the algorithm

For that reason there is a set of differential equations for adjusting the window size that is essentially equivalent to the NETWORK(A,C;p) under different forms of fairness ($(p, 1)$-fairness which is equivalent to Kelly's notion of proportional fairness and (p, α)-fairness which converges to max-min fairness as α tends toward infinity). For lack of time I will refer you to the papers [4], [6].

The pricing scheme is such that the price per unit flow charged to route r is equal to the sum of the queuing delay that user r is reponsible for adding to the network. Since user r can measure this delay he/she can know the price he/she will be charged without feedback from the network. The only thing that remaining is the user adapts his/her window size given the pricing schemes (which are assumed to change slower than the rate at which the user changes his/her window size). This detail can be examined in [4].

7 Conclusion

We have discussed two main pricing methodologies that are meant to relieve internet congestion. Anantharam's method is a window-based methodology that uses the existing TCP infrastructure whereas Kelly's method is a rate-based congestion method. Fairness was a major impetus in the development of these algorithms because TCP currently is a very delay-biased and high packet-loss algorithm.

References

[1] Van Jacobson and M. Karels. Congestion avoidance and control. *Computer Communication Review*, 18(4):314–29, August 1988.

[2] Frank Kelly. Charging and rate control for elastic traffic. *European Transactions on Telecommunications*, 8(1):33–7, January 1997.

[3] Frank Kelly, A. Maulloo, and D. Tan. Rate control for communication networks: shadow prices, proportional fairness and stability. *Journal of the Operational Research Society*, 49(3):237–52, March 1998.

[4] Richard J. La and Venkat Anantharam. Charge-sensitive TCP and rate control in the internet. In *INFOCOM (3)*, pages 1166–1175, 2000.

[5] Steven H. Low and David E. Lapsley. Optimization flow control-i: basic algorithm and convergence. *IEEE/ACM Transactions on Networking*, 7(6):861–74, 1999.

[6] J. Mo and J. Walrand. Fair end-to-end window-based congestion control. Technical Report M98/46, UCB/ERL, Electrical Engineering and Computer Science department, July 1998.

[7] Scott Shenker. Fundamental design issues for the future internet. *IEEE Journal on Selected Areas in Communication*, 13(7), September 1995.