Approaches to Congestion Control in Packet Networks

Lefteris Mamatas, Tobias Harks, and Vassilis Tsaoussidis

***Abstract*— We discuss congestion control algorithms, using network awareness as a criterion to categorize different approaches. The first category ("the box is black") consists of a group of algorithms that consider the network as a black box, assuming no knowledge of its state, other than the binary feedback upon congestion. The second category ("the box is grey") groups approaches that use measurements to estimate available bandwidth, level of contention or even the temporary characteristics of congestion. Due to the possibility of wrong estimations and measurements, the network is considered a grey box. The third category ("the box is green") contains the bimodal congestion control, which calculates explicitly the fair-share, as well as the network-assisted control, where the network communicates its state to the transport layer; the box now is becoming green.**

We go beyond a description of the different approaches to discuss the tradeoffs of network parameters, the accuracy of congestion control models and the impact of network and application heterogeneity on congestion itself.

I. Introduction

A network is considered congested when too many packets try to access the same router's buffer, resulting in an amount of packets being dropped. In this state, the load exceeds the network capacity. During congestion, actions need to be taken by both the transmission protocols and the network routers in order to avoid a congestion collapse and furthermore to ensure network stability, throughput efficiency and fair resource allocation to network users. Indeed, during a collapse, only a fraction of the existing bandwidth is utilized by traffic useful for the receiver.

Congestion collapse is considered, in general, as a catastrophic event. However, congestion itself is associated with different properties, depending on the characteristics of the underlying networks, the mechanisms of the transmission protocols, the traffic characteristics of the contenting flows, the level of flow contention, and the functionality of network routers. Therefore, the impact of congestion may be temporary and easily controllable; or it may be catastrophic. Consider, for example, a high speed network which hosts a number of competing flows that increases or decreases. The window of each flow also increases and decreases. However, unlike the traditional networks, the time it takes for the flows to exploit the available bandwidth is certainly longer; the amount of loss upon congestion is certainly higher; and the duration of congestion itself throughout the overall communication time may be relatively smaller.

Since the nature of acceptable congestion cannot be prescribed or even accurately defined in general, congestion control becomes a complex task. Furthermore, complexity increases due to the multipurpose-task of congestion control algorithms. They need to control congestion and avoid collapses, maximize bandwidth utilization, guarantee network stability, and ensure fair resource allocation.

Considering the network as a *black box* that only provides a binary feedback to network flows upon congestion, shifts all the burden to end users and calls for solutions that are more generic and perhaps less responsive. That is, a binary congestion signal does not reflect the particular network state. Each sender operates independently and goals to adjust its rate (or window) in a manner that the total bandwidth of the network will be expended fairly and effectively. From its algorithmic perspective the above problem is challenging because the distributed entities (sources) do not have any prior or present knowledge of the other entities' states; nor do they know the system's capacity and the number of competitors. Hence, the goal of fairness and efficiency appears initially difficult to attain. However, if the system is entitled to a prescribed behavior and the entities agree on common transmission tactics, convergence[1] to fairness becomes feasible [38]. AIMD, the traditional congestion control algorithm of the Internet, operates within that scope: it increases additively the rate of the senders (by a value α) until the system reaches congestion. Upon congestion, all senders decrease their rate multiplicatively using a decrease ratio β.

On the other hand, one can measure network conditions, estimate the available bandwidth or even flow contention, and obtain some knowledge about the network. However, measurements are taken at time-instances which may not necessarily represent current network dynamics, or may not correspond to the overall conditions; consequently, protocols may not manage to accurately estimate the load and predict its duration, resulting in either wrong estimations or wrong recovery strategies. Furthermore, some generic questions cannot really be addressed with certainty: How frequently should we measure the network? How far can we trust our measurements? How responsive should the recovery strategy be? How shall we associate the instantaneous measurements of congestion with the network load over some sufficiently long but also sufficiently recent time period? Consequently, the network may not be a black box but it is certainly not better than *grey*, involving occasionally a considerable risk.

One can go beyond the blind algorithms or the high risk of estimations and actually ask the network[2] for help. Of course, precision comes at some cost. Besides the practical difficulty of layer collaboration and the issue of convincing people to add functionality (and invest money) to their network, the issue of recovery strategies remains. That is, even when the network is really a *green box* (which practically is very difficult), changes

Manuscript received November 13, 2006; revised January 16, 2007.

Lefteris Mamatas and Vassilis Tsaoussidis are with the Department of Electrical and Computer Engineering, Demokritos University of Thrace, Xanthi 67100, Greece (E-mails: {emamatas, vtsaousi}@ee.duth.gr).

Tobias Harks is with the Zuse Institute Berlin (ZIB), Berlin 14195, Germany (E-mail: harks@zib.de).

[1] Convergence to fairness should be perceived in this paper as the procedure which enables different flows that consume different amount of resources each, to balance their resource usage.

[2] An approach to calculate the fair-share without any network support was published in [5]

may be so rapid and unpredictable that our costly and painful effort to obtain some information may go wasted.

Beyond our attempt to provide a categorized description of different congestion control strategies, we also attempt to introduce a manner to characterize them comparatively. Hence, we discuss an evaluation framework, which we use to highlight the advantages of different approaches, exploit the way they handle the tradeoffs of network parameters, and understand their impact on the network itself or the network applications. In order for us to explore the complexity of the issue, we discuss the nature of congestion itself along with other open issues. We devote considerable attention to pricing models. Finally, we discuss the open issues that arise mainly from the diversity of applications, the heterogeneity of internetworks, and the interrelation of network parameters.

More specifically, we define congestion-related terms, goals and metrics in Section II. In Section III we introduce the control models for congestion. In Section IV we discuss end-to-end congestion avoidance and control algorithms that belong in the first category, including the blind AIMD and AIMD-FC algorithms and other generalizations of AIMD. In Section V we detail measurement-based congestion avoidance. In Section VI, we present the explicit calculation of the fair-share along with network-assisted schemes. Having discussed the major approaches, we highlight in Section VII some issues which are yet to be solved. In Section VIII, we conclude our paper.

II. Goals and Metrics

The *congestion window* determines the number of packets that can be outstanding at any time. That is, the number of packets that can be sent without having received the corresponding ACK packets. It is incorporated into the transport layer and controls the number of packets put into the network. The *rate* describes packets per second or bits per second. The window or rate can be dynamically adjusted as the total load on the system changes, however, the former is strictly based on ACKs.

A *cycle* is the phase between two seriate feedbacks of 1 (indicating congestion). Hence, a cycle consists of one decrease step triggered by congestion and a number of additive increase steps. A step describes a single window adjustment in response to a single feedback (either 0 or 1).

The system is in an *equilibrium state*, when resource usage of all flows in a bottleneck is balanced. AIMD-based congestion control algorithms guarantee convergence to equilibrium [11]. In congestion avoidance algorithms this is not always guaranteed.

A non-TCP protocol is called *TCP-friendly* when it yields the same throughput as traditional TCP. TCP-friendly protocols are generally used for multimedia/real-time applications.

Although the sources might discover their fair-share early on, the dynamics of real systems in practice prohibit a straightforward adjustment, but instead, they call for continuous oscillations as a means of discovering the available bandwidth.

Our metrics for the system performance are as follows:

- **Efficiency:** Efficiency is the average flows throughput per step (or per RTT), when the system is in equilibrium.
- **Fairness:** Fairness characterizes the fair distribution of resources between flows in a shared bottleneck link. A well-known metric is [11]:

$$F(x) = \frac{\sum(x_i)^2}{n\sum(x_i^2)} \tag{1}$$

This index is bounded between 0 and 1.

- **Convergence Speed:** Convergence speed describes time passed till the equilibrium state.
- **Smoothness:** Smoothness is reflected by the magnitude of the oscillations during multiplicative decrease. It depends on the oscillations size.
- **Responsiveness:** Responsiveness is measured by the number of steps (or RTTs) to reach an equilibrium (i.e., to equate the windows in order to be in a fair state). The difference between Responsiveness and Convergence Speed is that the former is related to a single flow and the latter to the System.

Goals we set in the evaluation process of a congestion avoidance/control algorithm are:

- To achieve high bandwidth utilization.
- To converge to fairness quickly.
- To minimize the amplitude of oscillations.
- To maintain high responsiveness.
- To coexist fairly and be compatible with traditional widely-used (AIMD based) protocols.

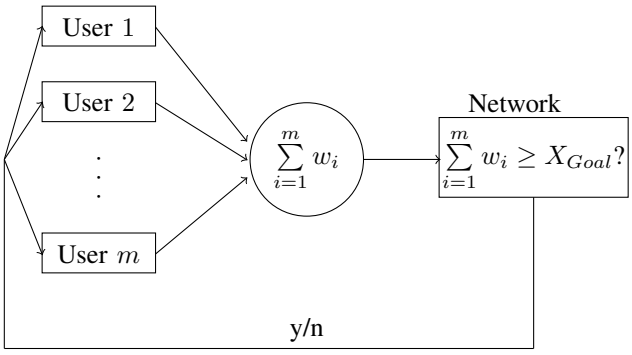


Fig. 1. Synchronous control system model of m users sharing a network.

III. Models for Congestion Control

A. Binary Feedback Models

A synchronous-feedback control system model is shown in Fig. 1. In congestion control literature, the load change is the response to one occurred event (e.g., a packet drop). The synchronous model is characterized by a synchronous generation of responses, in congruity with [11]. The system response is 1 when bandwidth is available and 0 when bandwidth is exhausted. The instruction to the system entities (sources) is to increase or decrease their data rate, respectively. Note that in real networks, the responsive behavior of the system is not administered by any centralized authority with additional knowledge of the network dynamics - it is simply a packet drop due to congestion that naturally happens when bandwidth is exceeded.

