Machine-learnt versus analytical models of TCP throughput

I. El Khayat, P. Geurts, G. Leduc [*]

University of Liège, Department of Electrical Engineering and Computer Science, Institut Montefiore, Sart Tilman B28, B4000, Liège, Belgique

Abstract

We first study the accuracy of two well-known analytical models of the average throughput of long-term TCP flows, namely the so-called SQRT and PFTK models, and show that these models are far from being accurate in general. Our simulations, based on a large set of long-term TCP sessions, show that 70% of their predictions exceed the boundaries of TCP-Friendliness, thus questioning their use in the design of new TCP-Friendly transport protocols. We then investigate the reasons of this inaccuracy, and show that it is largely due to the lack of discrimination between the two packet loss detection methods used by TCP, namely by triple duplicate acknowledgments or by timeout expirations. We then apply various machine learning techniques to infer new models of the average TCP throughput. We show that they are more accurate than the SQRT and PFTK models, even without the above discrimination, and are further improved when we allow the machine-learnt models to distinguish the two loss detection techniques. Although our models are not analytical formulas, they can be plugged in transport protocols to make them TCP friendly. Our results also suggest that analytical models of the TCP throughput should certainly benefit from the incorporation of the timeout loss rate.

Key words: TCP, throughput models, machine learning

1 Introduction and motivation

TCP is a transport protocol widely used by applications like remote access (ssh, telnet), file transfer (ftp), and Peer-to-Peer. It occupies more than 90% of

[*] Corresponding author.
Email addresses: elkhayat@run.montefiore.ulg.ac.be (I. El Khayat), p.geurts@ulg.ac.be (P. Geurts), guy.leduc@ulg.ac.be (G. Leduc).

Internet resources [1]. The success of this protocol lies in the reliable transfer it offers. To avoid network collapse, TCP reacts to congestions by reducing its rate. This reduction depends on the way the loss, which is used as indication of congestion, is detected. If the loss is detected by duplicate acknowledgements (typically 3 [2]), the congestion window is halved. Otherwise the loss is detected by timeout and the sender reduces the size of its congestion window to one Maximum Segment Size (MSS). Depending on the way the loss is detected, TCP enters a slow-start phase (in the case of a timeout) or congestion avoidance phase (in the case of triple duplicates). The way the sender increases its congestion window is also phase dependent. More details can be found in RFC-1122 and in [2,3].

Many studies have contributed to better understand the behaviour of TCP and the parameters its throughput depends on. Several analytical models (e.g. [4], [5], [6], [7], [8]) have been developed for the throughput of long-term TCP connections and have helped understand the impact of certain parameters. However, these models have been derived under different assumptions and all assume that fast recovery phase is negligible and that the source resumes the linear increase of its congestion window directly after the reduction (as pointed by Altman et al. in [9]). More sophisticated models exist that try and alleviate some of these hypotheses. For example, [10] and [9] take into account the effect of the window size on the round-trip time and also the correlation between losses. The model proposed in [11] is more accurate and furthermore it takes into account the slow start phase, which makes it usable also for short sessions.

Another consequence of the success of TCP is that any new protocol deployed on the Internet should be TCP-friendly [12] in order not to disturb 90% of the traffic. To reach this TCP-friendliness, some multicast and real time protocols (e.g. [13–17]) have used the analytical models of the TCP throughput, to adapt their rate so as to obtain similar throughput as TCP in the same network conditions. The most popular models for these applications are the SQRT [4] and the PFTK [18] formula.

According to these two models, the TCP throughput is inversely proportional to the round-trip time (at least at low loss rates). This statement has an important consequence: if two TCP sessions share the same bottleneck, then the ratio of their throughputs (in terms of packets) is equal to the inverse ratio of their round-trip times. A simple experiment can however show that this statement is not always true. To this end, we ran different scenarios (100) with a simple topology consisting of one bottleneck over which a certain number of TCP New-Reno sessions compete. They all have the same packet size and different round-trip times. In each simulation, the bandwidth of the bottleneck and the number of concurrent sessions are chosen randomly and the loss rate is low (under 2%). Over all the scenarios, we record for each pair (i, j) of TCP connections the ratio of their throughputs ($\frac{B_i}{B_j}$) and the inverse ratio of their average round-trip times ($\frac{RTT_j}{RTT_i}$). All the pairs ($\frac{RTT_j}{RTT_i}$, $\frac{B_i}{B_j}$) are represented by a point in the scatter plot of Figure 1. According to the models, each point should be on the line $y = x$ since the two ratios should be equal. Figure 1 shows that even in the case of simple topologies, the scatter plot is not fitting the model. In some cases, the ratio of the throughputs equals seven times the inverse ratio of the average round-trip times. This also means that TCP is not always fair towards other TCP sessions. In [19], the authors have observed this unfairness in short-term sessions.

In this paper we propose to study the accuracy of the SQRT and PFTK models that are recalled in Section 2. The approach we propose for the validation is based on gathering an important number of TCP sessions obtained in various randomly generated topologies and scenarios, and comparing the throughput really obtained with the predicted one. The way we proceed and the results of the validation are given in Section 3. We then propose in Section 4 an alternative to these analytical models based on machine learning techniques. The results obtained with this approach are significantly better than those of SQRT and PFTK. In Section 5, we analyse more thoroughly situations where analytical models make bad predictions and we show that the main reason of their inaccuracy is their non discrimination of the two types of losses (triple duplicates versus timeout). Other TCP models [9,10,20,21] do not discriminate them either, and to the best of our knowledge, there exists no analytical model of TCP that distinguishes the two loss rates. As a first step in this direction, we show in Section 6 that taking into account the timeout loss rate can greatly improve the accuracy of TCP throughput models obtained by supervised learning techniques. This result suggests that similar improvements could be gained with analytical models by taking into account the same variable.

This paper is an extended version of [22]. In addition to several local improvements, the new section 4 proposes a detailed treatment of machine learnt models of TCP throughput. Section 6 is extended with a study of the introduction of new variables in machine learnt models (6.2), an evaluation of a larger panel of machine learning algorithms, and a discussion of implementation issues (6.4).

2 Analytical models of TCP throughput

In this section, we give a brief reminder of the PFTK and SQRT models, which are often used by other protocols to offer TCP-Friendliness. Both have been developed for long-term TCP connections (i.e., for flows with a large amount

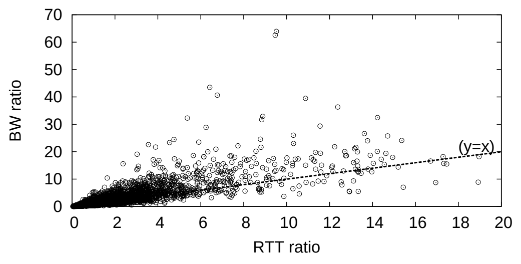


Fig. 1. The ratio of the throughputs of the two sessions versus the inverse ratio of their RTTs.

of data to send, such as file transfers).

