Computer Networks 108 (2016) 307–322

Contents lists available at ScienceDirect

Computer Networks

journal homepage: www.elsevier.com/locate/comnet

Predicting expected TCP throughput using genetic algorithm

Cristian Hernandez Benet[a,*], Andreas Kassler[a], Enrica Zola[b]

[a] Karlstad University (KAU), Universitetsgatan 2, Karlstad, Sweden
[b] Universitat Politécnica de Catalunya (UPC), C. Jordi Girona 1–3, Barcelona, Spain

ARTICLE INFO

Article history:
Received 19 January 2016
Revised 3 July 2016
Accepted 31 August 2016
Available online 9 September 2016

Keywords:
Genetic algorithm
TCP throughput
Prediction
IEEE 802.11

ABSTRACT

Predicting the expected throughput of TCP is important for several aspects such as e.g. determining handover criteria for future multihomed mobile nodes or determining the expected throughput of a given MPTCP subflow for load-balancing reasons. However, this is challenging due to time varying behavior of the underlying network characteristics. In this paper, we present a genetic-algorithm-based prediction model for estimating TCP throughput values. Our approach tries to find the best matching combination of mathematical functions that approximate a given time series that accounts for the TCP throughput samples using genetic algorithm. Based on collected historical datapoints about measured TCP throughput samples, our algorithm estimates expected throughput over time. We evaluate the quality of the prediction using different selection and diversity strategies for creating new chromosomes. Also, we explore the use of different fitness functions in order to evaluate the goodness of a chromosome. The goal is to show how different tuning on the genetic algorithm may have an impact on the prediction. Using extensive simulations over several TCP throughput traces, we find that the genetic algorithm successfully finds reasonable matching mathematical functions that allow to describe the TCP sampled throughput values with good fidelity. We also explore the effectiveness of predicting time series throughput samples for a given prediction horizon and estimate the prediction error and confidence.

1. Introduction

Achieving the best network performance is one of the main goals for the computer network research community. Many researchers have been focusing on studying and forecasting bandwidth demands so as to properly use and distribute the available resources. As many applications have been shifting towards TCP/IP based networks [1], the need for further research on TCP/IP throughput prediction is evident. Also, nowadays multi-homing capabilities enable concurrent data transmissions over different interfaces. To enhance this capability, protocols such as Multipath TCP (MPTCP) can be used to improve the throughput by sending different flows on each interface simultaneously. The main drawback is to determine in which subflows should the next packets be sent over in order to efficiently use several interfaces simultaneously [2,3]. Moreover, the limited power capacity of mobile devices requires an efficient use of vertical handover (VHO) (i.e., handover between one interface to the other). Forecasting the TCP throughput may improve decision making during a VHO as well as using several interfaces simultaneously and efficiently [4].

In the literature, several methods exist to estimate the end-to-end TCP throughput. For example, model based techniques try to model the TCP throughput as a function of e.g. packet loss rate, round-trip-time (RTT) and maximum segment size (MSS) [5,6]. In order to predict the TCP throughput over time, one needs to obtain predictions of e.g. loss rate and RTT, which is quite difficult to achieve. On the other hand, using probe based techniques [7,8], TCP throughput may be estimated by sending a train of probing packets (typically using packet pair techniques) to the destination. However, a prediction over time requires frequent packet pairs to be sent which may translate into high overhead and low prediction quality. Finally, history based techniques try to model the TCP throughput evolution over time as a time series and apply tools such as neural networks to find patterns [1]. Based on such patterns, one tries to predict future TCP throughput over time.

The application of Genetic Algorithms (GA) to optimise processes and solve complex problems is widely used in computer networks thanks to its easy applicability to a specific problem. GA can solve large optimization problems with large search spaces and it has been used e.g. to solve routing problems [9,10] and for network traffic prediction [11–13]. An important feature of GA is that it provides a near-optimal solution in quick time. Time-series

* Corresponding author.
E-mail addresses: cristian.hernandez-benet@kau.se, crishern@kau.se (C. Hernandez Benet), andreas.kassler@kau.se (A. Kassler), enrica@entel.upc.edu (E. Zola).

http://dx.doi.org/10.1016/j.comnet.2016.08.027
1389-1286/

modelling can be combined with GA in order to extend GA's domain of optimisation and apply it for forecasting. Authors in [14] applies this technique to predict the traffic demands for next-generation wireless networks in a cognitive wireless setting with primary and secondary users, which is of chaotic nature. Based on a time series model, the authors devise a GA that tries to fit a combination of mathematical expressions to model the time series of the traffic demands. After tuning the best combination of expressions, the GA can predict future traffic demands with reasonable fidelity. However, authors in [14] formulate a rather simple GA with a standard fitness function and selection strategy, which provides poor results when applied to TCP throughput prediction because of several TCP related issues such as slow start and congestion avoidance phases [6].

In this paper we take a similar approach in order to predict the estimated TCP throughput over time by using historical samples of observed throughput. Those samples are modelled as a time series and prediction techniques based on GA are applied to estimate the future TCP throughput sample values. Different from [14], we address the TCP prediction by introducing some modifications on the GA. The main goal is to show the impact of several tuning parameters in the GA over the reliability of the TCP prediction. To this end and to keep the study easy to understand, we focus on two representative TCP traces obtained in a Wireless LAN environment. Several fitness functions and selection methods of the GA are analyzed and combined in order to fully understand their impact on the prediction in two scenarios: one with a regular pattern and another one with abrupt changes in the trend[1].

The main contributions of this paper can be summarized as follows:

- We tackle the problem of predicting the TCP expected throughput as a mathematical optimisation problem to match a combination of mathematical expressions to a **time series** that is composed **of measured TCP throughput sample values**.
- In contrast to [14], we **customise the GA** by applying different selection and diversity strategies to find the equation that best fits the sampled TCP throughput.
- We **evaluate the reliability** of several fitness functions in calculating the suitability of a given chromosome.
- We assess the **impact of the proposed modifications of the GA** in two scenarios based on extensive simulations.
- Finally, we explore the possibility of **reducing the frequency of retraining**, thus showing the trade-off between the prediction error and the time to solve the GA.

The proposed algorithm may be beneficial in several context. For example, with our approach, a better MPTCP subflow scheduling method could be designed taking into account the expected throughput of each subflow over time. Also, it could help mobile users to improve their experience by assessing them during handover decision (e.g., to which AP one should connect while moving in a given area, which technology will provide the highest throughput in the near future in a HetNet scenario, etc.).

The remainder of the paper is structured as follows. Section 2 introduces the background on GAs, time series and forecasting method, together with a review of the related work. Section 3 describes the problem statement and the approach followed in this work. The setup for the numerical evaluation is detailed in Section 4, together with an investigation on the performance of the proposed GA using different scenarios and algorithmic settings. Finally, Section 5 concludes the paper.

[1] We explicitly acknowledge the fact that a larger set of TCP traces would be needed if one wants to assess the accuracy of a given configuration of the GA. Instead, several configurations of the GA are explored in this work, which allows the use of a smaller set of input data.

2. Background and related work

For several application scenarios, gaining information beforehand on the throughput that a TCP connection may provide in the near future may lead to a better planning of the network resources and thus to an improvement in the network performance. The TCP throughput evolution over time depends on several factors and is influenced by the TCP congestion control algorithm using packet loss detection to control and adapt the sending rate. As already mentioned, several approaches for the prediction of the TCP throughput can be found in the literature. Formula-based approaches attempt to mathematically model the TCP throughput according to some parameters. This approach requires an accurate model in order to find the correlation between the model parameters and the TCP throughput, or instead large measurement campaigns to find out the corresponding relation. However, such approach can be applied easily to different scenarios under the model assumptions. For instance, authors in [15] use the available bandwidth, while authors in [16] use the congestion window's evolution of long-lived TCP flows.

On the other hand, history-based techniques attempt to predict TCP throughput over time from saved measurement data using historical data series. The benefit of history-based techniques is that they can predict TCP throughput only by analysing the time series behavior using some algorithm or tool such as GA or neural network, in order to detect patterns in the time series that these techniques exploit for the prediction. Such approach does not require the information about specific TCP related parameters such as MSS or packet loss statistics, which may be difficult to obtain. However, history based techniques typically work on a small dataset so it is difficult to generalize the findings from the measurement to other scenarios without measuring the TCP throughput again. Previous studies, such as the ones carried out by Mirza et al. [17] and [18], demonstrate that history-based techniques are more accurate than formula-based. Authors in [19] claim and demonstrate that formula-based techniques are only accurate when the TCP flow does not saturate the path, and that using history-based prediction is only feasible when measurements of the system are available. Unlike us, the authors in [20] construct a time series based on measured segment windows at the receiver to predict future TCP throughput using different linear regressions. Other authors attempted to model TCP throughput as time-series using other tools for prediction, such as Support Vector Regression [17], neural networks [21,22], autoregresive and linear regression models [23,24]. To the best of our knowledge, there are no other similar works modeling TCP throughput as time-series and using GA for forecasting.

Current mobile terminals have several interfaces to connect to different networks such as WLAN and 2G/3G/4G. Although cellular networks such as 4G has a wide coverage area and can be seamless when performing horizontal handover, still the available capacity is often inadequate or it has a higher cost in terms of energy. On the other hand, WLAN provides higher data rates in its small radio coverage. When having multiple interfaces available, an important decision to make is when to change from one interface to another, which is called VHO. Protocols such as 802.21 or 802.11u uses VHO for seamless handover between networks of different types [25]. For example, when moving out of the coverage area of a WLAN access point (AP), the throughput typically goes down with the distance to that AP. At some point, the throughput will be zero and ideally, a handover occurs to e.g. a 4G network. Such handover can be based on SNR or achievable throughput. When the throughput goes down, one would like to initiate a handover in order to always be connected to the network providing the highest performance [26]. However, such handover strategy implies to have some knowledge of throughput estimates and ideally be able to

predict TCP throughput evolution over time. Moreover, mobile applications with throughput requirements such as video streaming can benefit from the predicted information to adapt their bitrate algorithms [27]. The problem lies in the difficulty to achieve this estimation due to many factors such as unpredictable link quality, unexpected interference situation or unknown traffic from other users who may be congesting the AP.

