RI01005 May, 2001 Subject: Computer Science

IBM Research Report

Diminishing Weight Scheduling: Algorithms for Game-Theoretic Congestion Control

Rahul Garg *garg@ieee.org*
Abhinav Kamra *abhinav@iitd.ernet.in*
Varun Khurana *varun@iitd.ernet.in*

IBM Research Division,
IBM India Research Lab,
Hauz Khas, New Delhi - 110016. INDIA.
Phone: +91-11-6861100, Fax: +91-11-6861555

IBM Research Division
Almaden - Austin - Beijing - Delhi - Haifa - T.J. Watson - Tokyo - Zurich

Diminishing Weight Scheduling: Algorithms for Game-Theoretic Congestion Control

Rahul Garg* garg@ieee.org
Abhinav Kamra† abhinav@iitd.ernet.in
Varun Khurana‡ varun@iitd.ernet.in

Abstract

Most of the end-to-end congestion control schemes are "voluntary" in nature and critically depend on end-user cooperation. Using a game-theoretic approach we show that, in the presence of selfish users, all such schemes may lead to a congestion collapse.

In this paper we propose a class of switch scheduling algorithms called the Diminishing Weight Schedulers (DWS) that have many desirable game-theoretic properties. With DWS scheduling, max-min fair allocation turns out to be a Nash and Stackelberg Equilibrium. Thus, the "best selfish end-user behaviour" is to estimate the max-min fair rate and send traffic at that rate. DWS solves the problems of excessive congestion due to unresponsive flows and TCP incompatible flows. Different weight functions may be chosen for different reward-penalty profiles.

Keywords: Game Theory, Nash Equilibrium, Stackelberg Equilibrium, Generalized Processor Sharing, GPS, Scheduling, Congestion Control, TCP, Fairness, RIS, DWS.

1 Introduction

Most of the end to end congestion control schemes [29, 3, 20, 39, 32, 17] are voluntary in nature and are critically dependent on end-user cooperation. The TCP congestion control algorithms [29, 3, 5, 24, 25, 23, 10] voluntarily reduce the sending rate upon receiving a congestion signal such as ECN [31], packet loss [16, 11, 22, 33] or source quench [30]. Such congestion control schemes are successful because all the end-users cooperate and volunteer to reduce their sending rates using similar algorithms, upon detection of congestion.

As the Internet grows from a small experimental network to a large commercial network, the assumptions about cooperative end-user behaviour may not remain valid. Factors such as diversity, commercialization and growth may lead to non-cooperative and competitive behaviour [13] that aim to derive better individual utility out of the shared Internet resources.

If an end-user acts selfishly and does not reduce its sending rate upon congestion detection, it could get a better share of the network bandwidth. Such flows are called unresponsive flows [9, 8]. Even responsive flows that react to congestion signal can get unfair share of network bandwidth by being more conservative in reducing their rates and more aggressive in increasing their rates. Such flows are termed as TCP-incompatible flows [9, 8]. This approach of defining TCP-compatible flows has severe limitations as different versions of TCPs give different performance [25] under different conditions. Such behaviours though currently not prevalent, are

*IBM India Research Lab, Hauz Khas, New Delhi - 110016, INDIA
†Indian Institute of Technology, Delhi, Hauz Khas, New Delhi - 110016, INDIA.
‡Indian Institute of Technology, Delhi, Hauz Khas, New Delhi - 110016, INDIA.

present in the Internet, and pose a serious threat to the Internet stability [13, 9]. If widespread, such behaviours may lead to a congestion collapse of the Internet (see Section 2). Therefore it is important to have an approach towards congestion control that is not dependent on cooperative end users voluntarily following the principles of congestion control.

In an earlier work [13], we identified this problem as a manifestation of a well known problem called the Tragedy of Commons [15]. We proposed a game-theoretic approach towards congestion control where cooperation and congestion avoidance becomes the "best selfish end-user behaviour". We proposed a scheduling algorithm called Rate Inverse Scheduling (RIS) which achieves this property. In this paper, we extend the RIS scheduler to a class of scheduling algorithms called Diminishing Weight Scheduling (DWS) with similar proporties. By using different weight functions of DWS, the switch designers and ISPs can choose from a variety of reward-penatly profiles to match their requirements.

Scheduling algorithms such as WFQ [7] only protect well behaved users from misbehaving users, without punishing the misbehaving users. The class of DWS schedulers approach WFQ in the limit as the diminishing weight function becomes flat ($\lim_{\alpha \to 0} : W(r) = 1/(log(r))^{\alpha}$).

There have been discussions on punishing misbehaving users [36] but none of the previous work in this category has talked about punishing the misbehaving flows in such a way that the resulting game-theoretic equilibrium (Nash Equilibrium) results in fair resource allocations. Use of pricing to carry out congestion control has been discussed in seminal works of Kelly [18] and extended in different ways [21, 38]. Game theory has also been used in competitive routing [27] and capacity provisioning [19].

Shenker [34] has proposed a very interesting method to analyze switch service desciplines in a game theoretic framework. Our approach is very similar to Shenker's approach except that instead of discrete queueing theoretic model of input traffic, we use a continuous fluid-flow based input traffic model which is more realistic and amenable to analysis in a network.

In Section 2 we present our game-theoretic formulation and show that in the presence of selfish users, the current resource management policies will lead to a congestion collapse. In Section 3 we present the DWS scheduling algorithm and discuss its properties in Section 4. We present some preliminary simulation results in Section 5. We conclude in Section 6.

2 A Game Theoretic Model of a Network

Consider a link of capacity C shared by N users. There is a sufficiently large shared buffer, a buffer management policy, and a service discipline to partition the link capacity among the users. Assume that user i sends a constant rate traffic flow at a rate r_i (the input rate). Some of this traffic may be dropped due to buffer overflows. Assume that, in steady state the traffic of user i is delivered at the destination with an average output rate γ_i, ($\gamma_i \leq r_i$). The output rate is a function of sending rate of all the N users, the switch service discipline S, and the buffer management policy B. Mathematically, this is written as $\underline{\gamma} = \underline{\gamma}^{SB}(\underline{r})$, where $\underline{r} = (r_1, r_2, \ldots, r_N)$ denotes the N-dimensional vector of input rates and $\underline{\gamma} = (\gamma_1, \gamma_2, \ldots, \gamma_N)$ denotes the N-dimensional vector of output rates and $\underline{\gamma}^{SB}()$ is the function dependent on scheduling discipline S and buffer management policy B mapping the vector of input rates to the vector of output rates.

In general, a user's utility (or satisfaction) depends on its output rate, loss rate, end-to-end delay and delay-jitter. However, for majority of traffic, the utility is primarily dependent on the output rate. For instance, "fire-hose applications" described in [36] are completely loss tolerant. Techniques such as selective retransmissions [4], forward error correction [2], and buffering reduce the dependence on loss rate and delays.

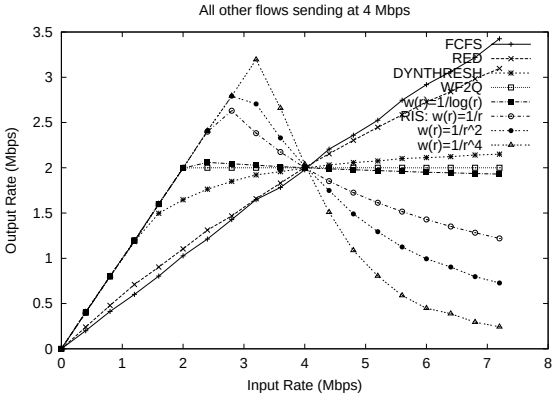

Figure 1: DWS Performance in the presence of CBR flows

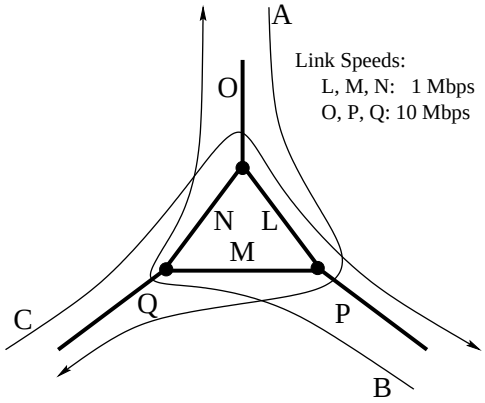

Figure 2: Congestion collapse in a sample network

Assume for simplicity, that a user's utility is an increasing function of its output rate only[1]. If user i was to act in a selfish manner, it would choose a sending rate r_i that maximizes the user's utility (and hence the output rate), irrespective of the amount of inconvenience caused (loss of utility) to the other users. Now, imagine what will happen in such a scenario with different packet service disciplines and buffer management policies.