In 1997 the only formula modeling the throughput of TCP was the one developed by Mathis et al. in [4] which is:

$$B_{tcp} = \frac{C * MSS}{RTT\sqrt{p}} \tag{1}$$

where $C \approx 1.22$, MSS is the maximum segment size, RTT the average round-trip time, and p the loss rate over the session. This model is often called the SQRT model. Only the congestion avoidance is taken into account in this model and all the losses are assumed to be detected by triple duplicates. This assumption, which requires that there is no timeout expiration, can only be valid if the loss rate is low.

In 1998 Padhye et al. [18] have proposed a more complex formula, taking into account losses detected by timeouts, which are frequent in high loss rate environments. The details and the hypothesis made for this model can be found in [18]. The formula, called PFTK, is summarised as follows:

$$\begin{aligned} &B_{tcp} \approx min(\tfrac{rwnd}{RTT}, \tfrac{MSS}{RTT\sqrt{\frac{2bp}{3}}+f(p)}) \\ &\text{with } f(p) = T_0 min(1, 3\sqrt{\tfrac{3bp}{8}})p(1+32p^2) \end{aligned} \tag{2}$$

where MSS, p and RTT are as described in the SQRT formula. T_0 is the initial value of the timeout, $rwnd$ the receiver window and b the number of packets acknowledged at once. This formula is said in [18] to be developed for Reno, but it is based on the hypothesis that all the packets sent after the loss and belonging to the same window are lost. This hypothesis means that the sender can only decrease its congestion window once during each round-trip time, which is true only in the case of NewReno (see [23]).

The two formulas do not take the effect of slow-start phases into account, which makes them unusable for the prediction of the throughput of short-term TCP sessions. They also neglect the fast recovery phase as said previously. Other assumptions have also been made for the modeling, but we do not develop them in the paper. The reader can refer to the original papers and to [23] or [24] for more detailed discussions of these models.

3 Models validation

In this section, we propose to validate the two models (SQRT and PFTK) by using a generic approach based on random simulations. More precisely, the quality of the models is measured by their ability at predicting the throughput of TCP in various topologies and scenarios. The way we generated these topologies and scenarios is described in Section 3.1. In Section 3.2, we give the criteria we use to measure the quality of model predictions. The results of the validation of the two formulas are discussed in Section 3.3.

3.1 Topologies and scenarios used

To validate the SQRT and PFTK formulas, we use 7600 TCP New-Reno[1] sessions chosen randomly over thousands of random topologies with random scenarios. Since the two formulas are developed for long-term throughput, we choose TCP sessions that last at least 400 seconds and for which the time needed to send all packets in a window is smaller than the RTT. The receiver window is chosen very large so as not to be the bottleneck. To create a topology and a scenario we proceed as follows: a network topology is generated randomly and then the network is simulated during a fixed amount of time, again by generating the traffic randomly. At the end of the simulation, we collect for all TCP New-Reno sessions that last at least 400 seconds, the loss ratio p computed over the whole session, the value of Maximum Segment Size *(MSS)*, the value of the timeout T_0, the average round-trip time RTT, the number of packets acknowledged at once b, and the TCP throughput obtained. This procedure is repeated until we have a sufficient number of sessions in the database.

To generate a random topology, we first select a random number of nodes (between 10 and 600) and then choose randomly the connections between these nodes. The bandwidth, the propagation delay, and the buffer size of the links were chosen randomly. The bandwidth is chosen between 56Kb/s and 100Mb/s while the propagation delay varies between 0.1ms and 500ms.

[1] We have used TCP NewReno, and not Reno, because, as stated earlier, both formulas are based on the assumption that the congestion window can only decrease once per RTT.

Concerning the traffic, the flows were chosen randomly among TCP and other types of traffic based on UDP and proposed by `ns-2`. The senders, the receivers, and the duration of each traffic were set randomly.

We are aware that this large set of random topologies and traffic conditions may also include many non realistic ones, but since the various analytical models of TCP are topology and traffic unaware, they are not supposed to give good results only in realistic scenarios. Moreover, as characterizing realistic scenarios is beyond the state of the art, trying to restrict ourselves to a large set of so-called realistic scenarios may create the risk of being too restrictive and thus introduce a bias.

3.2 Evaluation criteria

The quality of a model, or of an estimator, of which the goal is to predict a numerical output from some inputs, depends on how well it fits the data to predict. When the predicted value is closer to the observed one, the model is more accurate. Its accuracy, or adjustment, can be measured by different statistics. The mean square error is often used as well as the coefficient of determination. The mean square error is equal to:

$$MSE = \frac{1}{N}\sum_{t\in\tau}(\hat{X}_t - X_t)^2, \tag{3}$$

where τ is the set of data to predict of size N, X_t is the value to estimate, and $\hat{X}_t$ the value estimated by the model. The coefficient of determination is defined as:

$$R^2 = \frac{\sum_{t\in\tau}(\hat{X}_t - \overline{X})^2}{\sum_{t\in\tau}(X_t - \overline{X})^2} = 1 - \frac{\sum_{t\in\tau}(\hat{X}_t - X_t)^2}{\sum_{t\in\tau}(X_t - \overline{X})^2} \tag{4}$$

where $\overline{X}$ is the average of X_t over τ. Note that $(\frac{\sum_{t\in\tau}(\hat{X}_t - X_t)^2}{\sum_{t\in\tau}(X_t - \overline{X})^2})$ is the ratio of the mean square error to the variance.

When the coefficient of determination is close to 0, the scatter-plot is largely spread around the regression line, which means a less accurate model. And when the spread points are all near the regression line, the coefficient of determination is close to 1 (a perfect fit).

Since PFTK and SQRT formulas are used to provide TCP-Friendliness, the ratio of the predicted value to the value to predict is another important measure of the model accuracy. This ratio should be close to one to ensure that a protocol using one of the two formulas to determine its rate can provide TCP-Friendliness. This information is not carried in the MSE or in the coefficient of determination that are only sensitive to absolute distances between the predicted and real values. The ratio can be very high whereas the distance between them can be very small (in comparison to the average distance between the predicted values and the values to predict in the set τ).

We thus need another criterion that takes this ratio ($\frac{\hat{X}}{X}$) into account. The maximum (resp. the minimum) of this ratio gives an idea of how much the prediction can overestimate (resp. underestimate) the true value. However, the average and standard deviation of the ratio should be analysed with caution. Indeed, underestimation affects the average and the standard deviations less than overestimation while both have the same importance for our application. Therefore, in addition to the ratio, we propose to use also the *"absolute ratio"* defined by $max(\frac{\hat{X}}{X}, \frac{X}{\hat{X}})$, and which is sensitive in the same manner to both underestimation and overestimation.