To illustrate the concept, consider a system with m users (flows) and instantaneous throughput (window size) for the i_{th} flow, W_i. The system's goal is to operate at an optimal point X_{goal}. Note that this point is not necessarily bandwidth B, since throughput might decrease before we reach B. We

assume that responses are synchronous and consequently the duration of RTTs is common for all flows. Hence, the sources respond uniformly by decreasing their windows in response to a 0 signal; they increase their windows by one in response to a signal of 1 (in case of traditional AIMD).

The limitations of the system are derived from the dynamics of packet networks:

- Bandwidth B is limited.
- Each flow is not aware of the throughput rates (window sizes) of other flows.
- Each flow is not aware of the number of competitors in the channel.
- No flow is aware of the size of bandwidth B.

Although the synchronous model is widely adopted, it is associated with a number of assumptions and/or simplifications, which may not really hold in real networks. Gorinsky et al. in [23] shows that the choice of the model has a direct impact on the results and extends further the model of Chiu and Jain to include different RTTs and consequently asynchronous feedback. MIMD (Multiplicative Increase Multiplicative Decrease), which is not stable in Chiu-Jain model, does converge to fair-states under the more realistic assumption of proportional negative feedback.

In [39] Lahanas and Tsaoussidis describe an asynchronous-feedback model, which corresponds to the diverse round-trip times (RTTs) of competing flows within the same communication channel. In this system, authors show that congestion epoch, which equals[3] to the RTT of a flow times the number of additive increases, is a common knowledge among all competing flows. Based on this relation, they proposed a new algorithm that increases the consumption rate proportionally to the RTT of the flow using a mechanism which adjusts the sizes of the windows of the competing flows at the end of each congestion epoch. Flows with long RTT's are relatively favored (in terms of resources exploitation) because their window is increased faster than in traditional AIMD scheme. The system reaches a window-based equilibrium. This mechanism, named τ-AIMD, uses an extra adjustive component τ along with the additive increase formula of AIMD.

In paper [22], authors compare AIMD and MAIMD (Multiplicative additive increase and multiplicative decrease). They show that the convergence speeds to fair states of AIMD and MAIMD are close to each other. Furthermore, MAIMD has some advantages. For example, its speed to use available network bandwidth can be much faster than AIMD. Authors have also investigated a more realistic asynchronous system model, where round-trip-times differ.

B. Price Based Models

We review a price-based congestion control framework that relies on convex optimization theory. The original model was initially proposed by Kelly et al. in [35] and has been extended in the last few years by several research groups (see [20], [44], [57] and references therein). The crucial assumption of these works is that every link computes a price, based on load measurements and communicates it to the senders using that link. In turn, senders adapt their rates according to the sum of received link prices along the used path.

[3]This is not an exact equality because the RTT is variable [28] and increases with the load of the system. However, the definition gives an intuition of the relation between congestion epoch and the number of additive increases

1) Fluid Flow Model: A network is modeled by a set L of links (resources) with finite capacities $c = (c_l, l \in L)$. The resources are shared by a set S of sources indexed by s. Each source s uses a set $L_s \subset L$ of links. Equivalently, each link is used by a subset $S_l \subset S$ of senders s. The sets L_s or S_l define a routing matrix:

$$R_{ls} = \begin{cases} 1 & \text{if } l \in L_s, \\ 0 & \text{else.} \end{cases} \tag{2}$$

A transmission rate $x_s(t)$ at time t in *packets per second* is associated with each source s. With each link l, a scalar positive congestion-measure $p_l(t)$, called *price*, is associated. In current implementations, the congestion measure or price is based upon information about either loss (Drop-Tail), queuing length (RED) [19], or marks (REM) [4], which requires an ECN [62] bit. Let

$$y_l(t) = \sum_{s \in S} R_{ls} x_s(t - \tau_{ls}^f)$$

be the aggregate flowrate of link l (i.e., the sum over all rates using that link) and let

$$q_s(t) = \sum_{l \in L} R_{ls} p_l(t - \tau_{ls}^b)$$

be the end-to-end congestion measure of source s, where τ_{ls}^f and τ_{ls}^b are the forward- and backward-delay. Note that taking the sum of congestion measures of a used path is essential to maintain the interpretation of $p_l(t)$ as dual variables. Source s can observe its own rate $x_s(t)$ and the end-to-end congestion measure $q_s(t)$ of its path. Link l can observe its local congestion measure $p_l(t)$ and the aggregate flow rate $y_l(t)$. The source rate is adjusted each time period according to a function which depends only on $x_s(t)$ and $q_s(t)$:

$$x_s(t+1) = F_s(x_s(t), q_s(t)). \tag{3}$$

The link congestion measure $p_l(t+1)$ is adjusted based on information about $p_l(t)$ and $y_l(t)$:

$$p_l(t+1) = G_l(y_l(t), p_l(t)). \tag{4}$$

Here F_s models a TCP-like algorithm and G_l an AQM algorithm.

Assume (3) and (4) have a set of equilibria (x^*, p^*). The implicit function theorem yields a relation: $x_s^* = f_s(q_s^*)$, where $f_s(.)$ is a positive, strictly monotone decreasing function (increasing congestion q_s results in decreasing rate x_s). We define the utility-function for each source as:

$$U_s(x_s) = \int f_s^{-1}(x_s) dx_s \ , x_s \geq 0$$

Again f_s^{-1} is assumed to be monotonically decreasing and therefore its integral is monotonically increasing and strictly concave. An increasing utility function implies a greedy source and concavity implies diminishing return. According to Shenker [66] this is exactly the appropriate type of utility for elastic traffic. The problem becomes:

$$\max_{x_s} \sum_{s \in S} U_s(x_s) \quad s.t.\ Rx \leq c \tag{5}$$

An optimal solution exists because the feasible set is compact and the U_i are strictly concave. As the sources are coupled through the shared links, a coordination among all sources using that link is required to solve this problem directly. The key to solving this problem is to regard x_s as primal variables,

p_l as dual variables and to look at the dual problem with Lagrangian $L(x,p)$:

$$\begin{aligned}&\min_{p\geq 0}\max_{x_s\geq 0} L(x,p) = \min_{p\geq 0}\max_{x_s\geq 0}\sum_{s\in S} U_s(x_s) - p(Rx-c)\\ &= \min_{p\geq 0}\max_{x_s\geq 0}\sum_{s\in S}(U_s(x_s) - x_s\sum_{l\in L(s)} p_l) + \sum_{l\in L} p_l c_l\\ &= \min_{p\geq 0}\max_{x_s\geq 0}\sum_{s\in S}(U_s(x_s) - x_s q_s) + \sum_{l\in L} p_l c_l \qquad (6)\end{aligned}$$

With a given vector p^*, the first term of (6) can be divided into S separable subproblems

$$\max_{x_s\geq 0} U_s(x_s) - x_s q_s \qquad (7)$$

which can be solved separately by the sources knowing the congestion measure q_s of the used path. The source law for a sender s solving (7) becomes:

$$x_s^* = \frac{d}{dx_s} U_s^{-1}(q_s). \qquad (8)$$

The term $x_s q_s$ in (7) can be interpreted as the cost or price the network imposes when sending at rate x_s. The role of prices $p_l(t)$ that are Lagrange-Multipliers is to coordinate the actions of individual sources in order to align individual optimality with social optimality (i.e., to solve (5)). The simplest link algorithm to yield equilibrium prices is the gradient-projection algorithm applied to the dual:

$$p_l(t+1) = \begin{cases} p_l(t) + \gamma(y_l(t) - c_l) & \text{if } p_l(t) > 0 \\ p_l(t) + \gamma[y_l(t) - c_l]^+ & \text{if } p_l(t) = 0 \end{cases},$$

where $[z] = max\{0, z\}$. This algorithm can be implemented in a distributed environment, since the information needed at the links is the link bandwidth c_l and the aggregate flow rate $y_l(t)$ which are available. In equilibrium, the prices satisfy the complementary slackness condition (i.e., $p_l(t)$ are zero for non-saturated links and non-zero for bottleneck links). In compact form the gradient projection algorithm representing a source law and an AQM algorithm can be written as:

$$\dot{x_s} = F_s(x_s(t), q_s(t)) = \frac{d}{dx_s} U_s^{-1}(q_s), \quad s \in S \qquad (9)$$

$$\dot{p_l} = G_l(y_l(t), p_l(t)) = \gamma(y_l(t) - c_l), \quad l \in L \qquad (10)$$

In [45] it is shown that this algorithm converges to the unique solution of (5) in an asynchronous environment, where delays can be substantially different and sources and links can update and communicate at different times with different frequencies. In the context of TCP/AQM, the source algorithm F_s is carried out by TCP-Tahoe [28], TCP-Reno [29] and TCP-Vegas [8], and the link algorithm G_l is carried out by Drop-Tail, RED [19] or REM [4]. Various TCP/AQM protocols can be interpreted as different distributed primal-dual algorithms to solve the problem (5) in real time with different utility functions. Thus the optimization framework allows us to predict the equilibrium rates (windows) and the marking- or dropping probabilities of these congestion control schemes in a large network and to assess the *fairness* in a homogeneous network where the same scheme is adopted by all sources, as well as the *friendliness* of different interacting schemes in a heterogeneous network.