In a wireless environment, packet loss may occur due to non congestion-related effects such as biterrors, fading, wireless interference, etc., which are to a large extent hard to predict. Throughput studies and several measuring tools have been proposed for TCP in wireless environments. Franceschinis et al. [28] present a comprehensive study of the performance impact of TCP parameters such as the maximum congestion window. Bruno et al. [29,30] propose an analytical model and measurements for a WLAN persistent TCP-controlled download and upload data transfer and a wide-scope study of collision avoidance mechanisms of MAC protocols and TCP, respectively. Several forecast models have been proposed for WLAN and cellular networks: the autoregressive integrated moving average (ARIMA) model is the most commonly used for its simplicity [31]; however, algorithms based on the mean throughput [32] or neural networks [13,33] can also be used. In relation to computer networks, GA has many applications to serve as a meta heuristic for optimization purposes. For example in [14], the GA is used to calculate the best fit of a set of functions to a time series model which is used to describe the number of calls per minute of a switch centre with the aim of properly using the available resources for cognitive radio applications. Hence, the GA is used to find the best set of functions that relate past sample values with the future state of the network. Also, GA has also been applied in WiFi environments for different purposes such as for scheduling [34], congestion control [35] and optimization of wireless applications [36].

Therefore, in this paper we take the history-based approach to TCP throughput prediction. This is because we want to be independent of TCP intrinsic behavior and just use a history of measured samples of TCP throughput values over time in order to predict future throughput evolution. We apply GA based prediction techniques in order to best fit a set of functions to the given historic time series of measured TCP throughput values. Based on the GAs calculated best fit, we use the set of functions then to predict the TCP throughput over time. This work is a continuation of [37] where simulated traces where used to forecast TCP available bandwidth and study its relation to MAC busy time on different ON-OFF patterns governed by birth-death Makovian process. The good results obtained in the previous work on the prediction of simulated traces motivated us to use the GA tool from this work, improve it and use real traces to evaluate the prediction impact when several tuning parameters are changed; furthermore, different from [37], in this paper we study the possibility to reduce the frequency of retraining.

2.1. Time series analysis

Time series analysis [38–43] can be used to predict future values in a dynamic system. The theorem proposed by Takens [44] states that a non-linear chaotic dynamic system can be reconstructed from a sequence of observations. Therefore, having the following scalar time series $\{x_1, x_2, x_3, \ldots, x_{N_{samples}}\}$, obtained from observations during constant time intervals, it is possible to reconstruct a vector with embedding dimension m, into an m-dimensional space[45–47], as follows:

$$Z_i(m) = \left(x_i, x_{i+\tau}, \ldots, x_{i+(m-1)\tau}\right),\ Z_i \in R^m \quad i = 1, 2, \ldots, N_{samples} - (m-1)\tau \tag{1}$$

Here, Z_i is the reconstructed vector with the embedding dimension m, x_i is the observed discrete value at time i, τ is the time delay or embedding time and $N_{samples}$ is the length of the historical data series. The $m$coordinates of each Z_i and x_i are samples from the time series separated by a fixed τ. The result is a series of vectors

$$Z = Z_1, Z_2, .., Z_{N_{samples}-(m-1)\tau} \tag{2}$$

The idea of such reconstruction is to capture the original system states at each observation of the system output.

Applying this theorem to the problem to predict TCP available bandwidth, we assume that we are given a number of TCP throughput values sampled at different time instants as a sequence of discrete data points $\{x_t\}$, in chronological order. The aim is to study the time series behaviour in order to forecast the future evolution of the TCP throughput, up to a certain time horizon (also called prediction horizon).

2.2. Genetic algorithm

GA is a stochastic search method based on Darwins theory on natural selection and survival of the fittest. It has been applied to solve different optimisation problems without the necessity of finding an equation or series of steps to solve each problem. GA uses historical data, such as given by a time series, to find new points of search for an optimal solution of a problem, trying to improve the results and to converge into the best solution. The GA meta heuristic has three main processes or operators: selection, crossover and mutation. They are in charge of manipulating the current population in order to create optimal solutions for the problem to solve. These optimal solutions are tested over a set of time series samples called training set. Besides these operators, the fitness function plays an important role in the evaluation of the chromosomes because it influences the GA behaviour and its evolution. Fitness functions can be defined by several metrics that help to evaluate the goodness of the solutions over a training set.

The main general structure of the GA procedure is described in the following steps:

- Step 1: Randomly generate an initial population.
- Step 2: Evaluate each individual by means of the fitness function and sort them according to the selection method (i.e., by their fitness, calculated on the error from the real data).
- Step 3: Select the individuals for the reproduction (i.e., those with less error).
- Step 4: Through means of the crossover and mutation, new solutions are generated.
- Step 5: Evaluate the new population and repeat from step 3 until the termination criteria are met.

Once the algorithm terminates, the chromosome with the highest fitness within the current population is selected as the best solution to the original problem.

2.3. Forecasting

In this step, we look for a dependence of x_t on its N past values $\{x_{t-1}, x_{t-2}, \ldots, x_{t-N}\}$. The forecasting is done using time series analysis. Having an univariate time series $\{x_1, x_2, \ldots, x_t\}$ representing the observations (e.g. TCP throughput samples over time), it is possible to predict the next n points of this series (i.e., prediction horizon (ph), as the time interval $\{t+1, t+2, \ldots, t+n\}$ with a subset T of the previous samples (i.e., called the training set (ts)) [39,44,48].

The forecasting method used in this paper is the direct multistep-ahead prediction of several points, also known as independent value prediction in [49] or direct strategy in [50]. We apply Takens theorem [44] for the forecasting of the next samples,

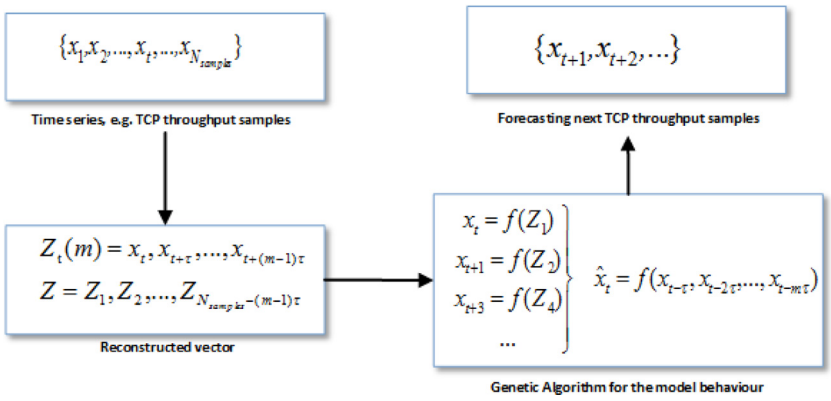

Fig. 1. Approach overview.

applying GA to look for an optimal function $f()$. The aim is to find a pattern in the past values $\{x_{t-\tau}, x_{t-2\tau}, \ldots, x_{t-m\tau}\}$ and use it to predict the future samples $\hat{x}_t^j$, as follows:

$$\hat{x}_t^j = f_j(x_{t-\tau}, x_{t-2\tau}, ..., x_{t-m\tau}) \quad m\tau + 1 \le t \le T \qquad 1 \le j \le N \tag{3}$$

Here, $\hat{x}_t^j$ represents the predicted sample at time instant t for the chromosome j in the current population; $T$is the length of the training set created from the time series of length $(m-1)\tau$ and Nrepresents the maximum number of chromosomes in the population.

We choose the direct prediction strategy because the error is not summing up in each iteration. When using the iteration prediction [50] instead, the predicted sample is included in every iteration and hence, the inherited error is added for the next prediction [49]. Although the direct strategy does not have this problem, it implies more computational resources because when increasing *ph* a larger *ts* is required to obtain a good prediction. The approach proposed in this paper is the combination of the time series analysis, GA and the forecasting method presented in Fig. 1.

3. Problem statement and approach

In a wireless scenario, different users using other wireless technologies in the same frequency band may cause interference. A prediction over the future state of the network may help to take appropriate decisions and ensure optimum use of available resources. The problem in a real environment lies in several factors that are affecting the TCP throughput such as multipath, shadow fading and interference. Therefore, it becomes difficult to model and predict with good accuracy the next state of the network using traditional time series models. Hence, it is important to select an appropriate tool to find a function that models the network behaviour in this dynamic and chaotic system. The GA was used before in chaotic settings such as [14], which motivated us to apply a similar technique to model and predict available TCP throughput from a time series of measurement samples. The idea is to let the GA find the best set of functions that when combined properly match the given time series in the best way and use that set of functions in order to predict future TCP throughput evolution over time.

In order to find the best matching functions, each potential solution is encoded in a chromosome that represents an individual in a population. The GA attempts to find a solution inside this search space where the chromosomes are manipulated by the GA operators (like crossover, mutation, etc.). Therefore, the first step is to define the rules to encode a chromosome through a set of functions. We have to create a valid mathematical expression which is able to evolve in the GA domain and once decoded can be verified to be a valid solution. As in [14], we use a combination of arguments (numerical values or past samples), functions $(cos(\theta), sin(\theta), ln(x), e^x)$ and arithmetic operations $(+, -, \times, \div)$. Moreover, the expressions are created using the reverse Polish notation [51] or also called *postfix*. Once the composition of the mathematical expression is defined, some rules must be followed in the encoded function in order to be able to decode these expressions [14]:

- The first and second position of the chromosome must be an argument and the last one an operator;
- At any position of the chromosome the number of arguments on the left must be greater than the operators;
- The chromosome must have the same number of arguments as operators plus 1.