To better explain our point, let us give a simple example. Suppose that $\tau = \{X_1 = 1, X_2 = 1\}$, that a first model returns $\hat{X}_1 = 1.5$ and $\hat{X}_2 = 0.05$, and that a second model predicts $\hat{X}_1 = 0.8$ and $\hat{X}_2 = 0.7$. The second model is clearly the most accurate, but if we use the ratio we will get for the first model an average equal to 0.775 and for the second 0.75. The first one offering a ratio closer to 1 will be considered as the best model. However, if we use the absolute ratio, the average with the first model will be 10.75 while it will be 1.34 with the second one. The absolute ratio is thus more representative of the respective quality of the two models.

Subsequently, we will use the coefficient of determination as well as the two ratios to evaluate TCP throughput estimators.

3.3 SQRT and PFTK accuracy

For each session of the validation set (consisting of 7600 TCP sessions), we compute, based on the collected parameters, the throughput predicted by SQRT and PFTK. We then compute for the set of predicted data, the coefficient of determination, and several statistics (average, minimum, maximum and standard deviation) concerning the two ratios ($\frac{\hat{X}_t}{X_t}$ and $\max(\frac{\hat{X}_t}{X_t}, \frac{X_t}{\hat{X}_t})$).

We plot in Figure 2 the predicted throughput as a function of the real throughput (the one we try to predict) for the two models. Ideally, the scatter plot should fit the regression line ($y = x$), i.e. the predicted value should be equal to the value to predict. Figure 2 shows that both models are far from fitting the regression line. A great amount of points are in fact spread around the

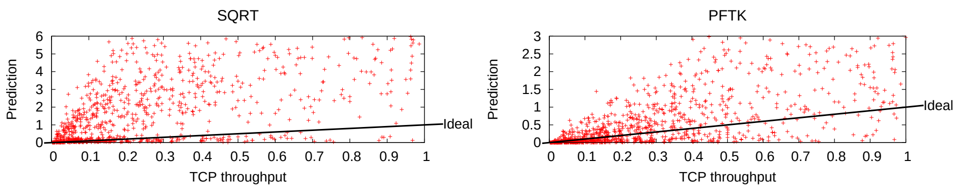


Fig. 2. The predicted throughput versus the real throughput.

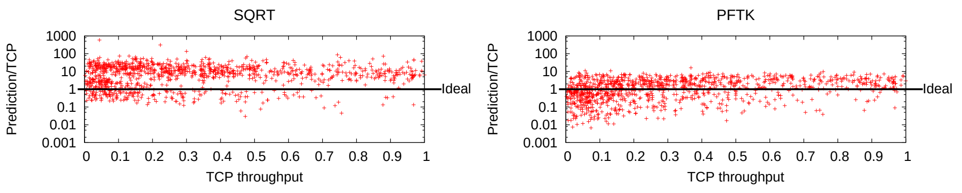


Fig. 3. The ratio of the predicted to the real throughputs versus the real throughput.

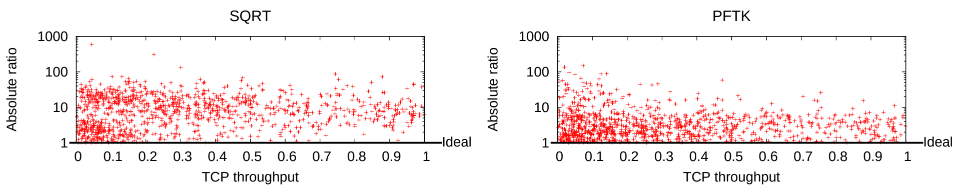


Fig. 4. The absolute ratio of the predicted to the real throughputs versus the real throughput.

latter. Figures 3 and 4 show the ratio and the absolute ratio with respect to the real throughput. In the ideal case, both graphs should be reduced to the straight line $y = 1$. This is again far from being the case. Whatever the performance criterion is, both models are inaccurate. Moreover, the range of the ratios for SQRT is $[0.013, 583.73]$ while the range for PFTK is $[0.006, 16.11]$. That means that a protocol that uses the SQRT model to provide TCP fairness can get 583.73 times more througput than a concurrent TCP flow, and 152.59 $(= 1/0.006)$ times less when using PFTK. In both cases, the protocol would not be TCP-Friendly.

Table 1 summarises the above figures numerically. In the formulas 3 and 4 τ is the validation set, X_t are the throughputs to predict and $\hat{X}_t$ is the throughput predicted by the model. The table shows the coefficient of determination and some statistics concerning the normal ratio and the absolute ratio of the two models. The SQRT model is less accurate than the PFTK one, which has already been shown in [5]. Its coefficient of determination is the lowest, which means that its points are much more spread around the regression line. Both its ratios are higher in average than those of PFTK. In other words, a protocol that will use the SQRT model will be less TCP-Friendly than one using the PFTK model. However, even if PFTK shows a better behaviour than SQRT, it is still not TCP-Friendly. The absolute ratio should be lower than 1.78 (as suggested in [16]) to provide the fairness towards TCP. Its average is 5.69 for SQRT against 3.15 for PFTK, which are both above 1.78. More precisely,

Table 1
The coefficient of determination (R^2), the mean square error (MSE) and statistics of the ratio (R) and the absolute ratio (AR) for the SQRT and PFTK models.

	R^2	MSE 10^{-3}	R				AR			
			avg	stdev	min	max	avg	stdev	min	max
Mathis	0.658	4.078	5.29	10.29	0.013	583.73	5.69	10.59	1	583.73
PFTK	0.814	2.211	2.2	1.19	0.006	16.11	3.15	5.81	1	152.59

over our validation set (7600 TCP sessions), 70% of PFTK predictions are not TCP-Friendly (against 76.92% for SQRT).

So, in conclusion, neither the SQRT model nor the PFTK model are accurate. This is due *in part*[2] to the fact that phases like slow-start and fast-recovery are not taken into account and that many hypotheses have been made to make the derivation of an analytical formula feasible[23,24].

4 Machine learnt models of TCP throughput

We have shown that the two analytical models fail to predict the TCP throughput accurately. 70% at least of the predicted throughputs provide an absolute ratio higher than 1.78. A more accurate model is thus needed. Our aim in this section is to check if it is possible to devise a better model than SQRT and PFTK, which would use the same parameters as these models. However, instead of developing a new analytical model, we propose here to automatically infer a model from random network simulations by using supervised learning algorithms developed for regression problems. Using such methods brings the advantage of not making any assumption.