2) Fairness Issues, Real-time Traffic, and Nonconvex Utilities: The fundamental assumption in the previous section concerned the choice of utility functions describing elastic traffic. This choice affects the resulting resource allocation in terms of fairness. Following [53], we can associate a class of concave utility functions with corresponding bandwidth fairness-criteria as follows:

$$U_s(x_s, \eta_s) = \begin{cases} -w_s \frac{x^{1-\eta_s}}{1-\eta_s}, \ \eta_s > 0, \ \eta_s \neq 1, \\ w_s \log(x_s), \ \eta_s = 1. \end{cases} \qquad (11)$$

Then, in the case $\eta_s = 1$, we have *weighted proportional* fairness [35]. In the case $\eta_s = 2$, we have *minimum potential delay* fairness, and for $\eta_s \to \infty$, we have *max-min* fairness.

If real-time traffic is considered, the concavity assumption for utility functions does not hold. For instance, the utility function for a VoIP application encoded into a set of layers would be a step function, where the steps correspond to the respective layers.

We can classify the work in this area into two categories. The first category covers approaches that aim at maximizing aggregate (nonconvex) utility subject to capacity constraints: due to the possible duality gap of this type of problem, stable decentralized algorithms are only in special cases to be derived. Lee, Mazumdar and Shroff show in [41] that the canonical distributed algorithms (9) and (10) may fail to converge to a feasible rate allocation and may lead to instability and congestion. To overcome these problems they propose a 'self regulating' heuristic for the special type of sigmoidal utilities in combination with a subgradient method to generate prices. Chiang, Zhang and Hande [10] examine the conditions under which the canonical algorithms converge to the globally optimal rate despite the nonconcavity of utility functions.

In the second category belong approaches that are concerned with a different fairness definition rather than maximizing aggregate utility: an equilibrium point should result in roughly equal utility values for different applications [64], [25], [42], [9]. In [64], only mild assumptions on the feasible utility functions are required (non-decreasing, not necessarily continuous, min. bandwidth exists for a given utility value). The drawbacks of this approach are that the links have to maintain per-flow states in order to allocate bandwidth utility fair, and that there are no stability results given in the presence of communication delay. Cao and Zegura present in [9] a link algorithm that achieves a utility max-min fair bandwidth allocation, where for each link the utility functions of all flows sharing that link is maintained. Hyang and Song [40] present a distributed algorithm without per-flow states that converges to a utility max-min fair operating point. However, they prove stability under bounded communication delay only in the single link case.

More recently, Harks et al. [25], [26] present a distributed utility fair congestion control framework. By transforming possibly non-concave bandwidth utility functions $U_s(\cdot)$ into strictly concave functions $F_s(\cdot)$, they rely on the concave optimization framework described before. The previous fairness characterizations are extended by including a *second order utility function*:

$$F_s(x_s) := \int f_s^{-1}(U_s(x_s))dx_s, \qquad (12)$$

where $f_s(\cdot)$ is a strictly decreasing *transformation* function. If all users $s \in S$ use the same transformation function $f_s =$

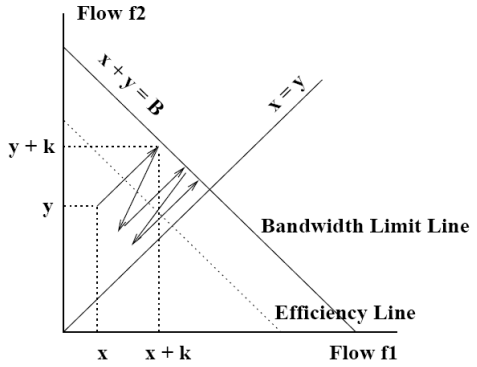


Fig. 2. Vectorial representation of two-flow convergence to fairness. Figure is based on [11].

f, $\forall s \in S$, *utility proportional fairness* as defined in [25] is achieved. If a certain class of transformation functions is used $f_s(q_s, \kappa) = q_s^{-\frac{1}{\kappa}}$, $s \in S$ which leads to

$$F_s(x_s, \kappa) = \int (U_s(x_s)^{-\kappa} dx_s, \tag{13}$$

we get in the limiting case $\kappa \to \infty$ *utility max-min* fairness.

IV. The Box is Black: Blind Congestion Control

The Additive Increase Multiplicative Decrease (AIMD) algorithm is used to implement TCP window adjustments; based on the analysis of Chiu and Jain the algorithm achieves stability and converges to fairness in situations where the demand (of competing flows) exceeds the channel's bandwidth [11].

The congestion control in the traditional TCP, is based on the basic idea of AIMD. In TCP-Tahoe [28], TCP-NewReno [15] and TCP-Sack [50], the additive increase phase is adopted exactly as in AIMD, when the protocols are in the Congestion Avoidance phase. In case of a packet drop, instead of the multiplicative decrease a more conservative tactic is used in TCP-Tahoe. The congestion window resets and the protocol enters again the slow-start phase. On the other hand, in TCP-NewReno and TCP-Sack, when the sender receives 3 DACKs, a multiplicative decrease is used in both window and slow-start threshold. In such case, the protocols remain at the Congestion Avoidance phase. When the retransmission timeout expires, they enter the slow-start phase as in TCP-Tahoe.

A. AIMD-FC

A recent improvement of AIMD, Additive Increase Multiplicative Decrease with Fast Convergence (AIMD-FC) was proposed in [37]. AIMD-FC impacts positively both efficiency and fairness. It is not based on a new algorithm, but rather on an optimization of AIMD during the convergence procedure that enables the algorithm to converge faster and achieve higher efficiency. AIMD-FC increases the bandwidth utilization of AIMD from 3/4 to 5/6.

Authors highlighted the following four observations which were the basis of that work:

1) During the additive increase phase, equal amount of system resources is being allocated to the flows. This amount ('k') is a public or common knowledge (i.e., it is known to every flow in the system).
2) AIMD affects both the initial windows and the amount of system resources (k), that has been fairly allocated, during the multiplicative decrease phase. Note that the manipulation of the initial (and unknown) windows is the real target for achieving fairness.
3) The distance between the Bandwidth Limit Line and the Efficiency Line when the system is in equilibrium depends only on the multiplicative decrease factor [11] (see Figure 2).
4) Two algorithms may need the same number of cycles to converge to fairness: for example, two variants of AIMD with different additive increase rate but the same multiplicative decrease ratio. The number of steps determines the relative efficiency of the algorithm to converge to fairness.

Practically, fairness is achieved in AIMD-FC by releasing (through multiplicative adjustments of the windows) the (unknown to other flows) initial resources of the flows, because during the additive increase phase the flows increase their resource consumption uniformly. So, it is becoming apparent that the distinctive difference of AIMD and AIMD-FC is centered on the portion of the congestion window that is affected by multiplicative decrease (this portion is called decrease window).

Furthermore, there are some open issues related with AIMD-FC:

- The efficiency boundaries of AIMD have not yet been exploited.
- Further modifications can be made in order for AIMD-FC to favor responsiveness or smoothness.

The same authors improved in [37] furthermore the efficiency, smoothness and fairness of AIMD-FC and proposed a new algorithm named AIMD-FC+.

B. Binomial Mechanisms

Bansal and Balakrishnan presented in [7] a new class of nonlinear congestion control algorithms named Binomial Congestion Control Algorithms. These algorithms are called binomial because their control is based on the involvement of two additional algebraic terms with different exponents.

While an AIMD control algorithm may be expressed as:

$$Increase : W_{t+R} \leftarrow W_t + a; a > 0 \tag{14}$$

$$Decrease : W_{t+\delta t} \leftarrow (1 - \beta) W_t; 0 < \beta < 1 \tag{15}$$

authors generalized the AIMD rules in the following way:

$$Increase : W_{t+R} \leftarrow W_t + \frac{a}{W_t^k}; a > 0 \tag{16}$$

$$Decrease : W_{t+\delta t} \leftarrow W_t - \beta W_t^l; 0 < \beta < 1 \tag{17}$$

where "Increase" refers to the increase in window as a result of the receipt of one window of ACKs within a single RTT, "Decrease" refers to the decrease in window upon detection of congestion by the sender, W_t the window size at time t, R the flow's RTT, and a, b, k, l are constants.

For example, for k=0, l=1 we get AIMD. Authors proposed, in the (k, l) space, two AIMD variations (which are also TCP-compatible). IIAD (with k=1 and l=0) and SQRT (with k=1/2 and l=1/2) algorithms. The first is called Inverse Increase Additive Decrease (IIAD) because its increase rule is in inverse proportion to the current window. The second is called SQRT because both its increase is inversely proportional and decrease proportional to the square-root of the current window. Note that a binomial algorithm is TCP-compatible if and only if $k + l = 1$ and $l \leqslant 1$ for suitable α and β.

C. SIMD

Another TCP-friendly nonlinear congestion control algorithm is SIMD [32]. SIMD is the first congestion control algorithm which utilizes history information.

The control rules of SIMD is defined as:

$$Increase : W_{t+R} \leftarrow W_t + a\sqrt{W_t - W_0}; a > 0 \quad (18)$$

$$Decrease : W_{t+\delta t} \leftarrow W_t - \beta W_t; 0 < \beta < 1 \quad (19)$$

where w_0 is the window size after the last decrease and W_t is the continuous approximation of the window size at time t (in RTTs) that elapsed since the window started to increase. Authors show that:

$$w(t) = w_0 + a^2\frac{t^2}{4} \quad (20)$$

SIMD uses the same multiplicative decrease but a different increase rule from those used by AIMD and binomial algorithms; this rule is based on history information.