The GA generates randomly an initial population of N chromosomes following the aforementioned rules. Yet, as these solutions are generated randomly, it is necessary to verify if they meet the rules and otherwise repair them. The verification and repair processes are also repeated after the crossover and mutation steps as in [14].

One of the most important tasks is the definition of an appropriate fitness function to properly evaluate the different solutions given by the meta heuristic. This is because different criteria and metrics can be used to set the fitness function and attempt to estimate the error or difference between the real and the predicted sample. While [14] uses a simple fitness function, we use several ones from the literature and analyze the impact of different fitness functions on the prediction quality. The main structure of a GA is depicted in Fig. 2, where the boxes in grey represent the functions that have been extended and evaluated in this paper.

3.1. Selection methods

The selection operator within the GA selects pairs of chromosomes in the population for reproduction. This is randomly done by favouring those chromosomes that have a better fitness. Several selection methods exist in literature such as the roulette-wheel selection (RWS) [52], rank-based roulette wheel selection (RRWS) [53], tournament selection [52] and exponential ranking wheel selection (ERWS) [54], among others. The authors in [14] used the RWS as the selection method even though the solution may not be the optimal one, because of its drawbacks such as the lower diversity and premature convergence of the population. For this reason, in Section 4.3 we will compare the prediction quality for different selection methods such as RWS, RRWS and ERWS.

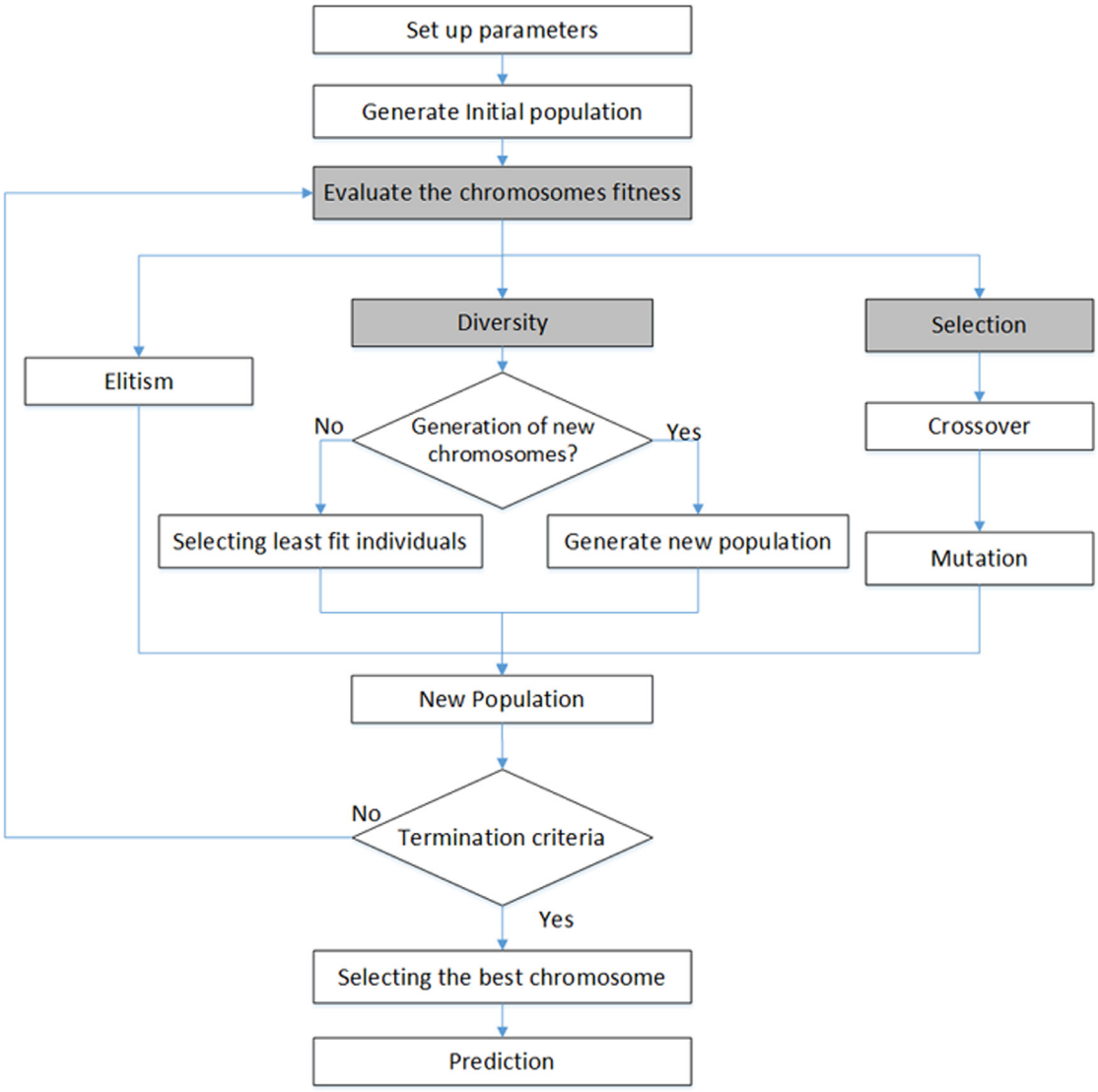

Fig. 2. GA structure approach. The functions extended and analyzed in this work are highlighted in grey.

The probability of selection (P_j^{RWS}) of a chromosome j using the RWS method is based on the fitness function of each chromosome and is calculated as:

$$P_j^{RWS} = \frac{FF_j}{\sum_{j=1}^{n} FF_j} \tag{4}$$

where FF_j is the fitness function of the chromosome j. Section 3.5 provides the details on how to calculate the fitness function.

The probability of a chromosome$j$ to be selected using the RRWS (P_j^{RRWS}) and ERWS (P_j^{ERWS}), the rank value is used as a fitness value. These two methods solve the problem when few chromosomes occupy most of the roulette wheel portion causing a big disadvantage for the remaining chromosomes. The probability of RRWS method [53] can be calculated as:

$$P_j^{RRWS} = \frac{2 - SP + \left(2 * (SP-1)\frac{(Pos_j - 1)}{(N-1)}\right)}{\sum_{j=1}^{N}\left[2 - SP + \left(2 * (SP-1)\frac{(Pos_j - 1)}{(N-1)}\right)\right]} \tag{5}$$
$$1.0 \leq SP \leq 2.0$$

Here, Pos_j is the position of the chromosome j, N the number of chromosomes, and $SP$the selection pressure. The position of the chromosome is obtained by sorting the population by their fitness value, where the fittest chromosomes will be in the first position and the least fit in the last position of the list.

The ERWS method uses as selection pressure the exponential weight (C), which controls the exponential degree. Therefore, the ERWS method tries to address the low convergence and the high diversity that RRWS presents due to the lower probability difference between the fittest and the least fit chromosomes:

$$P_j^{ERWS} = \frac{C^{N-Pos_j}}{\sum_{j=1}^{n} C^{N-Pos_j}} \tag{6}$$
$$0 < C < 1$$

The $SP$and Ccontrol the probability to select the chromosomes with higher or lower chance depending on their rank. However, a higher exponential weight (C) implies more equality while a higher SPimplies lower equality (i.e., lower probabilities to be selected for the chromosomes with a lower rank). The main advantage and drawbacks of using these selection methods are detailed in Table 1. The selection method to use is a trade-off between computational time i.e. the number of generations to converge, diversity of solutions and the feasibility or accuracy of the solution.

Table 1
Overview on selection methods.

	Advantage	Disadvantage
RWS	Probability depends on the fitness as occurs in nature Simple and widely used	Low diversity and premature convergence Scaling problems
RRWS	High diversity No scaling problems	Low convergence Computational resources
ERWS	Medium diversity No scaling problems	Low/medium convergence Computational resources

3.2. Elitism

The elitism operator keeps the best chromosomes during the crossover and mutation process, thus guaranteeing that the best chromosomes are going to survive and be present in the next generation [55]. This process takes the K best chromosomes as:

$$K = N \cdot elitism_{rate} \qquad \epsilon \mathbb{N}_o \tag{7}$$

where $elitism_{rate}$ is the percentage of elitism that ranges from 0 to 1.

3.3. Crossover

The crossover operator allows the exchange of features from one generation to the next and thereby the evolution of the species. The main objective is to get an improvement in the fitness for the next generation (offspring). During the crossover, the chromosomes selected for reproduction are paired up and crossed over. The crossover operator randomly selects one position along the chromosome and exchanges the part of the chromosome before and after that point of the two chromosomes to create the new offspring. This process is performed with a given probability, which fixes the number of chromosomes that are crossed over and therefore, the number of parents that will not survive.

3.4. Mutation

Once the crossover operator is finished, the mutation process is carried out to preserve and introduce diversity, i.e. to avoid premature convergence. This process ensures that the GA is not stuck in local minima, avoiding two consecutive populations to be very similar, and therefore allowing to diversify the solutions. The mutation process involves a number of random genes with a certain probability of mutation by randomly interchanging two values of a chromosome.

3.5. Fitness function

The fitness function in our use case evaluates the goodness of each chromosome by calculating the error between the real data samples from the time series and the training set (in our case this is the TCP expected bandwidth on the training set). The fitness function has a big impact on the solution quality because it determines at the end which chromosomes will survive. Several fitness functions have been proposed in the literature [38] and Table 2 summarizes the most frequently used metrics. In Section 4.3 we evaluate the performance of the prediction quality using different fitness functions as described below.

Several authors propose to use the sum squared error (SSE) between the prediction and the original sample [14,39,40]. Similarly, one can use the mean square error (MSE) for calculating the fitness according to:

$$FF_j^1 = \frac{1}{1 + MSE} \tag{8}$$

Table 2
Metrics for fitness function.