Before presenting our approach in Section 4.4, we give a brief introduction to supervised learning algorithms in Section 4.1 followed, in Section 4.2, by a description of the supervised learning algorithms used subsequently.

4.1 Supervised learning

Automatic learning denotes methods which aim at extracting a model of a system (in our case, a computer network) from the sole observation (or the simulation) of this system in some situations. By model we mean some relationships between the variables used to describe the system. The goal of this

[2] We will see later that other causes exist

model may be to predict the behaviour of this system in some unencountered situations or to help understand its behaviour.

Supervised learning is the part of automatic learning which focuses on modelling input/output relationships. More precisely, the goal of supervised learning is to identify a mapping from some input variables to some output variable on the sole basis of a sample of observations of these variables. Formally, the sample of observations is called the learning sample LS and is a set of input/output pairs, $LS = \{< x_1, y_1 >, < x_2, y_2 >, ..., < x_N, y_N >\}$, where x_i is the vector of values of the input variables (also called the attributes) corresponding to the ith observation (also called an object) and y_i is its output value. Attribute values may be discrete or continuous. The goal of supervised learning can be formulated as follows: From a learning sample LS, find a function $f(x)$ of the input attributes that predicts at best the outcome of the output attribute y for any new unseen values of x. When the output takes its values in a discrete set $\{C_1, C_2, ..., C_m\}$, we talk about a classification problem and when it is continuous, we talk about a regression problem.

This problem is solved by a (supervised) learning algorithm. Loosely speaking, a learning algorithm receives a learning sample and returns a function f (an hypothesis or a model) which is chosen in a set of candidate functions (the hypothesis space). There exist many learning algorithms, which differ mainly in the hypothesis space but also in the optimisation algorithm that searches this space for a good model.

The main criterion used to assess learning algorithms is their prediction accuracy, i.e. the way the model they produce generalises to unseen data. Usually, the ranking among algorithms depends largely on the problem and how well the basic hypotheses of the learning algorithm are satisfied by this problem. Thus, none of the existing algorithms can be claimed to be globally superior to all other ones.

In this paper we use 4 learning algorithms for regression that are briefly described in Section 4.2. We also give, in the same section, a description of decision tree, which is a classification method that will be used later to analyse bad predictions. For a complete reference on supervised learning algorithms, see for example [25] or [26].

4.2 Supervised Learning methods

Decision/regression trees [27]. This is one of the most popular learning algorithm. A decision or regression tree represents an input/output model with a tree where each interior node is labeled with a test on one input attribute and each terminal node is label-led with a value of the output, i.e. a discrete class in the case of decision trees or a number in the case of regression trees. Figure 6 shows one decision tree that will be analysed later. To make a prediction for some observed inputs with such a tree, we simply traverse the tree from the top node to a terminal node according to the test issues and the prediction for the observed inputs is the value associated with the terminal node. By construction, a tree is thus very readable. This is also one of the fastest learning algorithm for learning but also for testing. In our experiments, we have used the algorithm for tree induction proposed in [27].

Tree-based ensemble methods. Although they present several nice characteristics, decision and regression trees are often not competitive with other learning methods. In supervised learning, ensemble methods are generic techniques that improve a learning algorithm by learning several models (from the same learning sample) and then by aggregating their predictions. In our experiments, we will use three ensemble methods for regression trees: Bagging, Extra-Trees, and MART. Bagging [28] builds each tree of the ensemble from a bootstrap sample drawn from the original learning sample. The Extra-trees method [29] grows each tree from the complete learning sample but by randomising tree tests. With MART [26], the trees are built in sequence, each tree of the sequence trying to reproduce an output equal to the difference between the true output (in the learning sample) and the sum of the predictions given by the previous trees in the ensemble. In all cases, we use 25 trees. Bagging and Extra-trees were used with their default settings (see [28] and [29]). The two parameters μ and J of MART (see [26]) were fixed respectively to 0.2 and 40. These values were determined from some preliminary trials.

Multilayer perceptrons [30]. Multilayer perceptrons are a particular family of artificial neural networks. Neural networks represent a model as the interconnection of several small units called perceptrons that compute a weighted average of their inputs and send this average through a non linear functions (usually a hyperbolic tangent). This method usually gives more accurate models than decision trees but a neural network is not interpretable and is also much more demanding in terms of computing times and computer resources. In our experiments, we have used a Levenberg-Marquard optimisation algorithm to learn neural networks and we have tried several neural network architectures. Only the best results are presented.

4.3 Database

As for the validation of SQRT and PFTK, supervised learning techniques require a database as representative as possible of the conditions under which we will apply the model. So, the database generation should take into account all the uncertainties we have a priori about the topology of the networks, the

Table 2
The mean square error and the coefficient of determination for the inferred model using the PFTK parameters only.

Method	MSE 10^{-3}	R^2
SQRT	4.078	0.658
PFTK	2.211	0.814
MART	1.246	0.895
BAGGING	1.211	0.898
EXTRA-TREES	1.193	0.900
MLP	1.048	0.912

user behaviours, and the protocols. The database was generated exactly as described in Section 3.1. It contains 18000 observations, where each observation is a TCP session represented by an inputs/output pair $< x_i, y_i >$, where y_i is the throughput of the session and x_i collects some parameters that defines the system during the whole session. In this section, x_i contains the parameters used by PFTK, i.e the average round-trip time, the loss rate, the initial value of the timeout, the packet size, and the number of packets acknowledged at once.

4.4 Protocol and results

All the supervised learning algorithms described above have been trained on the database and each has inferred a model. The models obtained have been tested on the same validation set of 7600 sessions (disjoint from the set of 18000 observations used for learning) that was used for the validation of SQRT and PFTK models. We compute, for each inferred model, the coefficient of determination and the corresponding mean square error over the set of predicted data and represent them in Table 2. All the models obtained by learning get a mean square error at least 50% lower than the error of the PFTK model (which is the best of the two analytical models). Their coefficients of determination are also higher than those of PFTK and SQRT. The four models are thus more accurate than the analytical models. According to this criterion, MLP is the best method followed by the Extra-trees.

However, as said previously, the adjustment criterion is not the only parameter to take into account to assess the model quality. The ratio and the absolute ratio are also important. Table 3 collects some statistics related to the latter. All the methods offer better fairness towards TCP than PFTK and SQRT models. Their average absolute ratios are below 1.78 where the one of PFTK is equal to 3.15. Extra-trees offer the best ratio in average. Since it has also

Table 3
The ratio and the absolute ratio statistics of the inferred models using the PFTK parameters.