If two SIMD flows are competing, the flow with the smaller window size is more aggressive due to the nonlinear nature of SIMD. This results in better convergence behavior. Authors show also that SIMD converges faster than memoryless AIMD and binomial controls. Authors evaluated in [32] the TCP-friendliness of SIMD and showed that SIMD can maintain smoothness in steady state.

D. HIGHSPEED-TCP

It is well known that standard TCP does not perform well in high-speed and long-distance networks. Since standard TCP increases its congestion window by one at every round trip time (RTT) and reduces the window by half upon each loss event, it takes more than 83,333 RTT's to fully utilize a 10Gbps link with 1500-byte packet size. This limitation has triggered the proposal of HighSpeed-TCP (HS-TCP) [17]. The idea of HS-TCP is to modify the TCP response function in environments with large Delay-Bandwidth product and increase the congestion window more aggressively upon receiving an ACK and decreases the window more gently upon a loss event.

In [54], [76], however, is shown that HS-TCP does not cope well with fast convergence and RTT fairness. First, when a new HS-TCP flow starts provided there are already large HS-TCP flows active it takes a long time for the new flow to achieve its fair-share. Second, flows with different RTT's consume unfair bandwidth shares.

E. BIC-TCP

To remedy the round trip time unfairness of HS-TCP the Binary Increase Congestion Control Protocol (BIC-TCP) has been proposed in [76]. The BIC-TCP protocol can be characterized by three major concepts: *multiplicative decrease*, *binary search increase* and *additive increase*. Upon a packet loss event, BIC-TCP reduces its window by a multiplicative factor. The window size before the reduction is set to the maximum value and the window size just after the reduction is set to the minimum value. Then, BIC-TCP performs a binary search using these two values by jumping to the midpoint. However, if jumping to the midpoint is too much increase within one RTT, BIC-TCP increments linearly the current window size. If BIC-TCP does not experience packet losses within the updated window size, that window size becomes the new minimum. Otherwise, that window size becomes the new maximum. This process continues until the window increment is less than some small constant and the window is set to the current maximum.

F. Other Generalizations of AIMD

General AIMD Congestion Control (GAIMD) generalizes AIMD congestion control by parameterizing the additive increase value α and multiplicative decrease ratio β. Authors of [77], [13] extended the throughput equation for standard TCP, proposed in [56], to include parameters α, β:

$$T_{\alpha,\beta}(p, RTT, T_0, b) = \frac{1}{RTT\sqrt{\frac{2b(1-\beta)}{\alpha(1+\beta)}p} + T_0 min(1, 3\sqrt{\frac{(1-\beta^2)b}{2\alpha}p})p(1+32p^2)} \quad (21)$$

where p is the loss rate; T_0 is the retransmission timeout value; b is the number of packets acknowledged by each ACK. The overall throughput of TCP-Friendly (α, β) protocols is bounded by the average throughput of standard TCP ($\alpha = 1$, $\beta = 0.5$), which means that equation (23), which is derived from (22) (see [77], [13]) could provide a rough guide to achieve friendliness.

$$T_{a,b}(p, RTT, T_0, b) = T_{1,0.5}(p, RTT, T_0, b) \quad (22)$$

Authors of [77] derive from (1) and (2) a simple relationship for α and β:

$$\alpha = \frac{4(1-\beta^2)}{3} \quad (23)$$

Based on experiments, they proposed a $\beta = 7/8$ as the appropriate value for the reduced the window (i.e., less rapidly than TCP does). For $\beta = 7/8$, (3) gives an increase value $\alpha = 0.31$.

V. The Box is Grey: Measurement-based Congestion Control

Standard TCP relies on packet losses as an implicit congestion signal from overloaded links. However, packet loss is not a sufficient indication of congestion, in its own right, for a number of reasons:

1) Packet loss can be caused by random bit corruption when bandwidth is still available.
2) Acknowledgement-based loss detection at the sender side can be affected by the cross-traffic on the reverse path.
3) Packet loss, as a binary feedback, cannot indicate the level of contention before the occurrence of congestion.

Therefore, an efficient window adjustment tactic should reflect various network conditions, which cannot all be captured simply by packet drops. Several measurement-based transport protocols gather information on current network conditions.

A. TCP-VEGAS

A well-designed, measurement-based congestion avoidance mechanism is TCP-Vegas [8]. TCP-Vegas defines BaseRTT to be the minimum of all measured RTTs, and ExpectedRate to be the ratio of the congestion window to BaseRTT. The sender measures the ActualRate based on the sample RTTs. If the difference between the ExpectedRate and ActualRate is below a lower bound, the congestion window increases linearly during the next RTT; if the difference exceeds an upper bound,

TCP-Vegas decreases the congestion window linearly during the next RTT. According to [8], TCP-Vegas achieves better transmission rates than TCP-Reno and TCP-Tahoe. However, [27] shows that TCP-Vegas can not guarantee fairness or distinguish the nature of errors. From the research perspective of the present work it is important to consider that the authors of TCP-Vegas demonstrated effectively that measurement-based window adjustment is a viable mechanism.

B. FAST-TCP

Fast-TCP is a congestion control protocol that reacts to queuing delay and packet loss [73]. It exhibits the same equilibrium properties as TCP-Vegas but its convergence and stability properties are improved. In Fast-TCP, the congestion window $w_i(t)$ for source i evolves according to:

$$w_i(t+1) = \gamma\left(\frac{d_i w_i(t)}{d_i + q_i(t)} + \alpha_i\right) + \big(1-\gamma\big)w_i(t), \quad (24)$$

where $q_i(t)$ is the measured queuing delay, d_i is the round-trip propagation delay of source i, α_i is a parameter estimating the buffered packets along the used path, and $\gamma \in (0,1)$ is a positive protocol parameter. The equilibrium properties of the above control system, such as throughput, fairness, and delay, can be characterized by the solution of an associated optimization problem (c.f., [35], [46], [44] and references therein. Let $l \in L$ denote the links of a given network with corresponding capacity c_l. If the total round-trip time is denoted by $T_i(t) = d_i + q_i(t)$ than the sending rates are given by $x_i = \frac{w_i(t)}{T_i(t)}$. Let the individual queuing delay of link l be $p_l(t)$. The equilibrium values (w^*, p^*) are given by the solution of:

$$\max_{x\geq 0} \sum_i \alpha_i \log(x_i), \text{ s.t.: } Rx \leq c,$$

where R is the routing matrix and x, c are in vector notation. Using the dual of the above problem and interpreting the individual queuing delays $p_l(t)$ as Langrange multipliers for the corresponding capacity constraint the solution to the above problem is given by:

$$x_i = \frac{\alpha_i}{q_i}. \quad (25)$$

Relation (25) implies that in equilibrium α_i packets are buffered along source i's path. Furthermore, (24) has the appealing property that it is provably *locally* stable in the absence of delay. For a delayed version of (24) only limited results are known: for the single link and bounded delay, (24) is locally stable.

C. TCP-REAL

TCP-Real [78], [72] employs a receiver-oriented and measurement-based congestion control mechanism that significantly improves TCP performance over heterogeneous (wired/wireless) networks and over asymmetric paths. TCP-Real goes beyond the limitation of ack-based binary feedback. It estimates the level of contention and distinguishes the reason of packet losses. TCP-Real relies on:

- Receiver-oriented congestion detection that abrogates the impact of false assessments at the sender due to lost or delayed acknowledgments on a lossy reverse path. The receiver measures the network condition and attaches the results to the ACKs sent back to the sender.
- Measurements based on wave patterns that distinguish the nature of a packet loss (due to congestion or transient wireless errors

A wave [71] consists of a number of fixed-sized data segments sent back-to-back, matching the inherent characteristic of TCP to send packets back-to-back. The receiver computes the data-receiving rate of a wave, which reflects the level of contention at the bottleneck link. If a packet drop is due to a wireless error, the data-receiving rate shall not be affected by the gap of missing packets, since the wave size is published to the receiver. The congestion window is multiplicatively reduced only when a drop is associated with congestion.

D. TCP-WESTWOOD

In TCP-Westwood [48] (TCPW), the sender continuously measures the rate of the connection by monitoring the rate of returning ACKs. Upon three duplicate acknowledgments or timeout, the slow start threshold and the congestion window are set in consistence with the effective bandwidth used at the time packet loss is experienced. No specific mechanism exists to support error classification and the corresponding recovery tactics for wired/wireless networks, albeit the proposed mechanism appears to be effective over symmetric wireless links due to its efficient congestion control. An optimized version of TCP-Westwood is TCP-Westwood+ [49].

E. TFRC

TFRC [24] is a TCP-Friendly, rate-based congestion control protocol, which intends to compete fairly for bandwidth with TCP flows. The sending data rate is adjusted in response to the level of congestion as it is indicated by the loss rate. This adjustment is "gentle"; that is, its instantaneous throughput has, in general, a much lower variation over time, compared with TCP. The smoothing of the transmission gaps makes TFRC suitable indeed for streaming media, telephony or other applications requiring a smooth sending rate. However, smoothness has its own price: the protocol becomes less responsive to bandwidth availability [79].

Furthermore, TFRC is designed for applications that use fixed sized packets. In case of applications with a variance in their packet size (e.g., some audio applications), TRFC's congestion control mechanism cannot be used. A TFRC variant named TFRC-SP (TFRC Small-Packet) can be used instead (see [16]).