Consider a link of capacity $C = 10Mbps$ shared by five users sending traffic at constant rate. Figure 1 shows the variation in a user's output rate as a function of its input rate for different scheduling disciplines and buffer management policies. For first come first serve (FCFS) service and drop tail buffer management, the output rate of a user always increases by increasing its input rate. In such a case, there is an incentive for each user to increase its sending (input) rate, irrespective of what other users are doing. If each user was to act selfishly, to maximize its own utility, the link will end up becoming heavily congested with all the users sending traffic at their maximum possible rate and receiving only a tiny fraction of the traffic sent. In game theoretic language it can be said that the only Nash Equilibrium [12] for FCFS is when the input rate approach infinity. Observe that same is true for other buffer management policies such as RED [11] and dynamic threshold [6].

In a network comprising of multiple nodes and links, selfish user behaviour will lead to worse disasters [9, 8] (similar to the congestion collapse [26]) where input rate of each user will approach their maximum possible and output rates will approach zero. To see this, consider the network and flows shown in Figure 2. Assume that FCFS policy is deployed at every link. In the Nash Equilibrium, every user will send traffic at the rate of

[1]Later in Section 4 we also consider loss dependent utility functions

access link, 10 Mbps, and will get a net output rate of less than 100Kbps at the final destination.

Observe from Figure 1 that for policies such as WFQ any profile of input rates more than the fair rate (C/N) constitutes a Nash Equilibrium. Therefore it is fair to say that, although WFQ protects well behaved users, it does not punish misbehaving users. A good policy should not only protect well behaved users, but also punish the misbehaving users. None of the service disciplines and buffer management policies have this property. Note from Figure 1 that the policies in the class of DWS including RIS [13] indeed have this property. In the following sections we formally define the DWS scheduling and its important properties.

3 Diminishing Weight Schedulers

DWS is defined for the idealized fluid-flow traffic model. It is a generalization of Rate Inverse Scheduling (RIS) [13] which is derived from the Generalized Processor Sharing (GPS) scheduling algorithm [28]. Consider a link of capacity C shared by N users sending traffic as N distinct flows. Let $A_i(t)$ represent the amount of traffic of flow i entering the scheduler buffer in the interval $(0,t]$ and $S_i(t)$ be the amount of traffic of flow i served by the scheduler in the interval $(0,t]$. Define $A_i(0) = 0$ and $S_i(0) = 0$ for all i. Define the backlog of flow i at time $t > 0$ as $B_i(t) = A_i(t) - S_i(t)$. Define the total system backlog at time t as $B(t) = \sum_{i=1}^{N} B_i(t)$. Define the input rate of a flow at the link at time t as $r_i(t) = dA_i(t)/dt$ and define the output rate of flow i at the link at time t as $\gamma_i(t) = dS_i(t)/dt$.

A GPS scheduler [28] on the link is defined as the unique scheduler satisfying the following properties:

Flow Conservation:

$$B_i(t) \geq 0, \forall i, t \geq 0. \tag{1}$$

Work Conservation:

$$\text{If } B(t) > 0, \text{ then } \sum_{i=1}^{N} \gamma_i(t) = C. \tag{2}$$

GPS Fairness:

$$B_i(t) > 0 \Rightarrow \forall j : \frac{\gamma_i(t)}{\phi_i} \geq \frac{\gamma_j(t)}{\phi_j}, \tag{3}$$

where ϕ_i is the GPS weight assigned to flow i.

RIS [13] like GPS satisfies the flow conservation and work conservation properties. However, unlike GPS which assigns each flow a constant weight, in RIS each bit gets a GPS weight that is inversely proportional to that bit's arrival (input) rate. If the bit at the head of the queue of flow i at time t arrived at time $\tau_i(t)$, then in RIS $\phi_i(t) = \theta_i / r_i(\tau_i(t))$, where θ_i is the RIS weight of flow i. The RIS fairness property can be mathematically stated as:

RIS Fairness:

$$\begin{aligned} &B_i(t) > 0 \Rightarrow \\ &\qquad \forall j : \quad \tfrac{1}{\theta_i}\gamma_i(t) r_i(\tau_i(t)) \geq \tfrac{1}{\theta_j}\gamma_j(t) r_j(\tau_j(t)). \end{aligned}$$

Such a scheme allows RIS to penalize flows sending at higher rates by decreasing the GPS weights.

It is often desirable to assign different penalties to flows in different circumstances. We introduce DWS, a generalization of RIS, where the GPS weights of flows decrease with increasing input rates according to a

generic weight function $W()$. RIS is a particular instance of the DWS scheduler where $W(r) = 1/r$. In DWS, $\phi_i(t) = \theta_i.W(r_i(\tau_i(t)))$, where $W(r)$ is a continuous and strictly decreasing function of the input rate r. Note that $W()$ is the same for all flows arriving at a particular DWS scheduler. The class of DWS schedulers can be formally defined as satisfying the following properties:

Flow Conservation:

$$B_i(t) \geq 0, \forall i, t \geq 0. \tag{4}$$

Work Conservation:

$$\text{If } B(t) > 0, \text{ then } \sum_{i=1}^{N} \gamma_i(t) = C. \tag{5}$$

DWS Fairness:

$$B_i(t) > 0 \Rightarrow \forall j : \frac{\gamma_i(t)}{\theta_i.W(r_i(\tau_i(t)))} \geq \frac{\gamma_j(t)}{\theta_j.W(r_j(\tau_j(t)))}, \tag{6}$$

where θ_i is the DWS weight for flow i. Thus, DWS punishes flows with large rates by assigning them small GPS weights. The amount of punishment depends upon the steepness of the diminishing weight function. If it is flat such as the $1/log(r)$ function, then it resembles the PGPS like scheduling. If the diminishing weight function is steep then strict penalties are enforced to misbehaving users.

Assume, for simplicity that all the DWS weights are set to 1 and all the users send traffic at a constant rate. Now, all the output rates will also be constant. From the flow conservation property it follows that $\gamma_i \leq r_i$. The DWS fairness condition can be simplified as follows:

$$B_i(t) > 0 \Rightarrow \forall j : \gamma_i/W(r_i) \geq \gamma_j/W(r_j) \tag{7}$$

It follows that if two flows i and j are backlogged, then

$$\gamma_i/W(r_i) = \gamma_j/W(r_j) = k \text{ (constant)} \tag{8}$$

We now prove some important properties of DWS scheduling. Define the congestion characteristic function $G(x, \underline{r})$ as:

$$G(x, \underline{r}) = \sum_{i=1}^{N} \min\left(x.W(r_i), r_i\right). \tag{9}$$

We now define the rate constant κ for a link with a vector of input rates $\underline{r}$ as :

$$\kappa = \begin{cases} \frac{C}{W(C)} & \text{if } \sum_{i=1}^{N} r_i \leq C \\ x : G(x, \underline{r}) = C & \text{if } \sum_{i=1}^{N} r_i > C \end{cases} \tag{10}$$

Theorem 3.1 (Rate Constant) *Rate constant κ as defined in Eq 10 is unique and the output rate of flow i is uniquely given by $\gamma_i = \min\left(\kappa.W(r_i), r_i\right)$.*

The proof is provided in the Appendix. Hence, given the input rates of flows, using the rate constant κ it is possible to uniquely determine the output rate of any flow. We define the fair rate f as follows:

$$f = \{x : x = \kappa.W(x)\} \tag{11}$$

Lemma 3.1 *The fair rate f as defined in Eq.11 is unique.*

The proof is provided in the Appendix. The output rate γ_i can be represented in terms of the fair rate as follows:

$$\gamma_i = \min(W(r_i)\frac{f}{W(f)}, r_i) \tag{12}$$

We now show that if the input rate of a flow is less than or equal to the fair rate, then the flow will get all its bits transmitted without loss, otherwise it will suffer a loss according to the diminishing weight function $W()$.