Metric	Acronym	Formula
Mean square error	MSE	$MSE = \frac{1}{P}\sum_{t=m\tau+1}^{T}(X_t^j - x_t)^2$
Mean absolute percent error	MAPE	$MAPE = \frac{1}{P}\sum_{t=m\tau+1}^{T}\lvert\frac{X_t^j - x_t}{x_t}\rvert$
Normalized mean square error	NMSE	$NMSE = \frac{\sum_{t=m\tau+1}^{T}(X_t^j - x_t)^2}{\sum_{t=m\tau+1}^{T}(x_t - x_{t+1})^2}$
Prediction on change	POCID	$POCID = \frac{100}{P}\sum_{t=m\tau+1}^{T} D_j$
in direction		$D_j = \begin{cases} 1 & (x_t^j - X_{t-1}^j)(x_t - x_{t-1}) > 0 \\ 0 & \text{otherwise} \end{cases}$
Average relative variance	ARV	$ARV_j = \frac{\sum_{t=m\tau+1}^{T}(X_t^j - x_t)^2}{\sum_{t=m\tau+1}^{T}(X_t^j - \bar{x})^2}$

While both fitness functions are very simple to evaluate, they may result in poor results due to the scarcity of the forecasted model information. One of the problems encountered using FF_j^1 is when two or more chromosomes have the same fitness value but when different trends can be observed. In this situation the GA may select one of them randomly without taking into account the trend of the solution which may lead to large errors.

By considering the Prediction Of Change In Direction (POCID) metric, Eq. (9) takes into consideration not only the error between the original sample and the prediction using the MSE, but also the trend of the model:

$$FF_j^2 = \frac{POCID}{1 + MSE} \tag{9}$$

The Normalized Mean Square Error (NMSE) can provide information regarding the deviations between predicted and measured values. Such information may contribute to point out the most noticeable differences among models. Therefore, another possibility is to use the NMSE instead of the MSE in Eq. (9):

$$FF_j^3 = \frac{POCID}{1 + NMSE} \tag{10}$$

The feasibility of the results can be improved when combining both the use of individual metrics (i.e., MSE, NSME) and POCID along with other metrics as the Mean Absolute Percent Error (MAPE) and the Average Relative Variance (ARV), as in the following expression:

$$FF_j^4 = \frac{POCID}{1 + MSE + MAPE + NMSE + ARV} \tag{11}$$

However, Eq. (11) may lead to dissimilar results because of the difficulty to satisfy the requirement of all metrics at the same time, e.g. high POCID, but low MSE and NMSE, etc.

Finally, the resulting fitness of each chromosome is multiplied by an exponential expression, Eq. (12), that depends on the number of historical samples (X_{total}) that the functions depend upon and the number of preferred samples (l_z). This exponential expression [14] results in a maximum of 1 when the chromosomes conform to the preferred number of historical samples. Otherwise, the exponential expression results in a number smaller than 1 and therefore, it reduces the fitness of those chromosomes that do not have the preferred number of historical samples. For example, by selecting a low l_z, we prefer chromosomes (functions) that only depend on a few number of historical samples. As a consequence, chromosomes that have less or more historical samples than l_z will be penalized more.

$$FF_j = FF_j * exp^{-abs(X_{total} - l_z)} \tag{12}$$

3.6. Diversity

Diversity is necessary in a GA since it introduces new solutions in the current population. Increasing the probability in the mutation process may lead to a random search because both the fittest

and least fit chromosomes may be affected by the randomness. Therefore, two diversity methods are implemented in this work to provide new potential solutions without compromising the proper GA functionality. One of the methods consists in the selection of the least fit D random chromosomes of the last population and injecting them into the current population [56]. In the other one, Drandom chromosomes are removed from the current population, and later new D random chromosomes are generated and introduced to the current population to be part again of the N population. The D random chromosomes are calculated as follows:

$$D = N - (offspring + K) \quad \in \mathbb{N} \tag{13}$$

where *offspring*is the number of offspring created in the crossover process and K is the number of elitism chromosomes.

3.7. Stopping criteria

The stopping criteria defines the condition, when the GA terminates. In this paper, we use two different stopping criteria. The first one is when the maximum number of generations is reached. For the second one, we calculate the maximum tolerable error based on the MAPE. In our case, we terminate the GA if we can find a good enough solution (a solution which has an error smaller than a given threshold) or when we reach the maximum number of iterations as given by the maximum number of generations.

3.8. Use of feedback during the forecasting

The use of updated information (e.g. based on actual measurements during the predictions) may help the GA to improve the quality. However, it may happen that it is not possible to include the measured samples or that they can be added only after a certain time. The update time (i.e., the time that the system needs to get the real predicted samples) may affect the reliability of future predictions. When we try to predict further ahead (i.e., the prediction horizon is greater than the training set) and take the predicted samples $\hat{x}$ as an input for future predictions, the error may be summing up and, therefore, the quality of the prediction may decrease. It may be more beneficial to use the measured samples rather than the predicted sample as input for future predictions. For example, if one tries to predict with $ts = T$ and $ph =$ n where x_{t-T} is the oldest sample that the system can take from the whole time series $\{x_1, x_2, \ldots, x_t\}$, at time $t + n$ the system may collect real data based on actual measurements $\{x_{t+1}, x_{t+2}, \ldots, x_{t+n-1}\}$ and use it as input data to function f_j for the next prediction at time $t + n + 1$:

$$\hat{x}^j_{t+n+1} = f_j(x_t, x_{t+1}, \ldots, x_{t+n}) \tag{14}$$

We will evaluate the impact of feedback based forecasting on the prediction quality in the evaluation section.

3.9. Retraining

As described in Section 2, GA has been used for short-term forecasting with reliable results in order to match a given set of functions to a time series of sampled measurement points. Typically, once the set of functions is found, those equations can be used to forecast or predict future time series points. However, the characteristic of the time series may change over time so that a once found set of functions may be not a good fit for the time series in future instances. To increase the accuracy of the prediction, we may apply a retraining scheme, for which the GA may compute the best matching set of equations every δt. Clearly, retraining more frequently leads to a more computationally heavy scheme leading to a potential better match between the real measured data and the predicted ones. We will evaluate the impact of retraining on the prediction quality in the evaluation section.

Table 3
GA default parameters.

Parameters	Description	Value
p_c	Crossover probability (Single point)	0.7
p_m	Mutation probability (Single point)	0.05
elitism	Elitism percentage	0.1
generation	Number of generations	100
N	Number of population	100
T	Length of the *ts* (samples)	10 or 30
l_z	Number of preferred time series values in the chromosome	1
mating pool	Mating pool size	0.8

4. Numerical evaluation

In this section, we perform a series of numerical evaluations varying, among others, fitness functions and selection methods described in Section 3. First, we provide the details of the scenario and the evaluation setup in Section 4.1. Then, we discuss our results and study the effect of different settings on the prediction quality.

4.1. Evaluation setup

We aim at evaluating the suitability for using GA to predict available TCP bandwidth that is given by time series measurements. We want to fit a set of mathematical equations that operate on the time series in order to match the TCP available bandwidth and study the impact of different fitness functions, selection methods and prediction horizons. We implemented the GA in Matlab and set-up the GA with the standard values presented in [14,57,58], see Table 3. All GA tests are run using the same common parameters while the training set is varied *ts* depending on the scenario. In addition, one selection method and one fitness function is selected in each test to study the effect on the prediction quality. We perform 50 repetitions for each test and calculate the average over all repetitions.

We obtained real TCP throughput samples which are then used by the GA as input to create the time series of samples that are used to fit a set of functions. To get the throughput sample values, we used an IEEE 802.11 client in the public library in Karlstad university, where interference from other devices is common. The TCP throughput is measured every 100 ms at the client side (i.e., one sample is equivalent to 100 ms in the rest of the paper). Among the huge amount of data that we obtained, we selected two sets of 60 samples of TCP throughput representing: 1) a scenario with a more regular pattern (Scenario A, see Fig. 3) and 2) a scenario with abrupt changes in the trend (scenario B, see Fig. 4). These scenarios are investigated in order to 1) test the ability of the proposed GA to follow smoother or sudden changes in the throughput pattern, and 2) assess the prediction error using different settings in the GA (e.g, selection methods, fitness functions, training set, prediction horizon, etc.). The results of this first evaluation are presented in Section 4.3. Then, we select one fitness function, one selection method and one scenario and we further study the impact of using feedback during the prediction in Section 4.4. Finally, in Section 4.5 the impact of retraining on the prediction quality is studied.

4.2. Scenarios

The two scenarios are depicted in Fig. 3 (scenario A) and in Fig. 4 (scenario B). The input data (measured TCP throughput as a time series) is drawn with a blue solid line. The training set (*ts*) constitutes 30 samples (from sample 1 to sample 30) and the prediction horizon (*ph*) is also set to 30 samples (i.e., the GA predicts

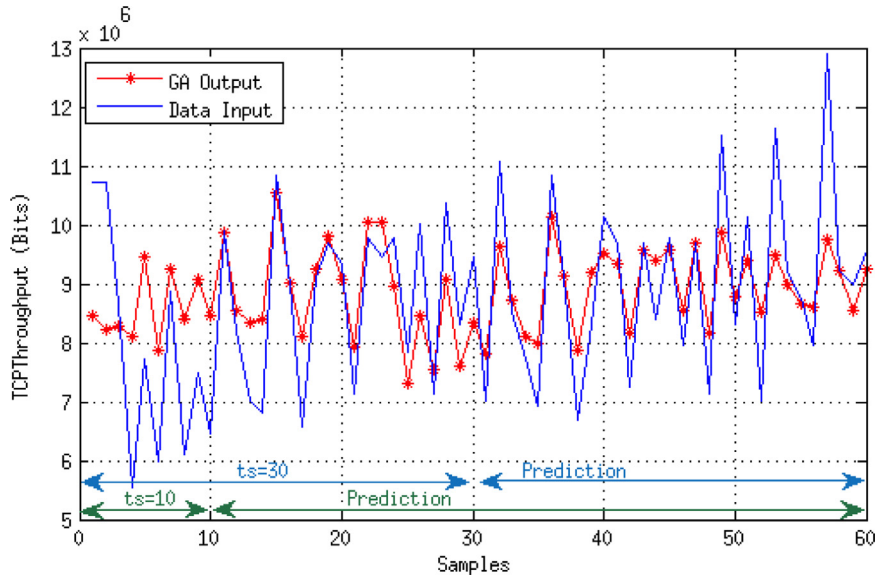

Fig. 3. Input data and GA output in Scenario A for FF_j^2 and RRWS. Different training sets are shown ($ts = 10$ and $ts = 30$).