Ratio

	avg	min	max	stdev
SQRT	5.29	0.013	583.73	10.29
PFTK	2.2	0.006	16.11	1.19
Extra-trees	1.18	0.22	24.33	0.51
Bagging	1.17	0.18	25.5	0.5
MART	1.23	0.07	25.91	0.7
MLP	1.62	0.05	18.9	2.2

Absolute ratio

	avg	min	max	stdev
SQRT	5.69	1	583.73	10.59
PFTK	3.15	1	152.59	5.81
Extra-trees	1.35	1	24.33	0.51
Bagging	1.35	1	25.5	0.5
MART	1.46	1	25.91	0.83
MLP	1.78	1	20	2.14

a low standard deviation, the fairness is often achieved with this method. Nevertheless, the maximum absolute ratio reaches 24.33, which is quite high, even though this value is still much lower than the maximum absolute ratios of PFTK and SQRT (respectively 152.59 and 583.73).

4.5 Conclusions

We have proposed in this section an alternative to the analytical models of SQRT and PFTK, based on supervised learning techniques. From the PFTK parameters only, the four supervised learning algorithms we have used provide more accurate estimators of the throughput of TCP than SQRT and PFTK. They present a higher coefficient of determination, which means that they fit more the regression line, and have a lower absolute ratio, which is a sign of better fairness. These models, e.g. the extra-trees one, are therefore interesting alternatives to PFTK and SQRT formulas since they are significantly more accurate and do not require more monitoring resources than these two models (as they use exactly the same set of parameters). Their application in the context of TFRC-like protocols would be straightforward. In some situations, however, these models may obtain an absolute ratio close to 25. Although much lower than the max ratio of PFTK and SQRT, this value is still not fully satisfactory. In the next section, we investigate the reason for this high ratio.

As a last remark, it should be noted that our machine-learnt models of TCP have been derived from a large set of random topologies and traffic conditions. The randomness of the learning set may be reduced by focusing more on so-called realistic topologies and traffic conditions, provided that good criteria are found to characterize them. Note however that doing so can only improve our learnt models, because the realistic cases, being a subset, will obviously drive the learning algorithms to more specialized models. A random learning set is thus actually a worst case for our approach. As our models are already quite better than existing analytical models in such a worst case, our approach and results are therefore already very encouraging.

5 Analysis of bad predictions

In Section 3.3, we have shown that SQRT and PFTK are not always accurate and their predicted throughputs are often not TCP-Friendly. In this section, we propose to investigate the reasons for the bad predictions of the models. This study aims at characterising the conditions leading the formulas to under or overestimate the throughput. To this end we propose to use an original approach based on the analysis of the validation test by the decision tree method (which provides *"interpretable"* models). With this method, we will build a classification model to discriminate the good and the bad predictions in terms of different parameters gathered from the network. The analysis of the tree so obtained will then provide a characterisation of the conditions under which the models give bad predictions. Before going to the analysis in Section 5.2, we first explain how we classify the predictions into good and bad predictions.

5.1 Classification of predictions

The prediction of a model will be considered as good if it preserves TCP-Friendliness. In other words, a prediction is considered as "good" if the ratio of the predicted throughput to the actual throughput belongs to $[1/K, K]$, where $K \geq 1$ is a factor that defines the TCP fairness bounds. If the ratio belongs to $(0, 1/K)$ (resp. (K, ∞)) the prediction underestimates (resp. overestimates) the throughput. These three areas are represented graphically in Figure 5.

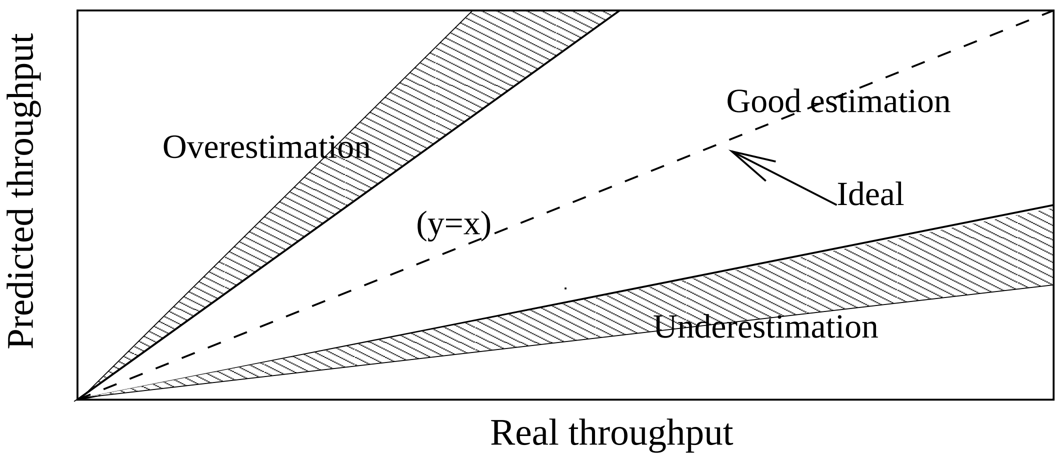


Fig. 5. The three areas defining the quality of the predictions of a model.

Table 4
Distribution of prediction types for the two models.

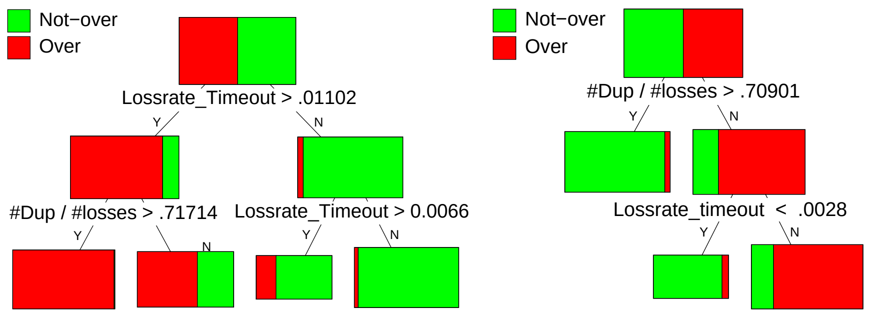
Interval	PFTK	SQRT
$[0.001, 1/3)$ (under)	6.14%	3.43%
$[1/3, 3]$	33.42%	25.66%
$(3, 1000]$ (over)	60.44%	70.91%

To define bad predictions, we choose another parameter $K' > K$ and we consider as bad predictions the predictions that are such that the ratio belongs to $(0, 1/K') \cup (K', +\infty)$. This definition of bad predictions thus leaves a region of fuzziness between good and bad predictions which is represented in grey in Figure 5. Subsequently, by abuse of language, the word overestimation (resp. underestimation) will be used to denote the overestimation (resp. underestimation) area of the graph minus the gray area. Thus, the words underestimation and overestimation will be synonyms of bad prediction.