Lemma 3.2 *If $r_i \leq f$ then $\gamma_i = r_i$ else $\gamma_i = \kappa.W(r_i) < f$.*

Proof: **Case 1** ($r_i \leq f$): Note that $r_i \leq f \Rightarrow W(r_i) \geq W(f)$, since $W()$ is a strictly decreasing function. Hence we get $r_i/W(r_i) \leq f/W(f)$. Using Eq.11 we get $r_i \leq \kappa.W(r_i)$. Therefore, $\gamma_i = \min(\kappa.W(r_i), r_i) = r_i \leq f$.
Case 2 ($r_i > f$): In this case we get $r_i/W(r_i) > f/W(f)$, since $W()$ is a strictly decreasing function. Use Eq.11 to get $r_i > \kappa.W(r_i)$. Therefore, $\gamma_i = \min(\kappa.W(r_i), r_i) = \kappa.W(r_i)$. Since $\kappa = f/W(f)$ and $W(f) < W(r_i)$, we also have $\gamma_i < f$. □

Corollary 3.1 *The output rate for any flow i is less than equal to the fair rate ($\gamma_i \leq f$).*

The above behaviour is also evident from Figure 1. We say that a flow i is contributing to the link congestion iff $r_i > f$. With DWS scheduling, the output rate of a flow remains equal to its input rate as long as the flow is not contributing to congestion (i.e., the input rate is less than the fair rate). However, as soon as the flow contributes to the congestion (i.e., the input rate exceeds the fair rate) the output rate begins to decline according to the penalty function $W()$. In DWS, different weight functions can be chosen to meet specific requirements. Observe from Figure 1 that the weight function $W(r) = 1/\log(r)$ gives very small penalty to misbehaving flows and is very similar to WFQ whereas the weight function $W(r) = 1/r^4$ gives very strict penalties. The following lemma establishes the relationship between the fair rate f, the link capacity C and the number of users N. It also suggests that if all the users are equal (with equal DWS weights), then the fair rate is indeed fair.

Lemma 3.3 *Fair rate f is greater than or equal to C/N.*

The proof is provided in the Appendix. From Lemmas 3.2 and 3.3, observe that every user is guaranteed a fair rate of C/N unless it misbehaves. More flows contributing to the congestion increases f, thus giving incentives to flows not causing congestion and penalties to the congestion causing flows.

3.1 Packetized Diminishing Weight Schedulers (PDWS)

In a network, traffic does not flow as a fluid. Instead there are packets containing chunks of data that arrive at discrete time boundaries. Therefore, a scheduler is needed that works in such a scenario. Packetized DWS is derived from the DWS scheduler in the same way as the packetized GPS is derived from the GPS scheduler. Therefore the implementation details of PDWS are very similar to those of PGPS except for some changes in equations computing the timestamps. It should be straightforward to adapt PDWS to the simplifications of PGPS like Virtual Clock [41], Self Clocked Fair Queueing [14], WF2Q+ [40], Frame based Fair Queueing (FFQ) [35], etc.

Denote the arrival time of k^{th} packet of flow i as a_i^k, length of k^{th} packet of flow i as L_i^k. We model the k^{th} arrival of flow i as if it was a fluid flow of rate $r_i^k = L_i^k/(a_i^k - a_i^{k-1})$ in the interval $(a_i^{k-1}, a_i^k]$. The rate of arrival

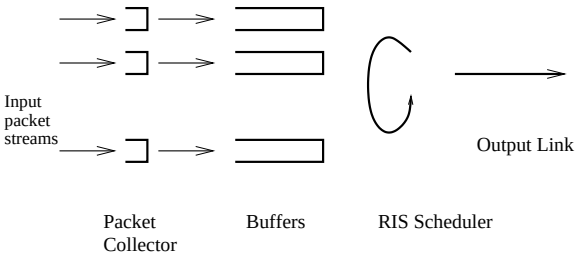

Figure 3: Hypothetical model for DWS with discrete packet boundaries

of all the bits of the packet is given by r_i^k and therefore this packet gets a GPS weight given by $\phi_i^k = \theta_i W(r_i^k)$. RIS being a special case of DWS has $\phi_i^k = \theta_i / r_i^k$ since $W(r_i^k) = 1/r_i^k$. However, the packet becomes eligible for service by the scheduler only after the last bit of the packet has arrived, i.e., at time a_i^k. We assume that there is a hypothetical packet collector before the DWS scheduler which collects all the bits of a packet and gives them to DWS only when they become eligible (see Figure 3).

Now, define the finish time of a packet as the time when the last bit of the packet gets serviced in a hypothetical DWS scheduler with a packet collector as shown in Figure 3. PDWS is defined as the scheduler that schedules packets in increasing order of their finish times.

Along the lines of PGPS implementation [37], PDWS is based on the concept of system virtual time and virtual finish time of packets. The scheduler maintains a virtual time function $v(t)$. Upon a packet arrival, each packet is tagged with a virtual finish time as follows:

$$F_i^k = \max\left(F_i^{k-1}, v(a_i^k)\right) + \frac{L_i^k}{\phi_i^k} \tag{13}$$

$$= \max\left(F_i^{k-1}, v(a_i^k)\right) + \frac{L_i^k}{\theta_i W(r_i^k)} \tag{14}$$

The packets are serviced in increasing order of their virtual finish times. To compute the virtual time at any instant, an emulation of hypothetical DWS system of Figure 3 is maintained which is similar to most PGPS implementations. When a packet k of flow i arrives, it is tagged with a GPS weight of ϕ_i^k and is also given to the DWS emulation. The emulation computes the virtual finish time of the packet by Eq 14. The rate of change of virtual time with real time is given by $d(v(t))/dt = C/(\sum_{i=1}^{N} \phi_i)$, where ϕ_i is the GPS weight corresponding to the packet of flow i which is currently in service in the DWS emulation.

In real practice traffic arrivals are bursty. Therefore, it is better to use a smoothed arrival rate, instead of instantaneous arrival rates for GPS weight computation. The scheduler PDWS with α smoothing sets $\phi_i^k = \theta_i W(\hat{r}_i^k)$, where $\hat{r}_i^k = \alpha \hat{r}_i^{k-1} + (1-\alpha) r_i^k$. The value of α is taken such that the half life of smoothing is of the order of one round trip time (R) when packet size of L_{max} is used to send traffic. This gives:

$$\alpha = 2^{-L_{max}/(fR)} \tag{15}$$

where f is the fair rate of the flow.

4 Properties of DWS

We now show some desirable game-theoretic properties of DWS Schedulers. In this section, for all results number of users $N \geq 2$ unless otherwise specified.

4.1 Single Link

With DWS scheduling, the vector of fair sending rates constitutes a unique Nash Equilibrium. This is formally illustrated in the following theorem.

Theorem 4.1 *Consider a link of capacity C, shared by N users, using DWS scheduling with unit DWS weights. Then $\forall_i r_i = C/N$ is the unique Nash Equilibrium for the system.*

The proof is provided in the Appendix. The property of Nash Equilibrium implies that if there are N users sharing a link of capacity C with each user sending its traffic at a rate C/N, then unlike FCFS, RED and DT policies, with DWS, a user has no incentive to unilaterally send its traffic at a different rate. Since Nash Equilibrium is unique with DWS (unlike WFQ), C/N is the only input rate that has this property.

However a sophisticated user can take advantage of the ignorance of naive users by leading it to a Stackelberg Equilibrium. We show that the Stackelberg Equilibrium coincides with the Nash Equilibrium.

Theorem 4.2 *Consider a link of capacity C, shared by N users, using DWS scheduling with unit DWS weights. Then $\forall_i r_i = C/N$ is the unique Stackelberg Equilibrium for the system.*

The proof is provided in the Appendix. Due to the coincidence of unique Nash and Stackelberg Equilibria, a user will benefit most by sending at its fair rate. Any user sending at a rate higher than its fair rate will be penalized, and other users can then receive a better output rate. This is characterized by the concept of Nash rate.

Definition 1 *Given a user with input rate r, define Nash rate for the remaining users as*

$$\eta(r) = \begin{cases} (C-r)/(N-1) & \text{if } r \leq C/N \\ x \geq 0 : xW(r)/W(x) + & \\ (N-1)x - C = 0 & \text{otherwise.} \end{cases} \tag{16}$$

When a user sends at r, the best strategy for other users is to send traffic at their nash rate $\eta(r)$. This is formally stated in the following Theorem.

Theorem 4.3 *If a user (say user 1) sends at r_1 then $\forall_{i\in(2,N)} r_i = \eta(r_1)$ is the unique Nash Equilibrium for the remaining $N-1$ users.*

The proof is provided in the Appendix. If a user misbehaves and sends traffic at a rate $r > C/N$, while the other users remain well behaved, then the fair rate f becomes equal to the nash rate $\eta(r)$, which is larger than C/N. In this case other users can safely increase their rate upto $\eta(r)$, whereas the misbehaving user gets penalized to its residual nash rate $\eta_R(r)$, defined as follows.

Definition 2 *Given a user with input rate r, define its residual Nash rate as:*

$$\eta_R(r) = C - (N-1)\eta(r) \tag{17}$$

Observe that if a user sends at $r_1 \leq C/N$, thereby not contributing to congestion, then the spare capacity is divided equally among the others. If a user sends at $r_1 > C/N$ then it gets penalized and the amount of penalty depends on the amount of congestion caused and the diminishing weight function. This is evident from the following Lemmas.