TFRC introduces the "loss event" instead of the traditional packet loss. A loss event is defined as one or more lost or marked packets from a window of data (a marked packet refers to a congestion indication from Explicit Congestion Notification). TFRC uses a receiver-based mechanism for the calculation of loss event rate. Such a mechanism is suitable for multicast congestion control and also fits in the case of a large server handling many concurrent requests from clients with more memory and the CPU cycles available. The receiver measures the loss event rate and then passes this information to the sender. The sender calculates its sending rate using a throughput equation that incorporates the loss event rate, round-trip time and packet size.

In summary, TFRC's congestion control mechanism works as follows:

- The receiver measures the loss event rate (based on lost or marked packets from ECN [62], in a single window) and then feeds this information back, to the sender.

- The sender measures the round-trip time (RTT).
- The loss event rate, round-trip time and packet size are used in the throughput calculation function (26). The sender adjusts its sending data rate to match the calculated rate.

$$X = \frac{s}{R \cdot \sqrt{\frac{2 \cdot \beta \cdot p}{3}} + (t\_RTO \cdot (3 \cdot \sqrt{\frac{3 \cdot \beta \cdot p}{8}}) \cdot p \cdot (1 + 32 \cdot p^2))} \quad (26)$$

Where:
X is the transmit rate in bytes/second.
s is the packet size in bytes.
R is the round trip time in seconds.
p is the loss event rate, between 0 and 1.0, of the number of loss events as a fraction of the number of packets transmitted.
t_RTO is the TCP retransmission timeout value in seconds.
β is the number of packets acknowledged by a single TCP acknowledgement.

It may be useful to associate each information fed back to the sender with a statistical figure, which indicates the possibility of a potential wrong decision. Then, based on this data either an aggressive or a conservative strategy can be chosen.

F. TCP-JERSEY

Authors of [75] proposed a new TCP scheme, called TCP-Jersey. They focused on the capability of the transport mechanism to distinguish the wireless from congestion packet losses. TCP-Jersey introduces an Available Bandwidth Estimation (ABE) algorithm and the Congestion Warning (CW) router configuration. ABE continuously estimates the available bandwidth and directs the sender to adjust the transmission rate according to the estimation. The CW-configured-routers mark packets when there is a sign of an incipient congestion to notify the sender, who in turn, classifies errors accordingly.

The performance of the above transport mechanisms is tightly coupled with the robustness of their estimators. Several investigations (such as [30]) have been carried out regarding the accuracy of the proposed estimators.

VI. THE BOX IS GREEN

A. Bimodal Mechanism

Bimodal Congestion Avoidance and Control mechanism [5] measures the fair-share of the total bandwidth that should be allocated for each flow, at any point, during the system's execution. If the fair-share were known, then the sources could avoid congestion by adjusting immediately after the fair-share was discovered, to a new state where the bandwidth allocation of each flow is exactly its fair-share. However, bandwidth availability is not only a matter of channel capacity but is also dependent upon the number of participating flows, and the transmitting behavior of the sources. So, fair-share can be measured only in an equilibrium state.

Authors proposed in [5] a bimodal mechanism, which is based on the idea that upwards and downwards adjustments need to operate in association with the system state. Action is determined based on whether the system is in equilibrium (fair-share is known) or not (fair-share is unknown).

When the fair-share is unknown, the algorithm behaves like AIMD, until two congestion cycles have passed, which is sufficient to recalculate the fair-share. The algorithm then sets the bandwidth allocation for flow f to $(1 - \epsilon)$ [4] times the calculated fair-share, and shifts to known fair-share mode. So, bimodal congestion control algorithm explicitly calculates the fair-share and converges in two congestion cycles to the fair-share. In this mode, the algorithm continues to use additive increase and multiplicative decrease, but the multiplicative decrease factor is α instead of β.

This algorithm is distinguished from the class of TCP-friendly algorithms. TCP-friendly algorithms favor smoothness at the cost of fairness. This algorithm calculates fair-share explicitly, and so fairness is not compromised in this approach. Furthermore, since this algorithm restricts its flows to use only their fair-share, it can be used in conjunction with any other transport protocols (e.g., standard AIMD) without monopolizing for itself most of the available bandwidth. This is in contrast with the TCP-friendly protocols, which attempt to grab all available bandwidth. Of course, this algorithm will not work well in conjunction with other protocols that aggressively (and unfairly) grab a disproportionate share of the bandwidth for their flows.

There are also some open issues related to the bimodal mechanism:

- The more the gain we have in goodput (i.e., by using a smaller ϵ) the less the free space left for incoming flows when contention increases.
- Investigating the optimal value of ϵ in conjunction with the dynamics of specific environments is a subject of future work.
- The modification of the algorithm for the asynchronous scenario by integrating the RTT into the fair-share calculation. For example, bandwidth allocation of a flow can be increased when the RTT of its packets decreases and be decreased when the RTT increases.
- The integration of such ideas with a receiver-oriented feedback approach (like TCP-Real [78]).

B. Network-Assisted Congestion Control

Red Gateways [19] drop packets when congestion is about to happen. RED randomly drops packets, triggering multiplicative decrease in some flows when the length of the queue exceeds a predetermined threshold. RED can function without requiring any change to the current transport level infrastructure.

Ramakrishnan and Floyd in [62] proposed an Explicit Congestion Notification (ECN) to be added to the IP protocol in order to trigger TCP congestion control. Unlike RED, ECN enables routers to probabilistically mark a bit in the IP header, rather than drop the packet, to inform end-hosts of pending congestion when the length of the queue exceeds a threshold. End-hosts multiplicatively reduce their congestion windows upon receiving packets with ECN bit set, before the router buffer overflows and packet drops are inevitable. A duality is served with ECN: TCP performance can be enhanced by means of avoiding losses of data windows due to limited buffer space at the bottleneck router, and congestion collapse can be avoided.

Recent work [52] presents a critical discussion of the performance expectations with RED. An interesting observation about RED and ECN is that they could, somehow, confine future evolution. Imagine a more sophisticated TCP which

[4] where ϵ is a small tuneable parameter

distinguishes between congestion and wireless losses. Since RED drops packets in proportion to sending rates, it is unclear how fair RED would be to the sophisticated TCP which just happens not to unnecessarily back off in case of transient wireless losses [36].

In [18] Floyd and Fall introduced mechanisms based on the identification of high-bandwidth flows from the drop-history of RED. The RED-PD algorithm (RED with Preferential Dropping) [47] uses per-flow preferential dropping mechanisms. Two other approaches that use per-flow preferential dropping with FIFO scheduling are Core-Stateless Fair queuing (CSFQ) [68] and Flow Random Early Detection (FRED) [43]. CSFQ marks packets with an estimate of their current sending rate. The router uses this information in conjunction with the flow's fair-share estimation in order to decide whether a packets needs to be dropped. FRED does maintain a state although only for the flows which have packets in the queue. The flows with many buffered packets are having an increased dropping probability.

The CHOKe mechanism [59] matches every incoming packet against a random packet in the queue. If they belong to the same flow, both packets are dropped. Otherwise, the incoming packet is admitted with a certain probability. However, a high-bandwidth flow may have only a few packets in the queue. Authors in [1] show that a minor modification to the CHOKe active queue management policy ensures efficient operation as well as reasonable fairness at Nash equilibrium.

Authors of [58] continued the CSFQ and CHOKe [59] approaches. Their proposed mechanism keeps a sample of arriving traffic. A flow with several packets in the sample, has an increased dropping probability. The Stochastic Fair Blue (SFB) [12] uses multiple levels of hashing in order to identify high-bandwidth flows. As the authors state, their mechanism works well only with a few high-bandwidth flows. Anjum and Tassiulas proposed in [3] a mechanism that drops packets based on the buffer occupancy of the flow while ERUF [63] uses source quench to have undeliverable packets dropped at the edge routers. On the other hand, SRED [55] caches the recent flows in order to determine the high-bandwidth flows.

A network-assisted proposal relevant to Bimodal Congestion Avoidance and Control mechanism [5] is Multimodal Control Protocol (MCP) [60]. MCP design relies on three key principles: multiple modes of operation, sufficiently high fairness after a relatively short sequence of transmission adjustments, and constant-rate transmission in the steady state. One of MCP contributions is its mechanism for limited explicit communication between hosts and routers. The mechanism enables the sender of a flow to urge all flows sharing its bottleneck links to operate in a fairing mode for seven increase-decrease cycles of AIMD(80 kbps; 0.5) control.

C. VCP, XCP and JETMAX

The variable-structure congestion control protocol (VCP) requires explicit feedback from the network about the state of congestion (see [74] for a detailed description). It relies on the two ECN bits that are available in current implementations of the IP header. Each router in the network calculates a *load factor* that is used to classify the level of congestion into three regions: low-load, high-load, and overload. Routers are responsible to encode the current state into the ECN bits. Upon receiving the congestion information that corresponds to one of the three regions, the sender performs multiplicative increase (low load), additive increase (high load), and multiplicative decrease (overload), respectively.

The explicit congestion control protocol (XCP) can be seen as a generalization of the ECN proposal [34]. Instead of one or two bits congestion notification, an XCP capable router communicates the explicit bandwidth share back to the senders. This is done via two control functionalities: fairness and efficiency controls. Efficiency control aims at maximizing link utilization while fairness control aims at enforcing fair bandwidth sharing between competing flows. The main drawback of XCP, however, is its unstable behavior in a multilink environment, where senders receive feedback on different time scales, that is, the feedback is heterogeneously delayed. In this case, the bottleneck may oscillatory switch between routers and the senders are not able to timely detect the most congested resource.