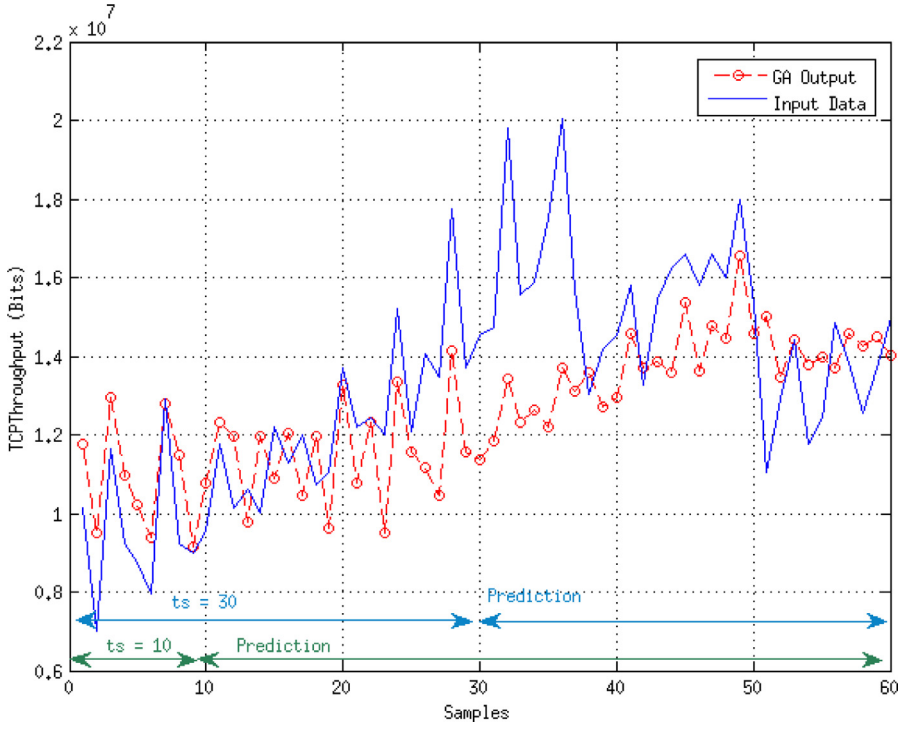

Fig. 4. Input data and GA output in Scenario B for FF_j^2 and RRWS. Different training sets are shown ($ts = 10$ and $ts = 30$).

the next 30 samples of the TCP throughput, from sample 31 to sample 60). The prediction obtained with FF_j^2 (Eq. (9)) and with RRWS as the selection method is also displayed (red dotted line) for both scenarios. Further in the evaluation, we will also show the case when the *ph* is set twice or four times the *ts*; in that case, we will predict from sample 31 to sample 90 or 150, respectively (for simplicity, samples from 60 to 150 are omitted in Figs. 3 and 4). Also, we will show results when *ts* is set to 10 samples; in that case, the *ts* goes from sample 1 to sample 10. For a *ph* of 10, samples 11 to 20 are predicted, while we will predict up to sample 50 when the *ph* is four times the *ts*.

As shown in Fig. 4, the TCP throughput trend in scenario B is increasing from sample 1 to 36 due to e.g. better conditions on the wireless channel or lower interference conditions. Then, the TCP throughput decreases abruptly around sample 36 and, again, around sample 50 due to e.g. more interference. It is important to detect these changes in the trend. Also, it would be interesting to estimate the duration of such changes.

4.3. Impact of the selection method and fitness function

In this section, we evaluate the prediction error for the given fitness functions and selection methods (see Sections 3.1 and 3.5). We select the MAPE as the evaluation metric since it is scale-independent of the input data range and the calculation results in a percentage expression.

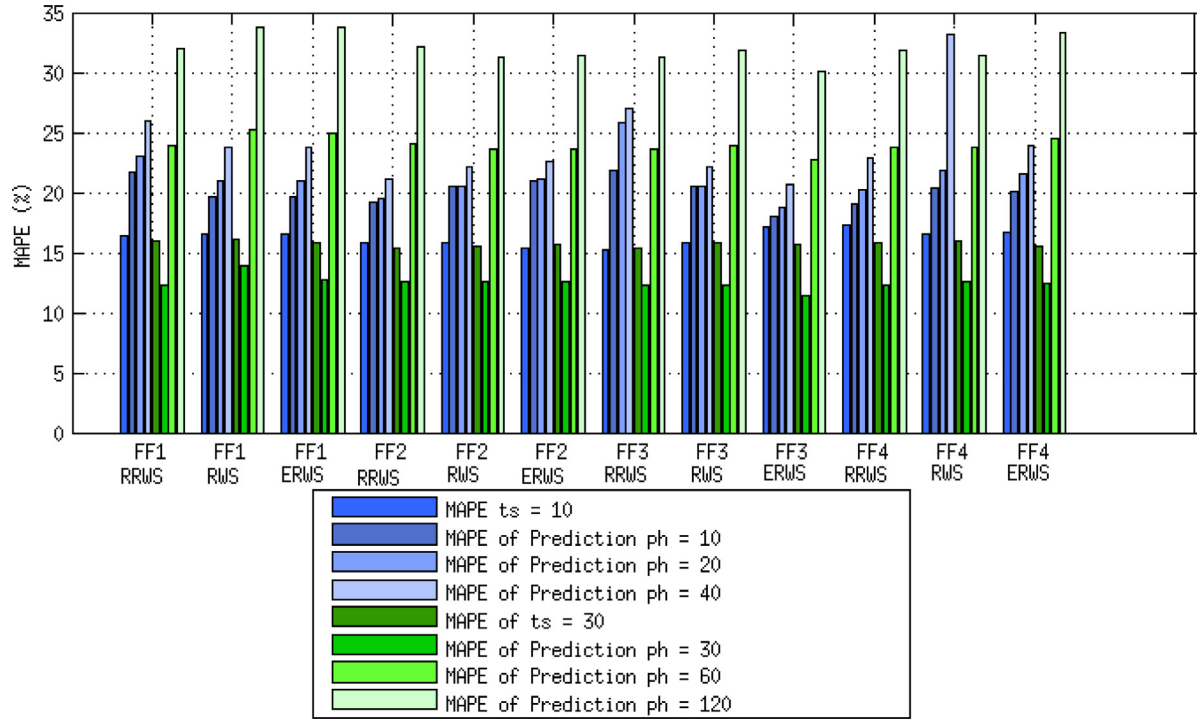

Fig. 5. MAPE for different selection methods and fitness functions using a training set of 10 samples vs. 30 samples and for different prediction horizons (scenario A). (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

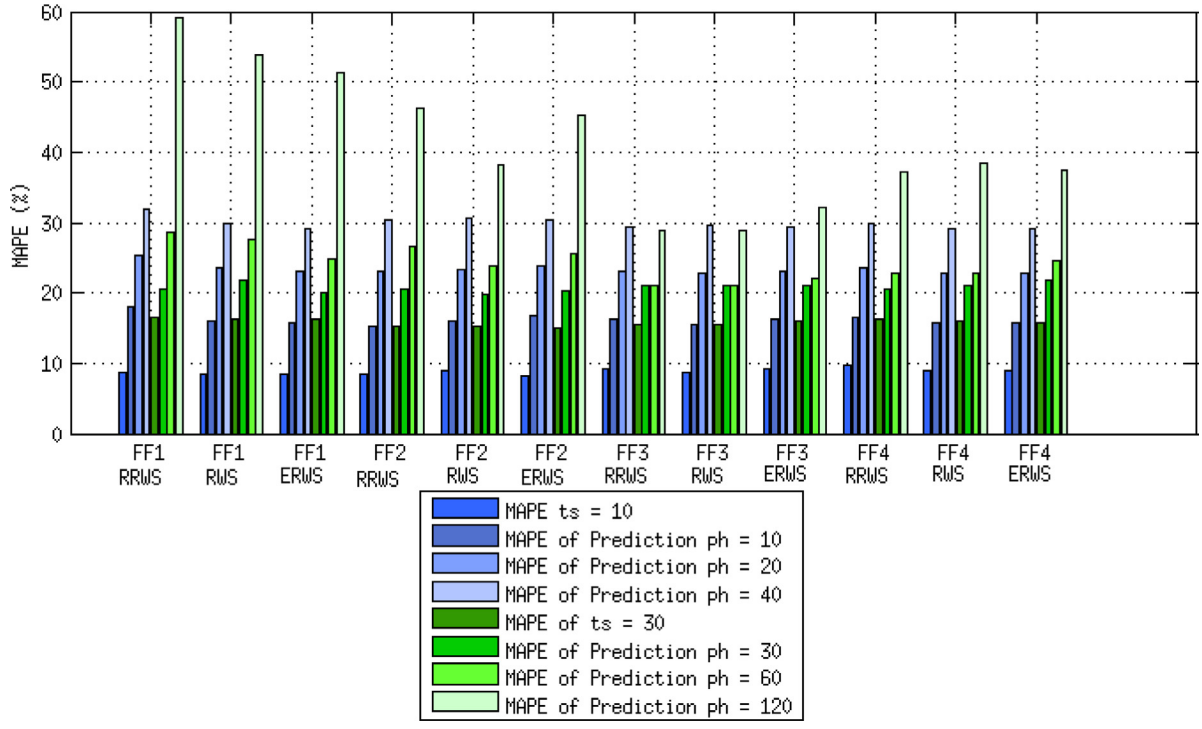

Fig. 6. MAPE for different selection methods and fitness functions using a training set of 10 samples vs. 30 samples and for different prediction horizons (scenario B). (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