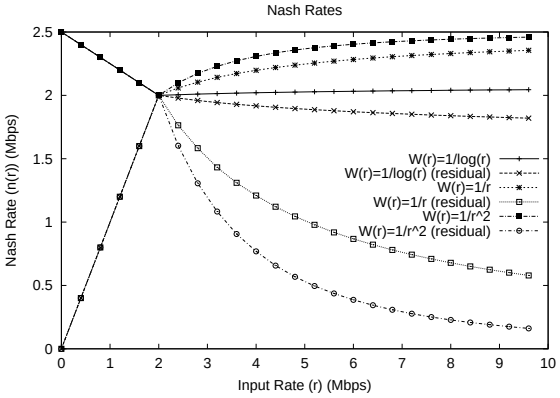

Figure 4: Nash rate

Theorem 4.4 *The Nash rate ($\eta(r)$) is a strictly increasing function of r in the range $(C/N, \infty)$.*

Proof: For $r \in (C/N, \infty)$, we rewrite definition 16 in the form

$$\eta(r) = x : W(r) = W(x)(\frac{C}{x} - (N-1))$$

Hence we see that r increases as x increases and vice-versa, since $W()$ is strictly decreasing. Also note that only a unique value of x satisfies the above equation for a given value of r. □

We now prove some lemmas based on the above theorem.

Lemma 4.1 *The Nash rate is greater than or equal to C/N i.e. $\eta(r) \geq C/N$.*

Proof: For $r \leq C/N$, we see from definition of Nash rate (Eq. 16) that $\eta(r) \geq C/N$.

For $r > C/N$, note from Theorem 4.4, that $\eta(r)$ is increasing in $(C/N, \infty)$. Also note that $\eta(r)$ is continuous at C/N (Eq. 16), and $\eta(C/N) = C/N$. Hence, $\eta(r) \geq C/N$. □

Lemma 4.2 *The resudual Nash rate is less than or equal to C/N i.e. $\eta_R(r) \leq C/N$.*

This immediately follows from Eq. 17 and Lemma 4.1.

Lemma 4.3 *$\eta_R(r)$ is strictly decreasing function of r in the range $(C/N, \infty)$.*

The proof immediately follows from Eq. 17 and Theorem 4.4.

A steep weight function will result in more severe punishment for a user contributing to congestion and larger output rate for well behaving users. This is illustrated in Figure 4 which plots Nash rate and residual Nash rate $(C - (N-1)\eta(r))$ for different diminishing weight functions. As can be easily observed, a steeper weight function($W(r) = 1/r^2$) results in a larger penalty as compared to a less steep weight function($W(r) = 1/log(r)$).

4.2 Arbitrary Network of Links

Max-min fairness [4] is a well known notion of fairness in an arbitrary network. Consider an arbitrary network servicing N flows. Denote by $\underline{\mathcal{M}}$, the $1 \times N$ vector of max-min fair rates of these flows through this network. The following theorem establishes that even in an arbitrary network of links, max-min fair input rates constitute

a Nash as well a Stackelberg Equilibrium if DWS schedulers are deployed at each link. Furthermore, its not necessary that the same weight function be used at each link. This makes it easier adopt DWS in a heterogenous environment with different administrative domains and policies.

Theorem 4.5 *Consider N users sending their traffic as N distinct flows through an arbitrary network with independent DWS scheduling at each link. Then the max-min fair rates $\underline{\mathcal{M}}$ constitute a Nash as well as a Stackelberg equilibrium for the users.*

The proof is provided in the Appendix. Besides $\underline{\mathcal{M}}$, there may be other equilibria also, and users may try to affect which equilibrium to reach. In such a case, it can be shown that atleast one user will experience losses in any other Nash equilibria. This is illustrated in the following Lemma.

Lemma 4.4 *Consider N users sending their traffic as N distinct flows through an arbitrary network with independent DWS scheduling at each link. $\underline{\mathcal{M}}$ is the unique Nash as well as the Stackelberg equilibrium in which there are no losses in the system.*

The proof is provided in the Appendix. In general, a user's utility may depend on its loss rate as well in addition to the output rate. Out of many Nash equilibrias giving the same output rates, generally users will prefer one with smaller losses. We formally define this class of utility functions ($U^{\gamma l}$) as follows:

1. $U \in U^{\gamma l}$ maps a user's output rate γ and loss-rate l to a real-valued non-negative utility.
2. $U(\gamma', l) > U(\gamma, l)$ iff $\gamma' > \gamma$.
3. $U(\gamma, l') < U(\gamma, l)$ iff $l' > l$.

If all users have such utility functions, it turns out that $\underline{\mathcal{M}}$ is the unique Nash as well Stackelberg Equilibrium. This is illustrated in the following theorem which is similar to Theorem 4.1 and Theorem 4.2 for a single link.

Theorem 4.6 *Consider N users sending their traffic as N distinct flows through an arbitrary network with independent DWS scheduling at each link. Let U_i be the utility function of user i. If $\forall_i U_i \in U^{\gamma l}$, then the max-min fair rates $\underline{\mathcal{M}}$ is the unique Nash and Stackelberg Equilibrium.*

The proof is provided in the Appendix. Therefore the "best selfish behaviour" for a user is to send traffic at its max-min fair rate.

5 Simulations

In this section we show the impact of using different weight functions on the penalty imposed to flows causing congestion. We also illustrate through a simulation that the TCP style Additive Increase Multiple Decrease (AIMD) algorithms are able to stabilize at the Nash rates in the presence of misbehaving users, when DWS scheduling is deployed [8, 9].

5.1 Simulation Scenario

The simulation scenario is shown in Figure 5. The buffer size for a flow at each link was set to its round trip delay- bandwidth product. Packetized DWS (PDWS) with per flow buffers and tail drop was used in all the simulations. NS [1] was used to carry out all the simulations.

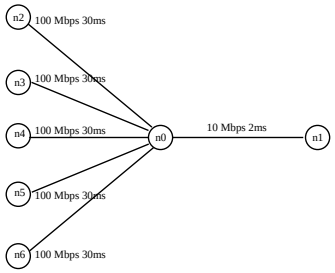

Figure 5: Simulation Scenario

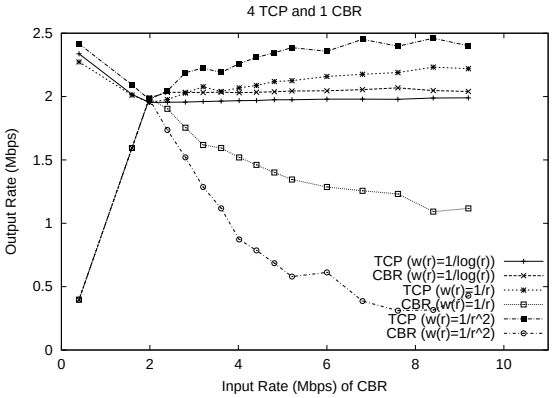

Figure 6: DWS Performance in the presence of TCP and CBR flows

5.2 CBR flows

The bottleneck link is shared by 5 CBR flows, 4 of which send at a constant rate of 4 Mbps and the fifth flow increases its rate from 0-7.2 Mbps. A plot of its output rate vs the input rate is shown in Figure 1 for various scheduling algorithms and buffer management policies.

This is a scenario of heavy congestion. Note from Figure 1 that with DWS scheduling the flow is able to receive its fair rate as long as it does not cause congestion. The flow starts receiving a penalty the moment its input rate exceeds the fair rate. The amount of penalty depends on the weight function $W()$. Note that the penalties are higher with $W(r) = 1/r^4$ and lower with $W(r) = 1/\log(r)$. Hence DWS rewards well behaved users and punishes misbehaving users.

5.3 TCP with unresponsive flows

The bottleneck link is shared by 5 users, 4 using (responsive) TCP Tahoe and one using unresponsive constant bit rate (CBR) source. Responsive TCP flows back off by reducing their sending rate upon detecting congestion while unresponsive CBR flows continue to inject packets into the network thus attempting to grab a larger share of the bandwidth. Figure 6 shows the average output rates for a representative TCP flow as the input rate of the CBR flow is varied. Each point in the graph represents a simulation of 20 seconds. However, the output rates correspond to the average rate in the last 10 seconds of the simulation when they get stabilized.

Figure 6 shows that the TCP flows are able to get close to their Nash rate (shown in Figure 4) according to Theorem 4.3.

Note that with $W(r) = 1/\log(r)$ the output rate of CBR flow is greater than that of the TCP flow. This is because the inverse log weight function gives very little penalty to CBR and is very similar to WFQ. The

bandwidth left by TCP because of timeouts and retransmits is grabbed by the CBR flow despite its small weight. As CBR increases its rate further, the penalty slowly increases allowing TCP to grab a larger share.

6 Conclusions

We noted that most of the current congestion control techniques are voluntary in nature. We argued that in the presence of selfish users, all such schemes will lead to a congestion collapse of the Internet.