To address this issue, JetMax has been proposed in [80]. This protocol relies on single-router feedback (as opposite to additive feedback approaches) and achieves max-min fair rate allocation in its steady state. The feedback is provided in form of changes to the packet loss. Each sender is assigned a bottleneck router which is the router with the highest packet loss value p on the path. p is periodically computed by each router l applying the equation:

$$p_l(n) = \frac{y_l(n) - \gamma_l C_l}{y_l(n)} \tag{27}$$

where $y_l(n)$ is the total load observed by router l at the time instance n, $\gamma_l \in (0, 1]$ is the desired link utilization and C_l the link capacity.

Within each router, the passing flows are divided into two categories (the bottlenecked and any additional flow). The authors call the first category of flows responsive and the second category unresponsive. Each router l periodically computes the fair rate of the responsive flows $g_l(n)$ applying the equation:

$$g_l(n) = \frac{\gamma_l C_l - u_l(n)}{N_l(n)} \tag{28}$$

where $u_l(n)$ is the aggregate rate of the unresponsive flows and $N_l(n)$ is the estimated number of responsive flows and inserts this value into the header of the data packets of the responsive flows. Each sender r updates its sending rate according to the equation:

$$x_r(n) = x_r\left(n - D_r\right) - \tau\left[x_r\left(n - D_r\right) - g_l\left(n - D_r^{\leftarrow}\right)\right] \tag{29}$$

where $D_r = D_r^{\rightarrow} + D_r^{\leftarrow}$ is the RTT $D_r^{\rightarrow}$ the forward delay between the sender and the bottleneck router and $D_r^{\leftarrow}$ the backward delay between the bottleneck router and the sender. $\tau \in (0, 1]$ is the gain parameter.

As stated in [80] the features of JetMax are:

1) Monotonic convergence to stationarity.
2) Capacity-independent convergence time (same number of RTT steps).
3) Zero packet loss both in the transient and steady-state.
4) Tuneable link utilization.
5) RTT-independent max-min fairness.
6) Global multi-link stability under consistent bottleneck assignment for all types of delay.

Furthermore, JetMax is supposed to overcome the problem of bottleneck oscillations, which leads to instability of XCP in networks with heterogeneous delays. This is achieved through monotonic and continuous behaviour of the feedback function

p for bottleneck switches, which leads to consistent choice of the bottleneck router.

VII. Discussion

The question of efficiency is associated with the utilized bandwidth; at a first glance, the system dynamics suggest that the higher the oscillation the less the efficiency. It appears[5] that the higher the oscillation, the faster we approach fairness. In [14], [77], [7], [32], authors attempt to take advantage of this property. More precisely, it has been observed that streaming applications could benefit from modest oscillations since these reflect the smoothness of adjusting the transmission rate backwards. Such protocols are characterized as TCP-Friendly because they consume the same amount of bandwidth as TCP(1, 0.5) does [56]. Theoretically, and in the context of Figure 2, these algorithms try to push the Efficiency Line closer to the Bandwidth Limit Line at the expense of convergence speed to fairness. For example, the GAIMD [77] algorithm with $\beta = \frac{7}{8}$, converges to fairness in $O(Blog_{1.14}B)$ steps, where B is the link bandwidth; the SIMD [32] with $\beta = \frac{15}{16}$ converges in $O(Blog_{1.06}B)$ steps, the SQRT [7] algorithm with $\beta = 1 - \sqrt{\frac{1}{B}}$, converges in $O(B^2)$; so does the IIAD [7]. The performance of AIMD algorithm (with parameters $\alpha = 1$ and $\beta = \frac{1}{2}$) and in general of TCP is studied in [51], [18]. The efficiency of the AIMD algorithm is described in [51] by the formula:

$$Efficiency = \frac{3}{4}B\frac{MSS}{RTT} \tag{30}$$

where B is the link bandwidth, MSS is the TCP packet size and RTT is Round Trip Time. The analysis suggests a 75% efficiency of the protocol when the system is in equilibrium. Several congestion control algorithms along with a cost analysis of their capacity to discover the available bandwidth have been presented in [33].

Although the system efficiency is well defined and adequately measured, the algorithm's potential to achieve fairness need further attention. From our perspective, fairness involves a *punctual*, a *vertical* and a *horizontal* aspect. The *punctual* represents the question whether the algorithm converges or not. The *horizontal* involves the issue whether the speed to convergence is optimal and is frequently called *responsiveness*. The *vertical* involves the question whether the level of window oscillations is high during the convergence procedure. This is commonly called *smoothness*. Note that this aspect can overlap with system efficiency during convergence; however, an example of two flows that over and under-utilize their fair-share, respectively, but do not leave bandwidth unexploited suffices to demonstrate the distinctive difference: our system in this case is efficient but not fair.

Furthermore, there was recently a significant effort to distinguish congestion-related errors from wireless errors. That effort invested mainly in distinguishing the locus of the error [6], [67], [21], rather than focusing on the dynamics from the combination of both errors. For example, persistent congestion which may be experienced in some router in a wired network may change to a more transient one when wireless errors are introduced at the last mile, where some wireless receivers reside. Furthermore, higher contention can be tolerated under the same circumstances of congestion. In a similar context, in a high-speed network congestion may cause more losses due to the fact that congestion windows may grow to very high values; however, it will last less and potentially, it will appear much less frequently [2], [17], [31], [34], [65]. The dynamics of combined wired and wireless errors appears to be far more important issue than the ability to geographically locate the error and apply the well-known techniques. Transient congestion combined with higher contention may not call for conservative recovery as this is implemented through the extension of the timeout and the shrinkage of the congestion window.

What mechanisms are appropriate to detect the presence of combined congestion and transient random wireless errors? A fairly recent probing scheme appears to measure network performance but also combines the ability to deal with wireless errors: it freezes the timeout and holds still the congestion window without transmitting any data when wireless errors do not allow the probing mechanism to be completed [70]. Other mechanisms measure contention and decouple wireless errors from others. With what precision can we estimate really network conditions? How can we take into account the risk of wrong estimation in order to avoid false recovery strategies. Or, in another context, what are the situations which may tolerate some risk? And, beyond detection, what strategies correspond to each distinctive new class of errors that are detected? How can we evaluate them? For example, authors in [69] have shown that congestion control determines the aggressive/conservative protocol behavior, which in turn affects energy consumption. This issue calls certainly for further investigation.

The TCP-related work discussed above raises another important question: where is the right place to add the required functionality? This question does not have a clear answer; error control is not exclusively a management property of the router or the base-station, nor is it exclusively assigned to the transport layer. A widely accepted approach, presented by the end-to-end argument [18], states that we can only implement a function at a lower layer, if that layer can perform the complete function. Since the lower level cannot have enough information about the application's requirements, protocol parameters, and device constraints, it cannot implement the whole function of error control; it can only be used to optimize the function of the higher layer.

One can go beyond architectural optimization criteria and worry about potential false strategies due to the locality of router decisions. For example, when a router experiences congestion or progresses towards congestion, it takes action. The action is taken in association with an inherent assumption: that the portion of the network that follows will not change the major traffic characteristics of the flows - something not necessarily true when a wireless network is deployed at the receivers' end. Certainly TCP throughput allows for assuming a monotonic behavior. However, when our assumption is violated, there is no indication that our strategy is still correct.

Departing from the same point, we justified earlier the logic behind measurement-based protocols. There is a significant difference however; the receivers have indeed the capability to measure flow traffic throughout the whole network path, hence they do not fall into wrong assessments due to limited view of the network. Perhaps one can trust end-to-end protocols on that aspect, more than network mechanisms.

By the same token, when many flows compete for a limited bandwidth, a window back-off strategy will not yield any significant gain. For example, a large number of flows over a

[5]Both statements have been made initially in [11]

10Mbps network would probably operate with single-packet windows that do not permit further shrinkage. Obviously, under such contention of packets, efficiency of the algorithm does not really become an issue. Only a proper timeout adjustment can permit all flows to use the network fairly. Authors in [61] investigate the properties of timeout when it becomes the schedule for the link.

Evaluation of congestion control is an important issue in its own right. Moreover, it lacks appropriate evaluation methodologies. It is very common that congestion control mechanisms are evaluated only in terms of channel exploitation. Our set of metrics combined (defined in Section II) capture a more accurate view of the protocol behavior. Furthermore, application-oriented performance metrics (e.g., jitter) can show whether protocol performance gains are reflected on user's satisfaction.

VIII. Conclusions

To summarize, we classified different approaches based on the potential knowledge of network dynamics during communication. Some approaches assume zero knowledge of such dynamics until the cliff point; these are classified as blind in the literature. That is, the network hides all its secrets in a 'black' box. Some measurements and bandwidth estimation techniques may reveal some network characteristics; however, they lack at least precision occasionally and are therefore dubious; the box is consider grey. In contrast, when the network reports explicit information about its state, information is not hidden and, in addition, is accurate indeed; that is, the box is green. In another context, the color may represent whether a corresponding action can be taken with certainty, based on estimated dynamics. Classification is also possible using other criteria. For example, protocols may be classified based on the transmission scheme they are using (e.g., rate-based window-based) or based on their proactive or reactive strategy (e.g., congestion avoidance versus control). We have also described modeling approaches to congestion control, pricing models and discussed issues that remain to be solved.