Fig. 5 and 6 show the MAPE between the given input data and the predicted samples for different settings of the GA in scenario A and B, respectively. In the x axis, each fitness function (i.e., FF_j^1 to FF_j^4) is evaluated for three selection methods (i.e., RRWS, RWS and ERWS). The four blue bars (darker bars in b/w printing) represent the MAPE when the *ts* is set to 10 samples; the first bar represents the MAPE of the training set (i.e., the error between the input data and the output of the GA during the training), while the second bar depicts the MAPE of the prediction when the *ph* is set to 10, as the *ts*. We can observe that, for scenario A, the MAPE of the *ts* is always between 15% and 17%, while the MAPE of the prediction is always higher (i.e., between 18% and 22%). Although it is not clear whether one fitness function or one selection method

performs better than the others, it seems that the combination of FF_j^3 with ERWS shows a smaller overall gap. On the other hand, scenario B shows better results regarding the training set: the MAPE of the *ts* is always lower than 10% and that of the prediction stays between 15% and 19%.

Once the GA finds an equation that properly fits the training set, the same amount of samples (i.e., $ph = ts$) can be predicted with good fidelity using that same equation; if one would like to predict for longer horizons, it is recommended to either retrain the GA [14,59] or increase the training set at the expense of higher computational complexity. While the impact of retraining is further investigated in Section 4.5, in this section we analyze the impact of using a larger prediction horizon, i.e. $ph > ts$. Using a large prediction horizon raises the question what is the impact regarding prediction quality of using a single equation for time series values that are further away in the future. On the other hand it is not clear, how far we can predict using the same equation with reasonable fidelity.

To further study those questions, we plot the MAPE when the *ph* is twice the *ts* and when it is four time the *ts* in Fig. 5 and 6 for each fitness function and for each selection method. As expected, the MAPE increases with larger *ph*; however, in the smoother scenario A, when we predict for a horizon twice the *ts*, the increase in the error is very small. For some combinations (i.e., FF_j^2, and FF_j^3 with RWS or ERWS), the MAPE is almost the same for $ph = ts$ and $ph = 2ts$. In Section 4.4 we will investigate further on this interesting result and we will study the effects on the time needed for the prediction.

Finally, we want to study the impact, in terms of the prediction error, when longer training sets are used. Although more computational resources are needed, the authors in [60,61] demonstrate that the longer the training set, the more accurate will be the equation obtained with the GA during the training. Following the statement given above, using a longer *ts* we expect a more accurate prediction. Thus, we use a *ts* of 30 samples and compare the results with those obtained with *ts*=10. Fig. 5 and 6 show the results for $ts = 30$ in green (lighter grey scale in b/w printing). First, we observe that in scenario A the MAPE of the *ts* when $ts = 30$ is always smaller or equal to that when *ts*=10. Also, the MAPE of the prediction with $ph = ts$ is always smaller than the MAPE of the *ts*, thus confirming what found in the literature. Also, when the prediction horizon is increased, the performance drops drastically. On the other hand, the results in scenario B are always worst when the *ts* is increased from 10 to 30 and $ph = ts$. However, when using $ph = 2ts$, there are some combination of the fitness function and the selection method for which the performance can even improve (i.e., FF_j^3, and FF_j^4 with RRWS or RWS). Again, with $ph = 4ts$ the performance drastically drops. However, when $ts = 30$ and $ph = 4$ts, the absolute number of samples we try to predict is much larger than compared to the case when $ts = 10$ and $ph = 4$ts. From our results we can conclude that a longer training set not always leads to better predictions, as this performance is also tied to the trend in the dataset. While longer *ts* is preferred for regular trends, a more irregular dataset may not necessarily benefit from a longer training as the prediction errors are not significantly reduced and more computational resources are required.

4.4. Impact of limiting the sample set and of feedback

[14] suggests to use the newest samples of the input data (i.e., most recent data) to validate the fitness of the equations on the training set. Using the newest data should better follow the trend of the input. In contrast, using old data may lead to erroneous predictions. In this section we study whether imposing a limit on how far in the past one can go (i.e., how old can be the samples used for the training phase) may have an impact on the quality of the prediction. We set four different constraints: 10, 20, 30 and 40 samples; e.g., with a limitation of 10 samples (i.e., *lim*10), the GA can take any sample among the 10 newest historical samples to generate an equation. We compare the results with the case where no limitation is used (i.e., *no_lim*), meaning that the GA can take any sample among the 150 newest samples. Also, from now on, the fitness function is set to FF_j^2 and the selection method to RRWS; only scenario A is further investigated in this and the following sections. Thus, the four blue bars presented in Fig. 5 for FF_j^2 and RRWS and for *ts* equal to 10 now reappear in Fig. 7 as "No lim" (green). The four bars in blue in Fig. 7 represent the MAPE of *lim*10 (dark blue), *lim*20 (blue), *lim*30 (turquoise), *lim*40 (light blue). The standard deviation (stdv) is also displayed for each bar. In general, when a limitation is introduced, an improvement can be observed. With *lim*10, the average MAPE and the stdv decrease if $ph = 2ts$, while the stdv increase for $ph = 4ts$. When older samples are included (i.e., *lim*20 and *lim*30), the MAPE and stdv decrease; however, when too old samples are included (i.e., *lim*40) the MAPE increases again, thus masking the benefits of the limitation.

Fig. 7 also shows the results when feedback is applied to the GA: *lim*10*fdb* (dark red), *lim*20*fdb* (crimson), *lim*30*fdb* (red), and *lim*40*fdb* (light red). That is, when the *ts* is set to 10 and at time $t = 0$ we want to predict during $ph = 2ts$=20, at time $t = ts$ the system may have collected the real data based on actual measurements during $\{0, 1, 2, \ldots, ts - 1\}$, so that it can use the real data instead of the predicted samples for further predicting the next *ts* samples (i.e., from $t = ts$ to $t = 2ts - 1$).

In this way, the prediction error does not accumulate, as explained in Section 3.8. As shown in Fig. 7, when $ph = ts$=10 we obtain similar results when applying feedback or not, as expected. In contrast, the more we try to predict the future (increase *ph*), the more using feedback reduces the MAPE and its standard deviation. For longer prediction horizons we can conclude that it is worth limiting the sample set to the newest values (i.e., *lim*10) if feedback can be employed.

4.5. Impact of retraining

Due to the high computational cost, it is essential to exploit as much as possible the resulting set of functions for predicting once the GA has found a solution. However, the use of the same function to predict over long-term periods may lead to a loss in prediction quality and, therefore, an increase in the prediction error. We intend to study the evolution of the prediction error when the same function is employed to predict samples that are further away than the prediction horizon (i.e., $ph > ts$) and the impact of the retraining on the time needed to find a solution.

We want to predict up to e.g. 80 samples (i.e., $ph = 80$, and $ts = 10$) applying different retraining schemes as follows: retrain every 10, 20, 30 and 40 samples. When we retrain every 10 samples (*ret*10), after the first 10 predictions are obtained, the GA is trained again over the last 10 real data samples (i.e., feedback as explained in Section 4.4) which provides a new set of functions, which is then used to predict the next 10 samples.

When *lim*10 with feedback was selected, we could observe a positive effect on the quality of the prediction since unless there is retraining, real samples are used every 10 samples instead of predicted samples. When a retraining scheme of 20 samples (*ret*20) is used, after the first 20 predictions are obtained, the GA is trained again. However, as feedback is also applied, after the first 10 predictions the GA uses the last 10 real samples to feed its equation and predict the other 10 samples. Then, after retraining, the new equation is used to predict the next 20 samples (10 plus 10 with

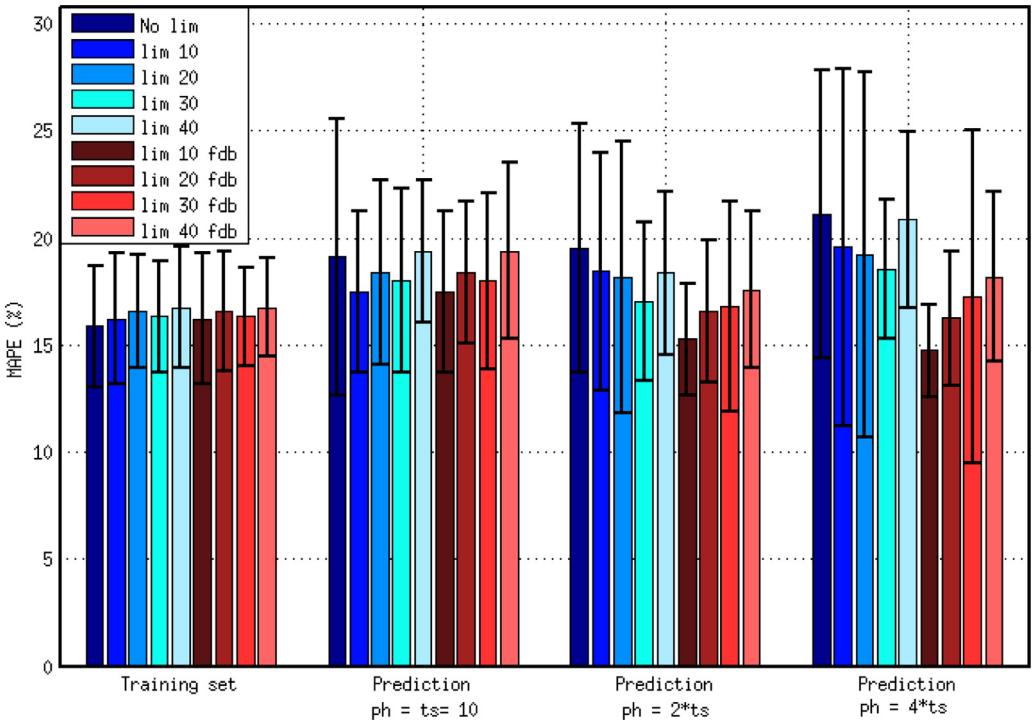

Fig. 7. MAPE when applying a limit on the oldest sample (scenario A, FF_j^2, RRWS, ts=10). (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