In this paper we extended the Rate Inverse Scheduling (RIS) algorithm [13] to a class of scheduling algorithms called Diminishing Weight Scheduling (DWS). DWS has several desirable game-theoretic properties similar to RIS [13]. For a single link shared by N users using DWS, fair rate constitutes the unique Nash and Stackelberg equilibrium. For an arbitrary network with DWS scheduling at every link, max-min fair rate constitutes the unique Nash and Stackelberg equilibrium. DWS does not require different nodes to use the same weight function. Therefore, it is well suited for heterogenous environment consisting of a number of different administrative domains.

It is possible to set different DWS weights for different users (or traffic classes). This should lead to (in a Game-theoretic manner) weighted fair sharing in case of a single link and weighted max-min fair sharing in case of a network. These weights may be set in accordance with the pricing or other resource sharing policies.

DWS encourages the users to send traffic at their max-min fair rates and punishes the users sending at higher rates. We defined the concept of Nash rate and showed how the choice of different weight functions can affect the reward-penalty profile of DWS. Using simulations we showed that with DWS, the TCP users indeed get rewarded (according to their Nash rates) in the presence of unresponsive, misbehaving CBR flows (who get punished). With the $1/r^2$ diminishing weight function, the penalty imposed is large, whereas with $1/\log(r)$ diminishing weight function, the behaviour of DWS is only marginally different from that of WFQ which imposes no penalty. Thus, different administrative domains can choose their own diminishing weight functions to match their policies.

One of the major limitation of DWS is that it requires a per-flow state in switches and routers. However, we hope that based on this work, it will be possible to design "core stateless" policies [36] with similar properties.

Another issue that needs to be examined is about the choice of the diminishing weight function. What is a good weight function that would give adequate incentive to the end users to adopt the TCP like end user behaviour?

References

[1] Network simulator (NS), 2001. URL: http://www.isi.edu/nsnam/ns/.

[2] A. Albanese, J. Blomer, J. Edmonds, M. Luby, and M. Sudan. Priority encoding transmission. In *IEEE Symposium on Foundations of Computer Science*, pages 604–612, 1994.

[3] M. Allman, V. Paxson, and W. Stevens. TCP congestion control. Request for Comments 2581, Internet Engineering Task Force, Apr. 1999.

[4] D. P. Bertsekas and R. G. Gallager. *Data Networks*. Prentice Hall, Englewood Cliffs, NJ, USA, 1992.

[5] L. S. Brakmo and S. W. O'Malley. TCP Vegas: New techniques for congestion detection and avoidance. In *Proceedings of SIGCOMM*, pages 34–35, London, United Kingdom, Aug. 1994. ACM.

[6] A. K. Choudhury and E. L. Hahne. Dynamic queue length thresholds in a shared memory ATM switch. In *Proceedings of INFOCOM*, San Fransisco, California, Mar. 1996.

[7] A. Demers, S. Keshav, and S. Shenker. Analysis and simulation of a fair queueing algorithm. *Journal of Internetworking Research and Experience*, pages 3–26, Oct. 1990. Also in Proc. ACM SIGCOMM'89, pp 3-12.

[8] S. Floyd. Congestion control principles. Request for Comments 2914, Internet Engineering Task Force, Sept. 2000.

[9] S. Floyd and K. Fall. Promoting the use of end-to-end congestion control in the internet. *IEEE/ACM Transactions on Networking*, August 1999.

[10] S. Floyd and T. Henderson. The NewReno modification to TCP's fast recovery algorithm. Request for Comments 2582, Internet Engineering Task Force, Apr. 1999.

[11] S. Floyd and V. Jacobson. Random early detection gateways for congestion avoidance. *ACM/IEEE Transactions on Networking*, 1(4):397–413, Aug. 1993.

[12] D. Fudenberg and J. Tirole. *Game Theory*. MIT Press, October 1991.

[13] R. Garg, A. Kamra, and V. Khurana. Eliciting cooperation from selfish users: A game-theoretic approach towards congestion control in communication networks. Technical Report RI01001, IBM Research, April 2001. Available at http://www.research.ibm.com/resources/paper_search.html.

[14] S. J. Golestani. A self-clocked fair queuing scheme for broadband applications. In *Proceedings of INFOCOM*, pages 636–646, April 1994.

[15] G. Hardin. The tragedy of the commons. *Science*, 162:1243–1248, 1968.

[16] V. Jacobson. Congestion avoidance and control. *Computer Communications Review*, 18(4):314–329, Aug. 1988. Proceedings of the Sigcomm '88 Symposium in Stanford, CA, August, 1988.

[17] R. Jain, S. Kalyanaraman, R. Goyal, S. Fahmy, and R. Viswanathan. ERICA switch algorithm: A complete description. Technical report, ATM Forum/96-1172, August 1996.

[18] F. Kelly, A. Maulloo, and D. Tan. Rate control in communication networks: Shadow prices, proportional fairness and stability. *Journal of the Operational Research Society*, 49:237–252, 1998.

[19] Y. A. Korilis, A. A. Lazar, and A. Orda. The designer's perspective to noncooperative networks. In *Proceedings of INFOCOM*, Boston, Massachusetts, Apr. 1995.

[20] H. T. Kung, T. Blackwell, and A. Chapman. Credit update protocol for flow-controlled ATM networks: Statistical multiplexing and adaptive credit allocation. In *Proceedings of SIGCOMM*, pages 101–114, London, UK, Sept. 1994.

[21] R. La and V. Anantharam. Charge-sensitive TCP and rate control in the Internet. In *Proceedings of INFOCOM*, Tel Aviv, Israel, Mar. 2000.

[22] D. Lin and R. Morris. Dynamics of random early detection. In *Proceedings of the ACM SIGCOMM'97 Conference*, pages 127–138, New York, September 1997.

[23] M. Mathis and J. Mahdavi. Forward acknowledgement: Refining TCP congestion control. *Computer Communications Review*, 26(4):281–291, Oct. 1996.

[24] M. Mathis, J. Mahdavi, S. Floyd, and A. Romanow. TCP selective acknowledgement options. Request for Comments 2018, Internet Engineering Task Force, Oct. 1996.

[25] J. Mo, R. La, V. Anantharam, and J. Walrand. Analysis and comparison of TCP Reno and Vegas. In *Proceedings of INFOCOM*, New York, Mar. 1999.

[26] J. Nagle. Congestion control in IP/TCP internetworks. Request for Comments 896, Internet Engineering Task Force, Jan. 1984.

[27] A. Orda, R. Rom, and N. Shimkin. Competitive routing in multiuser communication networks. *ACM/IEEE Transactions on Networking*, 1(5):510–521, Oct. 1993.

[28] A. K. Parekh and R. G. Gallager. A generalized processor sharing approach to flow control - the single node case. In *Proceedings of INFOCOM*, pages 915–924. IEEE, May 1992.

[29] J. Postel. Transmission control protocol. Request for Comments 793, Internet Engineering Task Force, Sept. 1981.

[30] W. Prue and J. Postel. Something a host could do with source quench: The source quench introduced delay (SQuID). Request for Comments 1016, Internet Engineering Task Force, July 1987.

[31] K. Ramakrishnan and S. Floyd. A proposal to add explicit congestion notification (ECN) to IP. Request for Comments 2481, Internet Engineering Task Force, Jan. 1999.

[32] K. Ramakrishnan and R. Jain. A binary feedback scheme for congestion avoidance in computer networks,. *ACM Trans. on Computer Systems*, 8(2):158–181, May 1990.

[33] A. Romanow and S. Floyd. Dynamics of TCP traffic over ATM networks. *IEEE Journal on Selected Areas of Communications*, 13(4):633–641, May 1995.

[34] S. Shenker. Making greed work in networks: A game-theoretic analysis of switch service disciplines. In *Proceedings of SIGCOMM*, pages 47–57, London, UK, Sept. 1994.

[35] D. Stiliadis and A. Varma. Design and analysis of frame-based fair queuing: A new traffic scheduling algorithm for packet switched networks. In *Proceedings of the ACM SIGMETRICS International Conference on Measurement and Modeling of Computer Systems*, pages 104–115, New York, May 1996.

[36] I. Stoica, S. Shenker, and H. Zhang. Core-stateless fair queueing: Achieving approximately fair bandwidth allocations in high speed networks. In *ACM Sigcomm '98*, Septermber 1998.

[37] A. Varma and D. Stiliadis. Hardware implementation of fair queueing algorithms for asynchronous transfer mode networks. *IEEE Communications Magzine*, pages 54–68, December 1997.

[38] H. Yaiche, R. R. Mazumdar, and C. Rosenberg. A game theoretic framework for bandwidth allocation and pricing in broadband networks. *IEEE/ACM Transactions on Networking*, 8(5):667–678, October 2000.