References

[1] A. Akella, S. Seshan, R. Karp, S. Shenker, and C. Papadimitriou, "Selfish Behavior and Stability of the Internet: A Game-Theoretic Analysis of TCP", in *Proc. ACM SIGCOMM 2002*, Pittsburgh, USA, August 2002.

[2] I. Akyildiz, G. Morabito, and S. Palazzo, "TCP Peach: A New Congestion Control Scheme for Satellite IP Networks", *IEEE/ACM Transactions on Networking*, vol. 9, no. 3, June 2001, pp. 307–321.

[3] F. M. Anjum and L. Tassiulas, "Fair Bandwidth Sharing among Adaptive and Non-Adaptive Flows in the Internet", in *Proc. IEEE INFOCOM 99*, New York, USA, March 1999.

[4] S. Athuraliya, V. H. Li, S. H. Low, and Q. Yin, "REM: Active queue management", *IEEE Network*, vol. 15, no. 3, May–Jun. 2001, pp. 48 – 53.

[5] P. C. Attie, A. Lahanas, and V. Tsaoussidis, "Beyond AIMD: Explicit fair-share calculation", in *Proc. ISCC 2003*, Antalya, Turkey, June 2003.

[6] H. Balakrishnan, S. Seshan, E. Amir, and R. H. Katz, "Improving TCP/IP Performance over Wireless Networks", in *Proc. ACM Mobicom '95*, Berkeley, USA, November 1995.

[7] D. Bansal and H. Balakrishnan, "Binomial Congestion Control Algorithms", in *Proc. IEEE INFOCOM'01*, Anchorage, Alaska, USA, April 2001.

[8] L. Brakmo and L. Peterson, "TCP Vegas: End-to-End Congestion Avoidance on a Global Internet", *IEEE Journal on Selected Areas of Communications*, vol. 13, no. 8, October 1995, pp. 1465–1480.

[9] Z. Cao and E. Zegura, "Utility Max-Min: An Application-Oriented Bandwidth Allocation Scheme", in *Proc. IEEE INFOCOM*, New York, USA, March 1999.

[10] M. Chiang, S. Zhang, and P. Hande, "Distributed Rate Allocation for Inelastic Flows: Optimization Frameworks, Optimality Conditions, and Optimal Algorithms", in *Proc. IEEE INFOCOM*, Miami, USA, March 2005.

[11] D. Chiu and R. Jain, "Analysis of the Increase/Decrease Algorithms for Congestion Avoidance in Computer Networks", *Journal of Computer Networks and ISDN*, vol. 17, no. 1, June 1989, pp. 1–14.

[12] W. Feng, D. Kandlur, D. Saha, and K. G. Shin, "BLUE: A New Class of Active Queue Management Algorithms", University of Michigan, Tech. Rep. CSE-TR-387-99, April 1999.

[13] S. Floyd, M. Handley, and J. Padhye, "A Comparison of Equation-Based and AIMD Congestion Control", May 2000, Available: *http://www.aciri.org/tfrc/*.

[14] S. Floyd, M. Handley, J. Padhye, and J. Widmer, "Equation-Based Congestion Control for Unicast Applications", in *Proc. ACM SIGCOMM 2000*, Stockholm, Sweden, May 2000.

[15] S. Floyd and T. Henderson, "The New-Reno Modification to TCP's Fast Recovery Algorithm", *RFC 2582*, April 1999.

[16] S. Floyd and E. Kohler, "TCP Friendly Rate Control (TFRC): the Small-Packet (SP) Variant", Internet Draft, March 2006.

[17] S. Floyd, "HighSpeed TCP for Large Congestion Windows", *RFC 3649*, December 2003.

[18] S. Floyd and K. Fall, "Promoting the Use of End-to-End Congestion Control in the Internet", *IEEE/ACM Transactions on Networking*, vol. 7, no. 4, 1999, pp. 458–472.

[19] S. Floyd and V. Jacobson, "Random Early Detection Gateways for Congestion Avoidance", *IEEE/ACM Transactions on Networking*, vol. 1, no. 4, 1993, pp. 397–413.

[20] R. Gibbens and F. Kelly, "Resource Pricing and the Evolution of Congestion Control", *Automatica*, vol. 35, 1999, pp. 1969–1985.

[21] T. Goff, J. Moronski, and D. Phatak, "Freeze-TCP: A True End-to-End Enhancement Mechanism for Mobile Environments", in *Proc. INFOCOM 2000*, Tel-Aviv, Israel, March 2000.

[22] S. Gorinsky and H. Vin, "Additive Increase Appears Inferior", Department of Computer Sciences, University of Texas at Austin, Tech. Rep. TR2000-18, May 2000, Available: *http://www.arl.wustl.edu/~gorinsky/pdf/tr2000-18.pdf*.

[23] S. Gorinsky and H. Vin, "Extended Analysis of Binary Adjustment Algorithms", Department of Computer Sciences, University of Texas at Austin, Tech. Rep. TR2002-39, August 2002, Available: *www.arl.wustl.edu/~gorinsky/pdf/TR2002-39.pdf*.

[24] M. Handley, S. Floyd, J. Padhye, and J. Widmer, "TCP Friendly Rate Control (TFRC): Protocol Specification", *RFC 3448*, January 2003.

[25] T. Harks, "Utility Proportional Fair Resource Allocation: An Optimization Oriented Approach", in *Proc. QoS in Multiservice IP Networks*, Catania, Italy, February 2005.

[26] T. Harks and T. Poschwatta, "Priority Pricing in Utility Fair Networks", in *Proc. IEEE International Conference on Network Protocols (ICNP)*, Boston, USA, November 2005.

[27] U. Hengartner, J. Bolliger, and T. Cross, "TCP Vegas Revisited", in *Proc. IEEE INFOCOM 2000*, Tel-Aviv, Israel, March 2000.

[28] V. Jacobson, "Congestion Avoidance and Control", in *Proc. ACM SIGCOMM '88*, Stanford, USA, August 1988.

[29] V. Jacobson, "Modified TCP Congestion Avoidance Algorithm", *Message to end2end-interest mailing list*, April, 1990.

[30] M. Jain and C. Dovrolis, "Ten Fallacies and Pitfalls on End-to-End Available Bandwidth Estimation", in *Proc. Internet Measurement Conference 2004*, Taormina, Italy, October 2004.

[31] C. Jin, D. Wei, and S. Low, "Fast TCP: motivation, architecture, algorithms, performance", in *Proc. IEEE INFOCOM*, Hong Kong, China, March 2004.

[32] S. Jin, L. Guo, I. Matta, and A. Bestavros, "TCP-friendly SIMD Congestion Control and Its Convergence Behavior", in *Proc. ICNP'2001*, Riverside, USA, November 2001.

[33] R. Karp, E. Koutsoupias, C. Papadimitriou, and S. Shenker, "Optimization Problems in Congestion Control", in *Proc. IEEE Symposium on Foundations of Computer Science*, Redondo Beach, USA, November 2000.

[34] D. Katabi, M. Handley, and C. Rohrs, "Congestion Control for High Bandwidth-Delay Product Networks", in *Proc. ACM SIGCOMM 2002*, Pittsburgh, USA, August 2002.

[35] F. P. Kelly, A. K. Maulloo, and D. K. H. Tan, "Rate Control in Communication Networks: Shadow Prices, Proportional Fairness, and Stability", *Journal of the Operational Research Society*, vol. 49, 1998, pp. 237–52.

[36] M. Khanna, C. Zhang, and V. Tsaoussidis, "Experimental Evaluation of RED in Heterogeneous Environments", in *Proc. 3rd International Conference on Internet Computing (IC)*, Las Vegas, USA, June 2002.

[37] A. Lahanas and V. Tsaoussidis, "Additive Increase Multiplicative Decrease - Fast Convergence (AIMD-FC)", in *Proc. Networks 2002*, Atlanta, USA, August 2002.

[38] A. Lahanas and V. Tsaoussidis, "Exploiting the Efficiency and Fairness Potential of AIMD-based Congestion Avoidance and Control", *Computer Networks, Elsevier*, vol. 43, no. 2, October 2003, pp. 227–245.

[39] A. Lahanas and V. Tsaoussidis, "τ-AIMD for Asynchronous Receiver Feedback", in *Proc. IEEE ISCC 2003*, Antalya, Turkey, June 2003.

[40] H.-W. Lee and S. Chong, "A Distributed Utility Max-Min Flow Control Algorithm", *Computer Networks*, vol. 50, no. 11, August 2006, pp. 1816 – 1830.

[41] J. W. Lee, R. R. Mazumdar, and N. B. Shroff, "Non-convex Optimization and Rate Control for Multi-class Services in the Internet", *IEEE/ACM Transactions on Networking*, vol. 13, no. 4, August 2005.

[42] R. F. Liao and T. Campbell, "A Utility-Based Approach for Quantitative Adaption in Wireless Packet Networks", *Wireless Networks*, vol. 7, no. 5, 2001, pp. 541–557.

[43] D. Lin and R. Morris, "Dynamics of Random Early Detection", in *Proc. ACM SIGCOMM '97*, Cannes, France, September 1997.

[44] S. H. Low and D. E. Lapsley, "Optimization Flow Control I", *IEEE/ACM Transactions on Networking*, vol. 7, no. 6, 1999, pp. 861–874.

[45] S. Low, F. Paganini, and J. C. Doyle, "Internet Congestion Control", *IEEE Control Systems Magazine*, vol. 22, 2002.