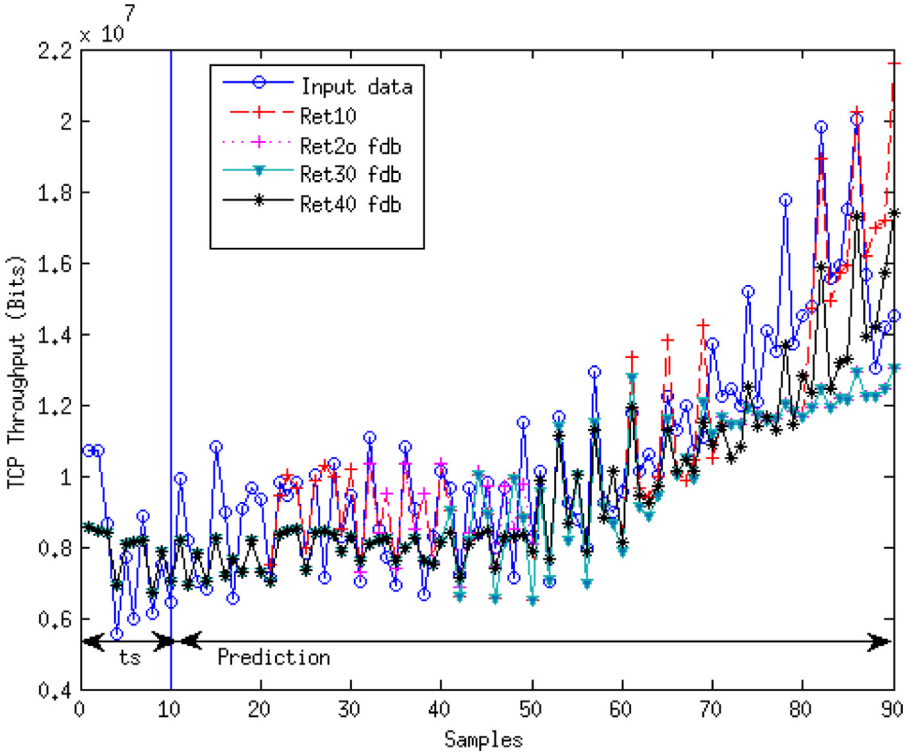

Fig. 8. Original and predicted samples using different retraining schemes with feedback (scenario A, FF_j^2, RRWS, $ts = 10$).

feedback). Therefore, the feedback process is applied every 10 samples for *ret*20, *ret*30 and *ret*40.

The impact of retraining and feedback using the different retrain schemes is illustrated in Fig. 8. This figure shows the mean predicted samples and the original data (Input data) when the retrain schemes of 10, 20, 30 and 40 samples are applied with feedback. Note, that from sample 10 to 20 all schemes uses the same function, thus the predicted samples are the same for all schemes. After sample 20, *ret*10 uses a new function to predict up to sample 30 and the other schemes update the real predicted samples from sample 10 to 20. At sample 30, ret10 scheme uses a new function along with *ret*20. *Ret*30 and *ret*40 use the new function

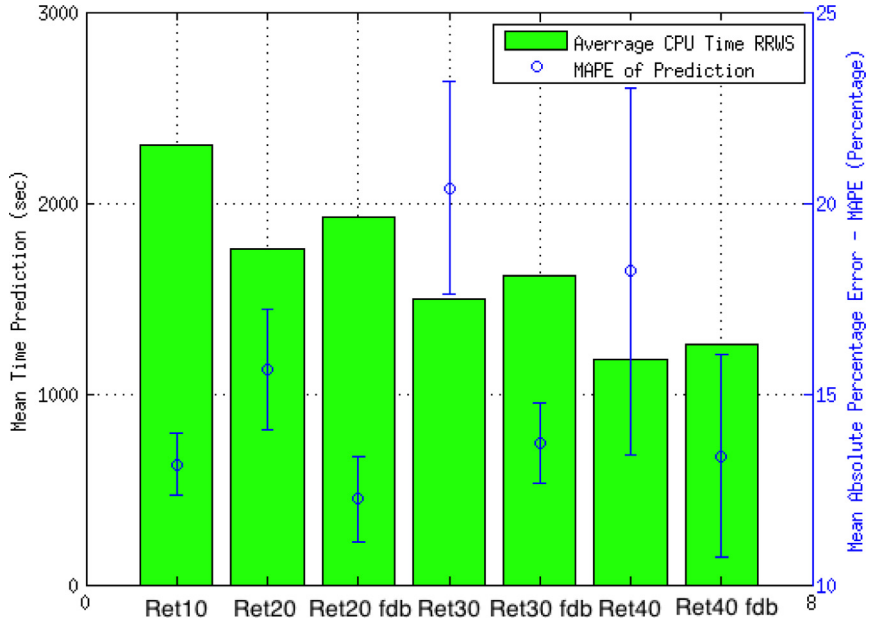

Fig. 9. MAPE and required time for different retraining schemes with and without feedback (scenario A, FF_j^2, RRWS, ts =10).

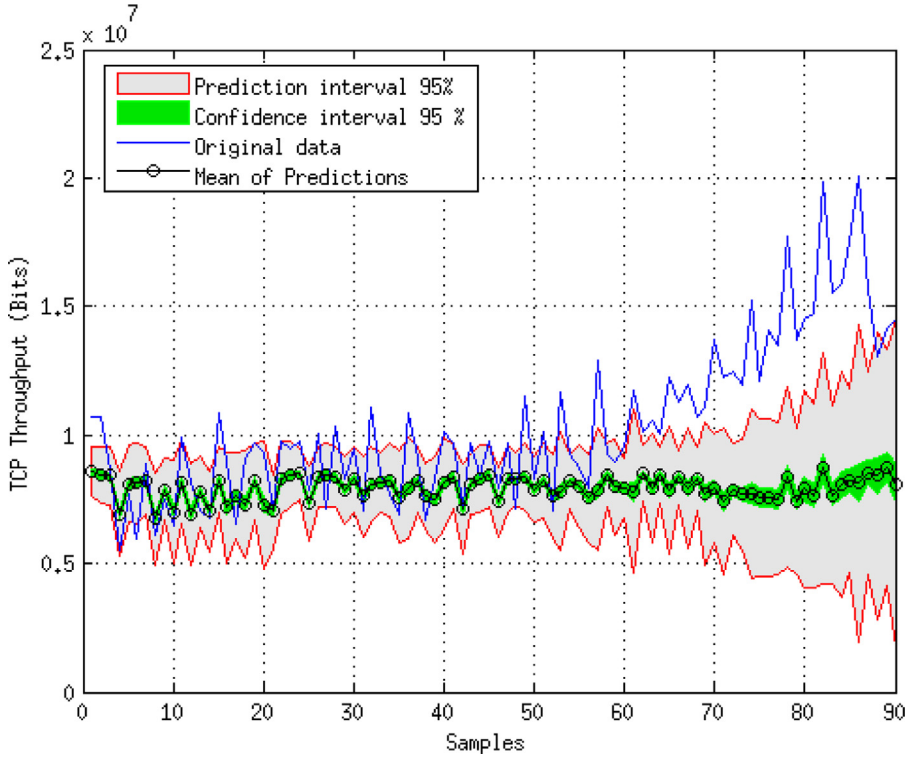

Fig. 10. 95% confidence interval and predicted interval for GA without retraining (scenario A, FF_j^2, RRWS, ts = 10).

at sample 40 and 50, respectively. When the retraining frequency is reduced, the MAPE variation increases due to the decrease in the prediction accuracy. Although better results are obtained with a retraining scheme of 10 samples, the difference between the mean MAPE of the retrain scheme of 10 samples and the retrain scheme of 40 samples with feedback is only about 8%. Moreover, the difference between the mean MAPE of the retrain schemes with feedback does not exceed 4%. Therefore, when a retraining scheme is selected along with feedback, the difference between 10 samples retrain and 40 samples retrain is reduced. Despite this small difference below 1%, we can see that the difference in the prediction interval increases for these schemes.