[39] C.-Q. Yang and A. V. S. Reddy. A taxonomy for congestion control algorithms in packet switching networks. *IEEE Network Magazine*, 9(5), July/August 1995.

[40] H. Zhang and J. C. R. Bennett. Hierarchical packet fair queueing algorithms. In *Proceedings of ACM SIGCOMM*, pages 143–156. ACM, September 1997.

[41] L. Zhang. Virtual clock : A new traffic control algorithm for packet switching networks. *ACM Transactions on Computer Systems*, 9:101–124, May 1991.

7 Appendix

Before proceeding to the proofs of our main theorems, we prove certain properties.

Lemma 7.1 *The congestion characteristic function $G(x, \underline{r})$, as defined in Eq. 9, is a strictly increasing function of x in $(0, k_{max})$ and is constant in $[k_{max}, \infty)$, where $k_{max} = \max_{0 \le i \le N}\{r_i/W(r_i)\}$.*

Proof: Note that

$$x < x' \Rightarrow, \forall j : \min(x.W(r_j), r_j) \le \min(x'.W(r_j), r_j). \tag{18}$$

Therefore, $G(x, \underline{r}) \le G(x', \underline{r})$.

Consider $x \in (0, k_{max})$, since $0 < x < k_{max}$, $\exists i$ s.t. $x < r_i/W(r_i)$ by definition of k_{max}. Therefore, $\min(x.W(r_i), r_i) = x.W(r_i)$. Hence for $x < x' < k_{max}$, we get $\min(x.W(r_i), r_i) < \min(x'.W(r_i), r_i)$. For all other flows j Eq. 18 holds. Hence summing over all flows we get $G(x, \underline{r}) < G(x', \underline{r})$.

Now consider $x \in [k_{max}, \infty)$, since $x \ge k_{max}$ we get $\forall i\ x \ge \frac{r_i}{W(r_i)}$. Therefore, $\forall i\ \min(x.W(r_i), r_i) = r_i$. The same holds for any $x' > x$. Hence we get, $\forall i\ \min(x.W(r_i), r_i) =$
$\min(x'.W(r_i), r_i) = r_i$. Summing over all flows we get $G(x, \underline{r}) = G(x', \underline{r}) = \sum_{i=1}^{N} r_i$ □

Proof (Theorem 3.1): [**Uniqueness of** κ] If $\sum_{i=1}^{N} r_i \leq C$, then κ is unique by definition. Using Lemma 7.1, $G(x, \underline{r})$ is a monotonically increasing function of x in the domain $(0, k_{max})$. Now if $\sum_{i=1}^{N} r_i > C$ then $G(k_{max}, \underline{r}) = \sum_i r_i > C$. Since κ satisfies $G(x, \underline{r}) = C$, using Lemma 7.1, $\kappa < k_{max}$. Therefore, κ is unique.
[**Output rate**] If $\sum_{i=1}^{N} r_i \leq C$, from the flow conservation and the work conservation properties of DWS scheduler, $\gamma_i = r_i \leq CW(r_i)/W(C)$, (since $r_i \leq C$ and $W()$ is strictly decreasing). Since $\kappa = C/W(C)$, it follows that $\gamma_i = \min(\kappa.W(r_i), r_i)$.

If $\sum_{i=1}^{N} r_i > C$, then there must exist at least one flow i such that $\gamma_i < r_i$. Flow i will have a non-zero backlog. From the DWS fairness property, Eq. 8, we have $\gamma_i = k.W(r_i) < r_i$. It follows that

$$\gamma_i = \min(k.W(r_i), r_i). \tag{19}$$

Now consider a flow j such that $\gamma_j = r_j$. From DWS fairness, Eq. 7, we get $\gamma_j/W(r_j) \leq k$. Therefore, $\gamma_j = r_j \leq k.W(r_j)$. It follows that

$$\gamma_j = \min(k.W(r_j), r_j). \tag{20}$$

Since the scheduler has a positive backlog, from the flow conservation and work conservation properties of DWS, $\sum_{i=1}^{N} \gamma_i = C$. Substituting values of γ_is of backlogged flows from Eq. 19 and γ_is of non-backlogged flows from Eq. 20, we get :

$$\sum_{i=1}^{N} \min(k.W(r_i), r_i) = C.$$

Since the rate constant κ is unique value satisfying the equation $G(x, \underline{r}) = C$, we get $k = \kappa$. Therefore, $\gamma_i = \min(\kappa.W(r_i), r_i)$ for all flows i. □

Proof (Lemma 3.1): ¿From Theorem 3.1, κ is unique. We show the uniqueness of f using contradiction. Suppose f is not unique then there are 2 distinct points x_1 and x_2 which satisfy $x = \kappa.W(x)$. Since x_1 and x_2 are distinct, WLOG assume that $x_1 < x_2$. Since $x_1 = \kappa.W(x_1)$ and $x_2 = \kappa.W(x_2)$ we get $W(x_1) < W(x_2)$ (note κ is always positive). However, since $W()$ is strictly decreasing $x_1 < x_2 \Rightarrow W(x_1) > W(x_2)$ which leads to a contradiction. Hence f is unique. □

Proof (Lemma 3.3): If $\sum_{i=1}^{N} r_i \leq C$ then $\kappa = C/W(C)$. Using Eq. 11 we get $f/W(f) = C/W(C)$. Hence $f = C \geq C/N$.

If $\sum_{i=1}^{N} r_i > C$, then from Corollary 3.1, $f \geq \gamma_i$. Summing over all flows we get $fN \geq \sum_{i=1}^{N} \gamma_i$. But from the flow conservation and work conservation properties of DWS scheduler $\sum_{i=1}^{N} \gamma_i = C$, therefore $f \geq C/N$. □

Lemma 7.2 *If the input rate of flow j is decreased to r'_j from r_j ($r'_j < r_j$), then either $\gamma'_j \geq \gamma_j$ or $\gamma'_j = r'_j$.*

Proof: If $\sum_{i=1}^{N} r_i \leq C$, then we have $\forall_i \gamma_i = r_i$, from the flow conservation and work conservation properties of the DWS scheduler. If r_j is decreased to r'_j, then $\sum_{i=1(i\neq j)}^{N} r_i + r'_j \leq C$ still holds. Therefore, $\gamma'_j = r'_j$ even now.

We now consider the case when $\sum_{i=1}^{N} r_i > C$. Again from the flow conservation and work conservation properties of the DWS scheduler, we have $\sum_{i=1}^{N} \gamma_i = C$. Using Lemma 3.2 we get $\sum_m r_m + \sum_n \kappa.W(r_n) = C$ where for all flows m, $r_m \leq f$ and for all flows n, $r_n > f$. Therefore,

$$\kappa = \frac{C - \sum_m r_m}{\sum_n W(r_n)} \tag{21}$$

We see that the value of κ also changes to κ' on changing the value of r_j. We have the following cases:

Case 1 ($r_j \leq f$): Since $r_j \leq f$, j is one among the m flows. We rewrite Eq. 21 in the form

$$\kappa = \frac{C - \sum_{m(m \neq j)} r_m - r_j}{\sum_n W(r_n)} \tag{22}$$

Hence we see that κ increases on decreasing r_j, and hence f also increases (using Eq. 11). Therefore, $r'_j < r_j \leq f < f'$. Since $r'_j < f'$, using Lemma 3.2 we get $\gamma'_j = r'_j$.

Case 2 ($r_j > f$): Since $r_j > f$, j is one among the n flows. We rewrite Eq. 21 in the form

$$\kappa = \frac{C - \sum_m r_m}{\sum_{n(n \neq j)} W(r_n) + W(r_j)} \tag{23}$$

We again have two cases, if $r'_j \leq f'$, then using Lemma 3.2 $\gamma'_j = r'_j$. If $r'_j > f'$, we show that $\gamma'_j > \gamma_j$. Define the residual capacity by $C_{res} = C - \sum_m r_m$. Since $r_j > f$ and $r'_j > f'$ we have:

$$\gamma_j = \kappa.W(r_j) = \frac{C_{res}}{(\frac{1}{W(r_j)} \sum_{n(n \neq j)} W(r_n) + 1)} \tag{24}$$

$$\gamma'_j = \kappa'.W(r'_j) = \frac{C_{res}}{(\frac{1}{W(r'_j)} \sum_{n(n \neq j)} W(r_n) + 1)} \tag{25}$$

using Eq. 23. Since $r_j > r'_j$ and $W()$ is a strictly decreasing function, we get $\gamma'_j > \gamma_j$. □

Lemma 7.3 (Flow removal) *If $\sum_{i=1, i \neq j}^{N} r_i > C - \gamma_j$ then removal of flow j and adjustment of link capacity as $C' = C - \gamma_j$ does not change the rate constant κ or the fair rate f.*

Proof: From Theorem 3.1, rate constant κ satisfies $G(\kappa, \underline{r}) = C$. Rewrite the above equation in the form

$$\sum_{i=1 (i \neq j)}^{N} \min(\kappa.W(r_i), r_i) = C - \gamma_j \tag{26}$$

Substitute the new capacity C' and the new rate vector $\underline{r}'$ in the above equation to get $G(\kappa, \underline{r}') = C'$. Since $\sum_{i=1, i \neq j}^{N} r_i > C'$, and κ is the unique rate satisfying $G(\kappa, \underline{r}') = C'$, the rate constant κ remains unaffected by removing flow j.