[46] S. Low, L. Peterson, and L. Wang, "Understanding Vegas: a Duality Model", *Journal of ACM*, vol. 49, no. 2, March 2002, pp. 207–235.

[47] R. Mahajan, S. Floyd, and D. Wetherall, "Controlling High-Bandwidth Flows at the Congested Router", in *Proc. IEEE ICNP '01*, Riverside, USA, November 2001.

[48] S. Mascolo, C. Casetti, M. Gerla, M. Sanadidi, and R. Wang, "TCP Westwood: Bandwidth Estimation for Enhanced Transport over Wireless Links", in *Proc. ACM MobiCom'01*, Rome, Italy, July 2001.

[49] S. Mascolo, L. A. Grieco, R. Ferorelli, P. Camarda, and G. Piscitelli, "Performance Evaluation of Westwood+ TCP Congestion Control", in *Proc. Internet Performance Symposium (IPS 2002)*, Taipei, Taiwan, November 2002.

[50] M. Mathis, J. Mahdavi, S. Floyd, and A. Romanow, "TCP Selective Acknowledgement Options", *RFC 2018*, April 1996.

[51] M. Mathis, J. Semke, J. Mahdavi, and T. Ott, "The Macroscopic Behavior of the TCP Congestion Avoidance Algorithm", *Computer Communication Review*, vol. 27, no. 3, July 1997, pp. 67–82.

[52] M. May, T. Bonald, and J. Bolot, "Analytic Evaluation of RED Performance", in *Proc. IEEE INFOCOM 2000*, Tel-Aviv, Israel, March 2000.

[53] J. Mo and J. Walrand, "Fair End-to-End Window-Based Congestion Control", *IEEE/ACM Transactions on Networking*, vol. 8, no. 5, October 2000, pp. 556–567.

[54] M. Nabeshima and K. Yata, "Improving the Convergence Time of HighSpeed TCP", in *Proc. IEEE International Conference on Networks (ICON 2004)*, Singapore, November 2004.

[55] T. J. Ott, T. V. Lakshman, and L. H. Wong, "SRED: Stabilized RED", in *Proc. IEEE INFOCOM*, New York, USA, March 1999.

[56] J. Padhye, V. Firoiu, D. Towsley, and J. Kurose, "Modeling TCP Throughput: A Simple Model and its Empirical Validation", in *Proc. ACM SIGCOMM*, Vancouver, Canada, August 1998.

[57] F. Paganini, Z. Wang, J. Doyle, and S. Low, "Congestion Control for High Performance, Stability and Fairness in General Networks", *IEEE/ACM Transactions on Networking*, vol. 13, no. 1, February 2005, pp. 43–56.

[58] R. Pan, L. Breslau, B. Prabhakar, and S. Shenker, "Approximate Fairness through Differential Dropping", *ACM SIGCOMM Computer Communication Review*, vol. 33, no. 2, April 2003, pp. 23–39.

[59] R. Pan, B. Prabhakar, and K. Psounis, "CHOKe - A Stateless Queue Management Scheme for Approximating Fair Bandwidth Allocation", in *Proc. IEEE INFOCOM 2000*, Tel-Aviv, Israel, March 2000.

[60] M. Podlesny and S. Gorinsky, "Multimodal Congestion Control for Low Stable-State Queuing", in *Proc. IEEE INFOCOM 2007 Minisymposium*, Anchorage, Alaska, USA, May 2007, Available: *http://www.arl.wustl.edu/~gorinsky/pdf/low-queuing.pdf*.

[61] I. Psaras and V. Tsaoussidis, "Why TCP Timers (still) Don't Work Well", *Computer Networks, Elsevier*, 2007, in press.

[62] K. Ramakrishnan and S. Floyd, "A Proposal to Add Explicit Congestion Notification (ECN) to IP", *RFC 2481*, January 1999.

[63] A. Rangarajan, "Early Regulation of Unresponsive Flows", University of California at Santa Barbara, Tech. Rep. TRCS99-26, July 1999.

[64] S. Sarkar and L. Tassiulas, "Fair Allocation of Utilities in Multirate Multicast Networks: A Framework for Unifying Diverse Fairness Objectives", *IEEE Transactions on Automatic Control*, vol. 47, no. 6, June 2002, pp. 931–944.

[65] S. Shalunov, "TCP Armonk (tcpar)", Tech. Rep., September 2002, Available: http://www.internet2.edu/ shalunov/tcpar.

[66] S. Shenker, "Fundamental Design Issues for the Future Internet", *IEEE Journal on Selected Areas in Communications*, vol. 13, 1995, pp. 1176–88.

[67] P. Sinha, T. Nandagopal, N. Venkitaraman, R. Sivakumar, and V. Bharghavan, "WTCP: A Reliable Transport Protocol for Wireless Wide-Area Networks", *Wireless Networks*, vol. 8, no. 2-3, 2002, pp. 301–316.

[68] I. Stoica, S. Shenker, and H. Zhang, "Core-Stateless Fair Queueing: Achieving Approximately Fair Bandwidth Allocations in High Speed Networks", in *Proc. ACM SIGCOMM*, Vancouver, Canada, August 1998.

[69] V. Tsaoussidis, H. Badr, X. Ge, and K. Pentikousis, "Energy / Throughput Tradeoffs of TCP Error Control Strategies", in *Proc. IEEE ISCC*, Antibes, France, July 2000.

[70] V. Tsaoussidis and A. Lahanas, "Exploiting the Adaptive Properties of a Probing Device for TCP in Heterogeneous Networks", *Journal of Computer Communications*, vol. 26, no. 2, February 2003, pp. 177–192.

[71] V. Tsaoussidis, A. Lahanas, and C. Zhang, "The Wave and Probe Communication Mechanisms", *The Journal of Supercomputing*, Kluwer Academic Publishers, vol. 20, no. 2, September 2001.

[72] V. Tsaoussidis and C. Zhang, "TCP-Real: Receiver-Oriented Congestion Control", *Computer Networks Journal, Elsevier*, vol. 40, no. 4, November 2002, pp. 477-497.

[73] D. X. Wei, C. Jin, S. Low, and S. Hedge, "FAST TCP: Motivation, Architecture, Algorithms, Performance", *IEEE/ACM Transactions on Networking*, 2007, in press.

[74] Y. Xia, L. Subramanian, I. Stoica, and S. Kalyanaraman, "One More Bit is Enough", *SIGCOMM Computer Communications Review*, vol. 35, no. 4, 2005, pp. 37–48.

[75] K. Xu, Y. Tian, and N. Ansari, "TCP-Jersey for Wireless IP Communications", *IEEE Journal on Selected Areas of Communications*, vol. 22, no. 4, May 2004, pp. 747–756.

[76] L. Xu, K. Harfoush, and I. Rhee, "Binary Increase Congestion Control for Fast Long-Distance Networks", in *Proc. IEEE INFOCOM*, Hong Kong, China, March 2004.

[77] Y. Yang and S. Lam, "General AIMD Congestion Control", in *Proc. IEEE International Conference on Network Protocols (ICNP)*, Osaka, Japan, November 2000.

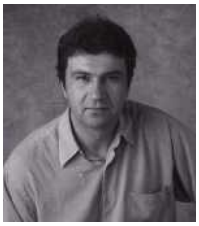
[78] C. Zhang and V. Tsaoussidis, "TCP-Real: Improving Real-time Capabilities of TCP over Heterogeneous Networks", in *Proc. IEEE/ACM NOSSDAV*, Port Jefferson, New York, USA, June 2001.

[79] C. Zhang and V. Tsaoussidis, "The Interrelation of TCP Responsiveness and Smoothness", in *Proc. IEEE ISCC*, Taormina, Italy, July 2002.

[80] Y. Zhang, D. Leonard, and D. Loguinov, "JetMax: Scalable Max-Min Congestion Control for High-Speed Heterogeneous Networks", in *Proc. IEEE INFOCOM*, Barcelona, Spain, April 2006.

Lefteris Mamatas received a diploma in Electrical and Computer Engineering from Demokritos University of Thrace, Greece in 2003, where he is currently a Ph.D. candidate. Lefteris worked as a research intern at DoCoMo Eurolabs (May-October 2005 and July-September 2006). His research interests lie in the area of transport protocols and their behavior over heterogeneous networks. He published 10 conference and 3 journal papers.

Tobias Harks received the diploma degree in mathematics from University of Muenster, Germany, in 2003. He is currently working towards the Ph.D. degree at the Technical University Berlin and Zuse Institute Berlin. His research interests include Combinatorial Optimization, Algorithmic Game Theory, Online Optimization, and Network Management.



Vassilis Tsaoussidis received a B.Sc in Applied Mathematics from Aristotle University, Greece; a Diploma in Statistics and Computer Science from the Hellenic Institute of Statistics; and a Ph.D in Computer Networks from Humboldt University, Berlin, Germany (1995). Vassilis held faculty positions in Rutgers University, New Brunswick, SUNY Stony Brook and Northeastern University, Boston. In May 2003, Vassilis joined the Department of Electrical and Computer Engineering of Demokritos University, Greece. His research interests lie in the area of transport/network protocols, i.e., their design aspects and performance evaluation. Vassilis is an editor-in-chief in the Journal of Internet Engineering. He is an editor for IEEE Transactions in Mobile Computing, the Journal of Computer Networks (Elsevier), the journal of Wireless Communications and Mobile Computing and the journal of Mobile Multimedia. He participates in several Technical Program Committees in his area of expertise, such as INFOCOM, NETWORKING, GLOBECOM and several others.