For our study we use the commonly accepted value of $K = 1.78$ to define TCP-Friendliness and we consider that if the absolute ratio is higher than $K' = 3$, then the prediction is bad. The value of this threshold is purely subjective. However, a study had been done with a threshold equal to 10 and had led to similar results. The choice of the value three is thus not restrictive.

5.2 Decision trees analysis

As said in the previous section, there are two kinds of bad predictions: underestimation and overestimation. We have then 4 cases to analyse: PFTK and SQRT in both the underestimation and overestimation cases. Table 4 shows the distribution of the predictions of PFTK and SQRT into the dif-



(a) SQRT overestimation (b) PFTK overestimation

Fig. 6. The top of the decision trees classifying bad predictions. (lossrate_timeout is defined as the number of losses detected by timeouts (#timeout) divided by the number of packets transmitted).

ferent classes. The cases of underestimation are too rare for both PFTK and SQRT to obtain statistically meaningful conclusions from their analysis. So we will drop these cases. It thus remains two situations: SQRT and PFTK in overestimation.

To analyse the reason for overestimation in both cases, we first classify each prediction into one of two classes: *"Over"* to denote the predictions that overestimate the TCP throughput and *"Not-Over"* to denote the other predictions. Then a decision tree is built to explain the classification using as inputs the PFTK parameters, the proportion of losses detected by triple duplicates, and the proportion of losses detected by timeout expirations. The top of each tree is represented in Figure 6 and is discussed below.

The tree of Figure 6(a) shows that if the proportion of losses detected by timeouts exceeds a certain threshold then SQRT overestimates the throughput. Indeed, each time a packet loss is recovered by timeout no data is transferred, and this sender inactivity is not taken into account by the model. The model still considers that data are sent and the predicted throughput obtained is higher than TCP's. This result is already known in the networking community and was clearly expected.

The tree of Figure 6(b) is related to the overestimation of PFTK. It points out that if the proportion of losses detected by triple duplicates is under a certain threshold, then the estimation exceeds the real throughput. When the number of losses detected by triple duplicates decreases, the number of losses detected by timeouts increases. The loss rate p used in $E[A]$ (eqn. (16) of [18]) then becomes higher than the value that should be used, since it should include only losses due to triple duplicates. Thus, $E[A]$ is lower than what it should be and the predicted throughput, B (inversely proportional to $E[A]$), is then higher than the real one. In addition, when the number of timeouts increases, the number of slow-start phases increases, and during these phases the predicted throughput is higher than the actual throughput. The timeout

Table 5
The mean square error and the coefficient of determination for the inferred model using the PFTK parameters + the timeout loss rate.

Method	MSE 10^{-3}	R^2
SQRT	4.078	0.658
PFTK	2.211	0.814
BAGGING	0.679	0.943
EXTRA-TREES	0.655	0.945
MART	0.482	0.960
MLP	0.322	0.973

loss rate affects also the prediction of PFTK.

In conclusion, a discrimination between the way the losses are detected seems to be required for a good prediction. To the best of our knowledge, no models of TCP, even recent models such as [9] or [11], make this distinction. In the next section, we highlight the importance of incorporating the timeout loss rate in the context of models inferred by machine learning techniques.

6 Addition of other variables in the models

The study conducted in the previous section shows that the proportion of losses detected by triple duplicates has an important role in the prediction of the throughput. Hence, introducing the rate of losses detected by triple duplicates, or its dual (the rate of losses detected by timeouts), in the learning phase seems to be needed in order to improve the models. In Section 6.1, we carry out experiments by taking into account this variable. In Section 6.2, we add some further variables measuring some statistics on the round-trip time. We end the section with a discussion of implementation issues related to the estimation of the timeout loss rate.

6.1 Timeout loss rate

In this section, we rerun the different learning algorithms over our database using as inputs the PFTK parameters and the timeout loss rate, and we validate the obtained models over the validation set. The results concerning the adjustment and the ratios are summarised respectively in Tables 5 and 6.

Table 5 illustrates the coefficient of determination and the mean square error

Table 6
The ratio and the absolute ratio statistics of the inferred model using the PFTK parameters + timeout loss rate.

Ratio

	avg	min	max	stdev
SQRT	5.29	0.013	583.73	10.29
PFTK	2.2	0.006	16.11	1.19
Extra-trees	1.07	0.2	9.71	0.34
Bagging	1.07	0.17	11.81	0.4
MART	1.11	0.16	6.49	0.49
MLP	1.12	0.2	5.2	0.54

Absolute ratio

	avg	min	max	stdev
SQRT	5.69	1	583.73	10.59
PFTK	3.15	1	152.59	5.81
Extra-trees	1.14	1	9.71	0.33
Bagging	1.15	1	11.81	0.4
MART	1.22	1	6.49	0.55
MLP	1.24	1	5.2	0.58

for the different methods. Both have been greatly improved by the introduction of the timeout loss rate in the models. The MSE is reduced in average by more than 50% and it is even reduced by a factor 3 in the case of MLP. The coefficient of determination is increased in consequence. The ratios (Table 6) have also been decreased significantly for all methods.

Among the different models, Extra-trees is the method that offers the lowest absolute ratio. Furthermore, its standard deviation is low, which means that in general the prediction are not far from the average. However, the maximum absolute ratio with this method is quite high (9.71). So, one may prefer to use MLP which has a higher average absolute ratio but has a maximum absolute ratio of 5.2.

6.2 Other variables

To see if further improvements could be gained, we have run a last experiment with even more variables. In addition to PFTK parameters and the timeout loss rate, we have included in the inputs the minimum, the maximum, and

Table 7
The mean square error and the coefficient of determination for the inferred model using all collected variables.

Method	MSE 10^{-3}	R^2
SQRT	4.078	0.658
PFTK	2.211	0.814
Bagging	0.525	0.956
Extra-Trees	0.501	0.958
MART	0.423	0.964
MLP	0.245	0.979

Table 8
The ratio and the absolute ratio statistics of the inferred model using all collected variables.

Ratio

	avg	min	max	stdev
SQRT	5.29	0.013	583.73	10.29
PFTK	2.2	0.006	16.11	1.19
Extra-trees	1.07	0.21	8.94	0.35
Bagging	1.07	0.24	11.33	0.42
MART	1.09	0.22	4.52	0.42
MLP	1.12	0.22	4.5	0.49

Absolute ratio

	avg	min	max	stdev
SQRT	5.69	1	583.73	10.59
PFTK	3.15	1	152.59	5.81
Extra-trees	1.14	1	8.94	0.35
Bagging	1.15	1	11.33	0.41
MART	1.21	1	4.52	0.46
MLP	1.24	1	4.5	0.52

the standard deviation of the round-trip time and the duration of the TCP session. The results of these new models are given in Table 7 and 8.