Since the rate constant κ is unaffected, the fair rate f also remains unchanged. □

Lemma 7.4 *If $r > C/N$ then: (1). $\eta(r) < r$ and (2). $\forall \underline{r} : r_1 = r, f(\underline{r}) \geq \eta(r) > C/N$.*

Proof (Lemma 7.4): If $r > C/N$ then $\eta(r)$ satisfies:

$$\frac{xW(r)}{W(x)} + (N-1)x = C \tag{27}$$

Since LHS is a strictly increasing function of x, it is easy to see that there is exactly one positive real root of the above equation.

1. Suppose $\eta(r) \geq r$. Then LHS of equation 27 becomes $\frac{\eta(r)W(r)}{W(\eta(r))} + (N-1)\eta(r) \geq Nr > C$ since $W()$ is a decreasing function and $r > C/N$. So equation 27 can never be satisfied. Hence $\eta(r) < r$.

2. Consider an arbitrary rate $r' \geq \eta(r)$. Since $W()$ is a diminishing weight function, $\frac{W(r')}{W(\eta(r))} \leq 1$. Therefore, $\eta(r)\frac{W(r')}{W(\eta(r))} \leq \eta(r) \leq r'$ and hence $min(r', \eta(r)\frac{W(r')}{W(\eta(r))}) \leq \eta(r)$.

   Now consider an $r' < \eta(r)$. Clearly, $min(r', \eta(r)\frac{W(r')}{W(\eta(r))}) = r' \leq \eta(r)$. Therefore, for any rate r',

$$min(r', \eta(r)\frac{W(r')}{W(\eta(r))}) \leq \eta(r) \tag{28}$$

   Also

$$min(r_1, \eta(r)\frac{W(r_1)}{W(\eta(r))}) = \eta(r)\frac{W(r)}{W(\eta(r))} \tag{29}$$

   Substituting $r' = r_i$ in Eq. 28 and summing over $i = 2..N$ and adding Eq. 29, we get $\sum_{i=1}^{N} min(r_i, \eta(r)\frac{W(r_i)}{W(\eta(r))}) \leq \eta(r)\frac{W(r)}{W(\eta(r))} + (N-1)\eta(r)$. Substituting from Eqs. 9 and 16, we get $G(\frac{\eta(r)}{W(\eta(r))}, \underline{r}) \leq C$. ¿From the above equation and from the definition of κ in Eq. 10, and the fact that $G(x, \underline{r})$ is an increasing function of x, it is easy to see that $\kappa \geq \frac{\eta(r)}{W(\eta(r))}$. ¿From Eq. 11, $f = \kappa W(f)$ and hence $\frac{f}{W(f)} \geq \frac{\eta(r)}{W(\eta(r))}$.

   Since $W()$ is a monotonically decreasing function, no $f < \eta(r)$ can satisfy the above equation. Hence $f \geq \eta(r)$.

□

Proof (Theorem 4.1): [**Nash Equilibrium**] Let $\forall_i r_i = C/N$. Then $\forall_i \gamma_i = C/N$ since $f \geq C/N$ always. WLOG, suppose that user 1 changes its input rate to r_1'.

Case 1: $\mathbf{r_1' < C/N}$: Clearly, $\gamma_1 < C/N$.

Case 2: $\mathbf{r_1' > C/N}$: Now $\sum_{i=1}^{N} r_i > C$, so the scheduler will have a positive backlog. ¿From Lemma 3.3, $f \geq C/N$. Now $\forall_i r_i = C/N$, so from Lemma 3.2, $\forall_i \gamma_i = C/N$. So, from the flow conservation and work conservation properties of DWS $\gamma_1 = C - \sum_{i \in (2..N)} \gamma_i = C/N$.

So user 1 can never increase its output rate by unilaterally changing its input rate. Hence $\forall_i r_i = C/N$ is a Nash equilibrium.

[**Uniqueness**] Assume WLOG a vector for input rates $\underline{r}$ constituting a Nash Equilibrium, s.t $r_1 \neq C/N$. There are two cases:

Case 1: $\mathbf{\exists_i r_i < C/N}$: Here $\gamma_i < C/N$. So by increasing r_i to C/N, γ_i can be increased to C/N. Therefore $\underline{r}$ does not constitute a Nash Equilibrium.

Case 2: $\mathbf{\forall_i r_i \geq C/N}$ and $\mathbf{r_1 > C/N}$: WLOG, assume that $r_1 = max(r_1, r_2, ...r_N) > C/N$. There are three subcases:

(a) $\mathbf{\exists_{i \geq 2} \gamma_i < \eta(r_1)}$: Using Lemma 7.4, $\forall_{r_2, r_3 ... r_N} f \geq \eta(r_1)$. So by making $r_i = \eta(r_1)$, γ_i can be increased to $\eta(r_1)$ (Lemma 3.2).

(b) $\mathbf{\forall_{i \geq 2} \gamma_i = \eta(r_1)}$: $\gamma_1 = C - \sum_{i \in (2,N)} \gamma_i = C - (N-1)\eta(r_1) < C/N$. So by making $r_1 = C/N$, γ_1 can be increased to C/N.

(c) $\mathbf{\forall_{i \geq 2} \gamma_i \geq \eta(r_1)}$ and $\mathbf{\exists_{j \geq 2} \gamma_j > \eta(r_1)}$: It is easy to see that $f < r_{max} = r_1$. From Lemma 7.4 $f \geq \eta(r_1)$. Now $f = \kappa W(f)$ and since $\eta(r_1) \leq f$, it follows that $\eta(r_1) < \kappa W(\eta(r_1))$. So $\gamma_1 = \kappa W(r_1) \geq \frac{\eta(r_1) W(r_1)}{W(\eta(r_1))}$. Now $\sum_i \gamma_i = \gamma_1 + \gamma_j + \sum_{i \geq 2, i \neq j} \gamma_i > \frac{\eta(r_1) W(r_1)}{W(\eta(r_1))} + \eta(r_1) + (N-2)\eta(r_1) = C$. So this case is not possible.

It follows that $\forall_i r_i = C/N$ is a unique Nash equilibrium. □

Proof (Theorem 4.3): There are two cases:

Case 1: $\mathbf{r_1 \leq C/N}$: In this case $\gamma_1 = r_1$, irrespective of the value of r_1 because $f \geq C/N$. ¿From Lemma 7.3, removing flow 1 will neither change the rate constant κ nor the fair rate f. The spare capacity $C_{res} = C - r_1$, will be divided among the remaining users. Using Theorem 4.1, the unique Nash equilibrium for the remaining users is when each sends at $C_{res}/(N-1)$ which is $\eta(r_1)$.

Case 2: $\mathbf{r_1 > C/N}$: Let $\forall_{i\in(2,N)} r_i = \eta(r_1)$. We now show that this vector of input rates constitute a Nash Equilibrium for the remaining users. Using Lemma 7.4, $f \geq \eta(r_1)$ so $\gamma_{i\in(2,N)} = \eta(r_1)$. WLOG, assume that user 2 deviates its rate from $r_2 = \eta(r_1)$ to r_2'. There are the following two subcases:

(a) $\mathbf{r_2' < \eta(r_1)}$: Here $\gamma_2' = r_2 < \eta(r_1)$. So clearly this cannot be a Nash Equilibrium.

(b) $\mathbf{r_2' > \eta(r_1)}$: If f' is the new fair rate, then $f' \geq \eta(r_1)$. Since $W(x, \underline{r})$ is a decreasing function of x, we get $\kappa' = \frac{f'}{W(f')} \geq \frac{\eta(r_1)}{W(\eta(r_1))}$. Now using Eq. 29, $\gamma_1' = \min(r_1, \kappa' W(r_1)) = \kappa' W(r_1) \geq \frac{\eta(r_1)W(r_1)}{W(\eta(r_1))}$ and so $\gamma_2' = C - \sum_{i\neq 2} \gamma_i' \leq C - (\frac{\eta(r_1)W(r_1)}{W(\eta(r_1))} + (N-2)\eta(r_1)) = \eta(r_1)$. So the output rate of user 2 cannot increase when it deviates from $\underline{r}$. Hence $\underline{r}$ is a Nash equilibrium for the N - 1 user subsystem.