The mean square error of all methods are again decreased when using all the variables. However, the improvement is this time not very important. The average ratios are comparable to those obtained with the intermediate set of attributes (PFTK parameters + timeout loss rate), but the range and the

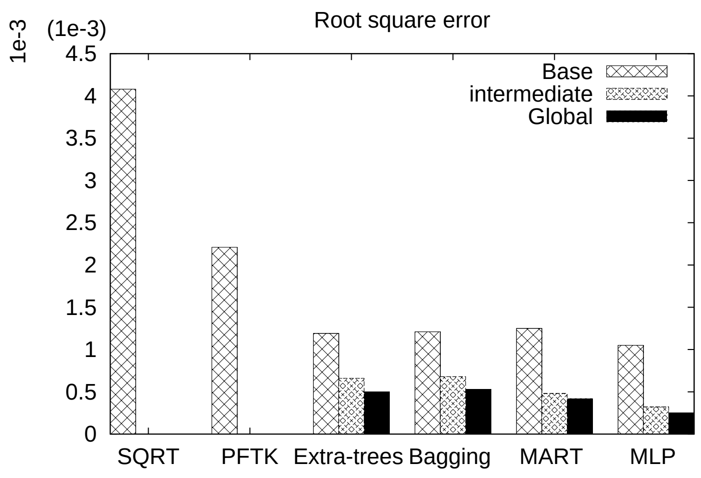


Fig. 7. Recapitulation: mean square error for the different methods with different set of attributes.

standard deviation have been greatly decreased. With this attributes set, MLP and MART are the methods to recommend.

6.3 Synthesis

Figure 7 makes a synthesis of the tables in terms of the mean square error. The terms "base", "intermediate", and "global" qualify the set of attributes used. They respectively denote the set of attributes used by PFTK, the latter plus the timeout loss rate, and finally all collected variables. In the three cases, the supervised learning methods give better results than PFTK and SQRT. The introduction of the timeout loss rate more than halves the mean square error. The introduction of all the attributes still improves the accuracy, but only slightly.

The same effect has been seen on the ratios. The introduction of all attributes have essentially decreased the maximum absolute ratio as shown in Figure 8.

In general, the usage of the whole set of variables does not improve significantly the results. We can be satisfied with a model that is built over the PFTK parameter plus only the timeout loss rate. It is clear that the introduction of such variable in an analytical model will capture more phenomena and will also lead to better accuracy.

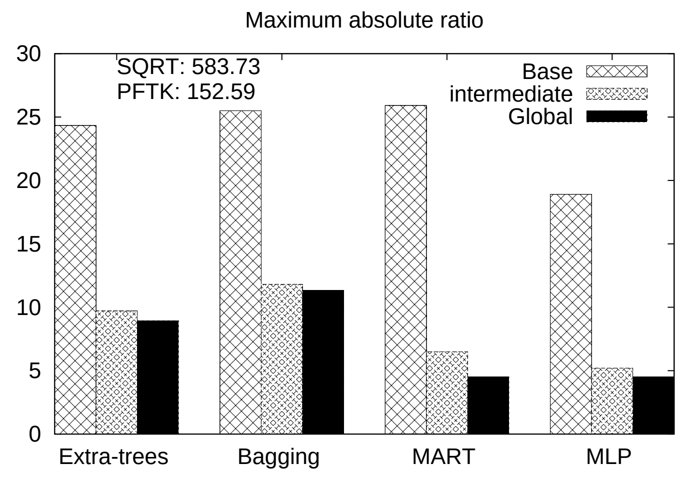


Fig. 8. Recapitulation: The maximum absolute ratio reached by the different methods with different sets of attributes.

6.4 Implementation issues in a TFRC-like protocol

The models we have proposed (and also a possible analytical model that would use the timeout loss rate) aim at predicting in an accurate manner the throughput of a long TCP session. However, in practice these models are often used to provide TCP-Friendliness in a TFRC-like protocol.

Generally, a source that uses such a model to provide TCP-Friendliness sends a UDP flow, whose rate is adapted according to the model from information collected from the network. The packets of this UDP source are not acknowledged as in the case of TCP, and so the rate cannot depend on the acknowledgements. The source does not know if a certain packet is received or not (it needs to know only the loss rate). Neither does it know if a burst is received or not. Thus, if a burst is not received, the source will continue sending data while TCP will not. A TCP source stops sending data when it does not receive any acknowledgement. Therefore, the notion of timeout expiration, and hence of timeout loss rate, does not exist in the context of a TFRC-like protocol. Consequently, even if we have an accurate model using the timeout loss rate, its application in the context of a non TCP protocol is not straightforward since it requires the estimation of the timeout loss rate a TCP flow would experience in the same conditions.

A solution to estimate the timeout loss rate could be to predict, according to the network state, whether this is a state where TCP would undergo a timeout expiration and then to count the losses that occur in such states. A prediction model for timeout expiration could be obtained in a similar way as our throughput models, i.e. by the application of supervised learning techniques on a database of random network simulations. This solution needs however further investigation.

7 Conclusions

We have studied the accuracy of SQRT and PFTK models. To this end we have built a database with a high number of TCP sessions gathered in random scenarios and have compared the results predicted by the models with the observed throughputs. SQRT and PFTK are not very accurate and we have pointed out the reason. PFTK, which is the reference among the analytical models of TCP throughput, uses the global loss rate p that accounts indifferently for losses detected by triple duplicates and losses detected by timeouts. This non discrimination affects the result: the TCP throughput is overestimated. The application of machine learning algorithms allows us to highlight the importance of the distinction between the two types of losses, which can indeed greatly improve the quality of the models. Our analysis suggests also that future research aiming at the analytical modelling of the throughput of TCP should certainly take into account the timeout loss rate.

We have also proposed an alternative to analytical modelling based on supervised learning, which offers better results than the two tested models even without discriminating the losses. By incorporating the timeout loss rate, it improves even further the quality of the predictions. The embedding of these automatically induced models into TFRC-like protocols is not very expensive in terms of computer resources. The learning phase has to be carried out only once offline. There is no need to monitor more network environment variables than with the PFTK model. And finally the cost of making a prediction with a model is very low in all cases (in the order of a microsecond with the slowest method).