We now show that the above Nash Equilibrium is unique. Consider any other input rate profile $\underline{r'}$. Let $\underline{\gamma'}$ be the corresponding output rate profile. There are two subcases:

(a) $\mathbf{\sum_{i=1}^{N} \gamma_i' < C}$: Clearly $\exists_j$ s.t. $\gamma_j' = r_j' < C/N$. Now, r_j' can be increased to C/N to get an output rate of C/N. So this cannot be a Nash Equilibrium.

(b) $\mathbf{\sum_{i=1}^{N} \gamma_i = C \text{ and } \exists_{i\in(2,N)} r_i \neq \eta(r_1)}$: Since $f \geq \eta(r_1)$ and $W(x, \underline{r})$ is an increasing function of x, we get $\kappa = \frac{f}{W(f)} \geq \frac{\eta(r_1)}{W(\eta(r_1))}$. Using Eq. 29, we get $\gamma_1 = \frac{f}{W(f)} W(r_1) \geq \frac{\eta(r_1)}{W(\eta(r_1))} W(r_1)$. ¿From the flow conservation and work conservation properties, $\sum_{i\in(2,N)} \gamma_i = C - \gamma_1 \leq C - \frac{\eta(r_1)}{W(\eta(r_1))} W(r_1) = (N-1)\eta(r_1)$.
$\Rightarrow \exists_j \gamma_j \leq$ ‘$\eta(r_1)$. Since by making $r_j = \eta(r_1)$, γ_j can be made equal to $\eta(r_1)$, this can never be a Nash equilibrium for the N-1 user subsystem.

Hence if user 1 sends at r_1, then $\forall_{i\in(2,N)} r_i = \eta(r_1)$ is the unique Nash equilibrium for the remaining N-1 users. □

Proof (Theorem 4.2): WLOG assume that user 1 becomes the leader and sends at r_1 in a Stackelberg Equilibrium.

Case 1: $\mathbf{r_1 < C/N \text{ and } \gamma_1 < C/N}$: This cannot be the case since $f \geq C/N$ and making $r_1' = C/N$ makes $\gamma_1' = C/N > \gamma_1$.

Case 2: $\mathbf{r_1 > C/N}$: Using Theorem 4.3, the unique Nash Equilibrium for the remaining flows is when the input and output rate of each of the N-1 users is equal to $\eta(r_1)$. ¿From Lemma 7.4 $\eta(r_1) > C/N \Rightarrow \gamma_1 \leq C - \sum_{i\in(2,N)} \gamma_i = C - (N-1)\eta(r_1) < C - (N-1)C/N = C/N$. This cannot be a Stackelberg Equilibrium as making $r_1' = C/N$ makes $\gamma_1' = C/N > \gamma_1$. The only case left is the following.

Case 3: $\mathbf{r_1 = C/N}$: $\gamma_1 = C/N$ since $f \geq C/N$ always.

So every Stackelberg Equilibrium will have $r_1 = C/N$. Since the unique Nash Equilibrium for other flows is $\forall_{i\in(2,N)} r_i = \eta(r_1) = C/N$, it follows that the only Stackelberg equilibrium is when $\forall_i r_i = C/N$. □

Before proving Lemma 4.4 we give a brief overview of max-min fair rate computation in an arbitrary network. Let C_ℓ denote the capcacity of link ℓ and N_ℓ denote the number of flows passing through link ℓ. Let F_ℓ be the set of flows passing through link ℓ.

The algorithm to compute max-min fair rates [4] works as follows. At every stage k (k is initially 1), it finds the minimum bottleneck link ℓ s.t. for all other links ℓ', $C_\ell^k/N_\ell^k \leq C_{\ell'}^k/N_{\ell'}^k$. WLOG, it labels this link as ℓ_k. For all residual flows P^k passing through ℓ_k (k = 1), it assigns max-min fair rates $\mathcal{M}_{i\in P^k} = C_\ell^k/N_\ell^k$, where C_ℓ^k and N_ℓ^k are respectively the residual capacity and the residual number of flows passing through the link ℓ_k. The link ℓ_k and all the flows in P^k are removed and the residual capacities and residual number of flows of all the other links are updated as follows:

$$C_\ell^{k+1} = C_\ell^k - \sum_{i\in P^k, i\in F_\ell} \mathcal{M}_i \tag{30}$$

$$N_\ell^{k+1} = N_\ell^k - \sum_{i\in P^k, i\in F_\ell} 1 \tag{31}$$

Initially, $C_\ell^1 = C_\ell$ and $N_\ell^1 = N_\ell$. The algorithm terminates when there are no more flows to be removed.

Proof (Lemma 4.4): Consider the set of flows F_{ℓ_1} through the minimum botleneck link ℓ_1.

Claim 1 *In any Nash or Stackelberg Equilibrium without losses, flows $i \in F_{\ell_1}$ will have $r_i = \gamma_i = C_{\ell_1}/N_{\ell_1}$.*

Proof: Since there are no losses, $\forall_i \gamma_i = r_i$ and $\sum_{i\in F_{\ell_1}} r_i \leq C_{\ell_1}$.

Assume for contradiction, $\exists i \in F_{\ell_1}$ s.t. $r_i < C_{\ell_1}/N_{\ell_1}$.

Since ℓ_1 is the minimum bottleneck link, the fair rate f_ℓ of any other link ℓ as defined by Eq. 11 is greater than or equal to C_{ℓ_1}/N_{ℓ_1}. Set r_i' equal to C_{ℓ_1}/N_{ℓ_1} and the new output rate at each link will satisfy $\gamma_i' = r_i' > \gamma_i$. So this profile of rates cannot constitute a Nash equilibrium.

Now assume WLOG, that flow $1 \in F_{\ell_1}$ becomes the leader in a Stackelberg Equilibrium. If $r_1 = C_{\ell_1}/N_{\ell_1}$, then using a similar argument as above, Nash Equilibrium of the remaining flows $i \in F_{\ell_1}, i \neq 1$ is when $r_i = C_{\ell_1}/N_{\ell_1}$. In this case γ_1 becomes equal to C_{ℓ_1}/N_{ℓ_1}. So any Stackelberg Equilibrium with flow 1 as leader should have $\gamma_1 \geq C_{\ell_1}/N_{\ell_1}$.

If $r_1 > C_{\ell_1}/N_{\ell_1}$ then the Nash rate of the remaining flows will be atleast C_{ℓ_1}/N_{ℓ_1}. Therefore flow 1 will have losses. Hence the proof. □

To extend Claim 1 to the subsequent bottleneck links, we use the flow removal property of Lemma 7.3. Since the flows in F_{ℓ_1} do not get any losses, they may be removed from the network without changing the fair rate f and the rate constant κ at any other link.

Therefore we have the following property:

In any Nash or Stackelberg Equilibrium without losses, any flow $i \in F_{\ell_k}$ s.t. $i \notin F_{\ell_{k'}}$ where $k' < k$ will have $\gamma_i = r_i = C_{\ell_k}^k/N_{\ell_k}^k = \mathcal{M}_i$. This is the set of max min fair rates and will hence constitute the unique Nash and Stackelberg Equilibrium without losses. □

Proof (Theorem 4.5): Follows directly from Lemma 4.4. □

Lemma 7.5 *Let U_i be the utility function for user i. If $\forall_i U_i \in U^{\gamma l}$, then in any Nash or Stackelberg equilibrium, no user will experience losses.*

Proof:

- **Nash**: Suppose that for user i, $\gamma_i < r_i$. So the loss-rate is $l_i = r_i - \gamma_i > 0$. If user i changes its input rate to $r_i' = \gamma_i$, its input rate at the first link is decreased. Using Lemma 7.2 if the output rate should either stay the same, or increase, or become equal to the new input rate. Since $r_i' = \gamma_i$, the output rate at this link becomes equal to its new input rate, γ_i. The same argument applies to all the following links in the path. Therefore $\gamma_i' = \gamma_i$. The new loss rate is therefore zero. Since loss rate is decreased and output rate is same as before, user i's utility increases. So in any Nash equilibrium, no user can have losses.

- **Stackelberg**: In any Stackelberg equilibrium, the leader cannot have any losses, else it can increase its utility by decreasing the input rate to its current output rate. All the other flows are in Nash equilibrium, so by a similar argument they also cannot have any losses.

□

Proof (Theorem 4.6): ¿From Lemma 7.5, any Nash or Stackelberg equilibrium with utility functions $U_i \in U^{\gamma l}$ cannot have losses. If the loss rate is zero, then the a user's utility becomes only a function of its output rate and hence belongs to U^{γ}. Then using Lemma 4.4, it follows that the unique Nash and Stackelberg equilibrium is at $\underline{\mathcal{M}}$. □