In the future it would be very interesting to compare these models with more recent TCP models such as those of Altman et al. [9] and of Sikdar et al. [11]. However, the first one uses an infinite sum of terms which is difficult to compute in practice, while the second one has been developed for Reno and not NewReno. On a broader point of view, we would like to further exploit machine learning techniques in networking. These methods do not make any hypothesis and can thus take implicitly into account all TCP phases and long as well as short sessions, provided that they are represented in the database. The approach can also be easily extended to any version of TCP (e.g. SACK) or any other protocol. Finally, the application of supervised learning techniques is automatic and needs much less time and effort than the derivation of an analytical model, which may be of great importance in the rapidly evolving domain of networking. Still, an analytical formula like PFTK (even imperfect) has also lots of merits, as it allows researchers to better understand the influence of the parameters

Acknowledgements

This work has been partially supported by the Belgian Science Policy in the framework of the IAP program (MOTION P5/11 project) and by the European Commission (E-NEXT NoE and ANA IST-FET project). P. Geurts is a Research Associate of the Belgian National Fund of Scientific Research (FNRS).

References

[1] C. Fraleigh, S. Moon, B. Lyles, C. Cotton, M. Khan, D. Moll, R. Rockell, T. Seely, S. Diot, Packet-level traffic measurements from the Sprint IP backbone, IEEE Network 17 (6) (2003) 6– 16.

[2] M. Allman, V. Paxson, W. Stevens, TCP Congestion Control, RFC 2581 (April 1999).

[3] C. Barakat, TCP/IP modeling and validation, IEEE Network 15 (3) (2001) 38–47.

[4] M. Mathis, J. Semke, Mahdavi, T. Ott, The macroscopic behavior of the TCP congestion avoidance algorithm, ACM Computer Communication Review 27 (3) (1997) 67–82.

[5] J. Padhye, V. Firoiu, D. Towsley, J. Kurose, Modeling TCP Reno performance: a simple model and its empirical validation, IEEE/ACM Transactions on Networking 8 (2) (2000) 133–145.

[6] A. Kumar, Comparative performance analysis of versions of TCP in a local network with a lossy link, IEEE/ACM Transactions on Networking 6 (4) (1998) 485–498.

[7] E. Altman, K. Avrachenkov, C. Barakat, TCP in presence of bursty losses, in: Proceedings of the 2000 ACM SIGMETRICS, ACM Press, New York, NY, USA, 2000, pp. 124–133.

[8] O. Aït-Hellal, L. Yamamoto, G. Leduc, Cycle-based TCP-Friendly Algorithm, in: Proceedings of IEEE Globecom'99, IEEE Press, Rio de Janeiro, 1999, pp. 776–780.

[9] E. Altman, K. Avrachenkov, C. Barakat, A Stochastic Model of TCP/IP with Stationary Random Losses, IEEE/ACM Transactions on Networking 13 (2) (2005.) 356–369.

[10] M. Garetto, R. Cigno, M. Meo, M. A. Marsan, Closed queueing network models of interacting long-lived TCP flows, IEEE/ACM Transactions on Networking 12 (2) (2004) 300–311.

[11] B. Sikdar, S. Kalyanaraman, K. S. Vastola, Analytic models for the latency and steady-state throughput of TCP Tahoe, Reno, and SACK, IEEE/ACM Transactions on Networking 11 (6) (2003) 959–971.

[12] S. Floyd, K. Fall, Promoting the use of end-to-end congestion control in the Internet, IEEE/ACM Transactions on Networking 7 (4) (1999) 458–472.

[13] L. Vicisano, J. Crowcroft, L. Rizzo, TCP-like congestion control for layered multicast data transfer, in: Proceedings of IEEE INFOCOM'98, San Francisco, CA, 1998, pp. 996–1003.

[14] D. Sisalem, A. Wolisz, MLDA: A TCP-friendly congestion control framework for heterogenous multicast environments, in: Proceedings of the Eighth International Workshop on Quality of Service (IWQoS 2000), Pittsburgh, 2000, pp. 65–74.

[15] J. Widmer, M. Handley, Extending equation-based congestion control to multicast applications, in: Proceedings of ACM SIGCOMM'01, ACM Press, 2001, pp. 275–285.

[16] S. Floyd, M. Handley, J. Padhye, J. Widmer, Equation-based congestion control for unicast applications, in: Proceedings of ACM SIGCOMM'00, Stockholm, Sweden, 2000, pp. 43–56.

[17] I. El Khayat, P. Geurts, G. Leduc, Improving TCP in wireless networks with an adaptive machine-learnt classifier of packet loss causes, in: Proceedings of the International Conference on Networking, LNCS 3462, Springer-Verlag, 2005, pp. 549–560.

[18] J. Padhye, V. Firoiu, D. Towsley, J. Krusoe, Modeling TCP Throughput: A Simple Model and its Empirical Validation, in: Proceedings of ACM SIGCOMM'98, 1998, pp. 303–314.

[19] F. Baccelli, D. Hong, AIMD, Fairness and fractal scaling of TCP Traffic,, in: Proceedings of IEEE INFOCOM'02, Vol. 21, 2002, pp. 229 – 238.

[20] A. Misra, T. J. Ott, The window distribution of idealized TCP congestion avoidance with variable packet loss, in: Proceedings of IEEE INFOCOM'99, 1999, pp. 1564–1572.

[21] V. Misra, W.-B. Gong, D. F. Towsley, Fluid-based analysis of a network of AQM routers supporting TCP flows with an application to RED, in: Proceedings of ACM SIGCOMM'00, 2000, pp. 151–160.

[22] I. El Khayat, P. Geurts, G. Leduc, On the accuracy of analytical models of TCP throughput, in: Proceedings of the International Conference on Networking, Vol. 3976 of LNCS, Springer, 2006, pp. 488–500.

[23] E. Altman, K. Avrachenkov, C. Barakat, A stochastic model of TCP/IP with stationary random, in: Proceedings of the ACM SIGCOMM'00, 2000, pp. 231–242.

[24] J. Widmer, R. Denda, M. Mauve, A Survey on TCP-Friendly Congestion Control, IEEE Network 15 (3) (2001) 28–37.

[25] R. Duda, P. Hart, S. D.G., Pattern Classification, 2nd Edition, John Wiley & Sons, 2000.

[26] T. Hastie, R. Tibshirani, J. Friedman, The elements of statistical learning: data mining, inference and prediction, Springer, 2001.

[27] L. Breiman, J. Friedman, R. Olsen, C. Stone, Classification and Regression Trees, Wadsworth International (California), 1984.

[28] L. Breiman, Bagging predictors, Machine Learning 24 (2) (1996) 123–140.

[29] P. Geurts, D. Ernst, L. Wehenkel, Extremely randomized trees, Machine Learning 36 (1) (2006) 3–42.

[30] C. Bishop, Neural Networks for Pattern Recognition, Oxford: Oxford University Press, 1995.