Fig. 9 shows the mean prediction time (the time to run the algorithm to find the solution) versus the MAPE and its standard deviation for the four retraining schemes and with or without feedback. When we use feedback, we need to update the predicted samples by using the real measured samples. Adopting feedback consumes more CPU to calculate a solution. However, retraining needs more CPU resources and time than adopting a feedback method. Retraining very often (i.e., *ret*10) requires

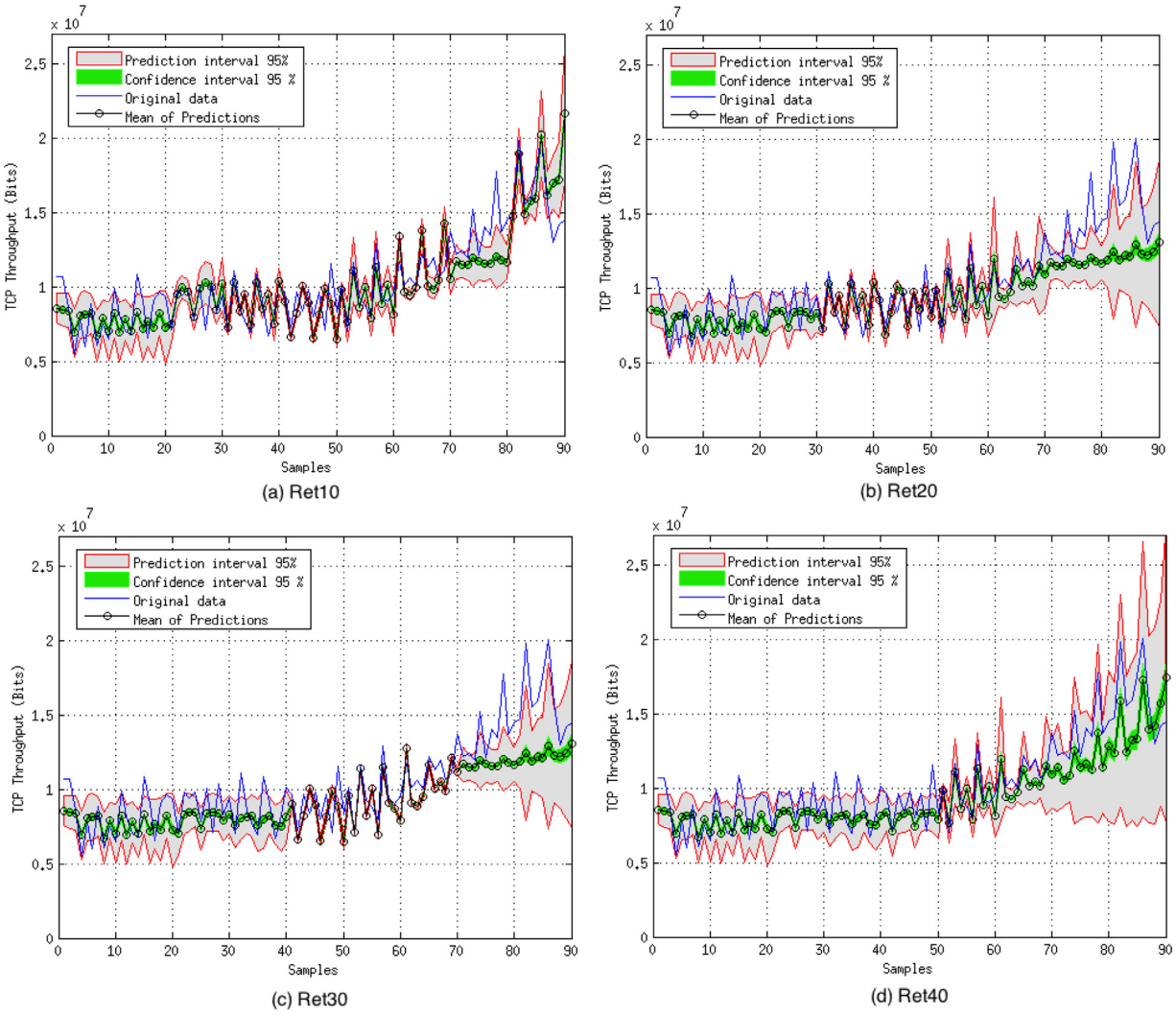

Fig. 11. 95% of confidence and predicted intervals for GA with ret10, ret20_fdb, ret30_fdb and ret40_fdb (scenario A, FF_j^2, RRWS, ts =10).

the highest CPU time but results in the smallest MAPE. On the other hand, retraining every 40 samples consumes less time but at the expense of a reduced prediction quality. Note, that *ret*30 with feedback provides a good tradeoff as the MAPE is comparable with the one in *ret*10, while the time required to find the solution is around 33% less than in *ret*10. Also, it is evident from Fig. 9 that the feedback always benefits the accuracy of the prediction.

Finally, we were interested how confidence intervals over time evolve when using different retrain schemes and prediction intervals. As a consequence, we calculate the range of the mean prediction samples given by the set of functions calculated by the GA. We calculating the 95% confidence interval, i.e. the area where, with 95% confidence, the mean predicted samples will be located. This statistical measure gives important information about the accuracy of the prediction. On the other hand, the prediction interval provides information about the distribution of the predicted samples. The prediction interval is based on the past observations and shows, with a certain probability, where we can expect any future sample. This range is always wider than confidence interval as it considers the uncertainty on the mean value and its distribution properties. Fig. 10 shows the 95% confidence interval and the prediction interval calculated from the resulted predictions when the input data from Fig. 8 is used.

Fig. 11a illustrates the effect on the confidence interval and prediction interval when a retraining scheme of 10 samples is applied. In this case, as can be compared with Fig. 10, both confidence and prediction interval are narrower since this retrain scheme provides more accurate results at the expense of more CPU resources to find the soution (also compare Fig. 9).

When we use the same function to predict further ahead we can expect less accuracy and uncertainty on the results. For example, when the training set is composed of bandwidth samples that conform to a more steady throughput, it will be hard to predict a sharp increase in available bandwidth later on. This can be seen from Fig. 11 where we show the confidence and predicted intervals for a retraining scheme using different samples (10, 20, 30 or 40). After sample 70, the TCP throughput starts to increase exponentially, which makes it hard to predict without proper retraining. Although the error increases significantly after sample 70, mostly the prediction interval covers the input data, meaning that the algorithm is able to follow successfully the trend.

When a retraining scheme of less frequency is selected (i.e., *ret*30 or *ret*40) as in the case of 30 samples, depicted in Fig. 11 c), the effect in the confidence and prediction intervals remain similar to the case of 20 samples, since the retraining has been applied from sample 60 and 70, i.e. before the TCP throughput rises sharply. When a retraining scheme of 40 samples is applied, the

prediction band is consistent across all input samples as Fig. 11 d) illustrates. In this scenario, the GA is retrained only once (i.e., at sample 50). However, the prediction interval is quite wide, but covering mostly all the input data, including the area where the throughput rises sharply. Despite we are using a less frequent retraining scheme compared to 20 or 30 samples, we can much better follow the trend because the GA is retrained in a different area (starting at sample 50). We conclude that even though retraining with higher frequency may reduce the uncertainty and provide higher accuracy, it is important to retrain at specific time intervals in order to capture changes in input statistics.

5. Conclusion

The difficulty of predicting the TCP throughput in interference prone WiFi environments is challenging because of different unpredictable effects such as interference, multipath or other users traffic leading to collisions and unpredictable available capacity. In this paper, we propose to model measured TCP throughput samples as a time series and apply the meta heuristic genetic algorithm to match a set of mathematical functions to best represent the time series. By using the set of functions one can predict future samples, given the GA is trained properly. Using our strategy, one can effectively predict TCP throughput evolution over time by just looking at measured throughput samples without the need to have information available from the TCP stack such as estimates on e.g. round-trip-time or packet loss. We have evaluated the impact of different fitness functions and selection algorithms on the accuracy of predictions. When a more accurate prediction is needed, different retraining schemes can be applied at the expense of more computational power required to find the best set of matching functions. Finally, we have demonstrated that the use of feedback strategies always increases the accuracy of the prediction. In order to improve the accuracy even more, a good strategy has to be found that determines when retraining should be applied.

As a future work, we intend to develop heuristics that guide when a retraining should be executed, for example based on knowledge about the TCP congestion control phase. Also, we want to study the impact of different sampling intervals on prediction quality as well as study more scenarios such as different interference situations, different bottleneck links, etc. and their impact on the quality of the prediction.

Acknowledgment

This research was supported by the Spanish Government and ERDF through CICYT project TEC2013-48099-C2-1-P.

References

[1] P. Cortez, M. Rio, M. Rocha, P. Sousa, Multi-scale internet traffic forecasting using neural networks and time series methods, Expert Syst. 29 (2) (2012) 143–155, doi:10.1111/j.1468-0394.2010.00568.x.
[2] D. Wischik, C. Raiciu, A. Greenhalgh, M. Handley, Design, implementation and evaluation of congestion control for multipath TCP, in: Proceedings of the 8th USENIX Conference on Networked Systems Design and Implementation, in: NSDI'11, USENIX Association, Berkeley, CA, USA, 2011, pp. 99–112.
[3] C. Paasch, S. Ferlin, O. Alay, O. Bonaventure, Experimental evaluation of multipath TCP schedulers, in: ACM SIGCOMM Capacity Sharing Workshop (CSWS'14), ACM, 2014, pp. 27–32.
[4] M. Scharf, Multipath TCP (MPTCP) application interface considerations, Request for Comments: 6897, Alcatel-Lucent Bell Labs, 2013.
[5] J. Padhye, V. Firoiu, D. Towsley, J. Kurose, Modeling TCP throughput: a simple model and its empirical validation, ACM SIGCOMM Comput. Commun. Rev. 28 (4) (1998) 303–314.
[6] M. Mathis, J. Semke, J. Mahdavi, T. Ott, The macroscopic behavior of the TCP congestion avoidance algorithm, ACM SIGCOMM Comput. Commun. Rev. 27 (3) (1997) 67–82.
[7] C. Dovrolis, P. Ramanathan, D. Moore, Packet-dispersion techniques and a capacity-estimation methodology, IEEE/ACM Trans. Netw. 12 (6) (2004) 963–977.
[8] C. Dovrolis, P. Ramanathan, D. Moore, What do packet dispersion techniques measure? in: INFOCOM 2001. Twentieth Annual Joint Conference of the IEEE Computer and Communications Societies, vol. 2, IEEE, 2001, pp. 905–914.
[9] C.W. Ahn, R.S. Ramakrishna, A genetic algorithm for shortest path routing problem and the sizing of populations, IEEE Trans. Evol. Comput. 6 (6) (2002) 566–579.
[10] Y.-S. Yen, H.-C. Chao, R.-S. Chang, A. Vasilakos, Flooding-limited and multi–constrained qos multicast routing based on the genetic algorithm for MANETs, Math. Comput. Model. 53 (11) (2011) 2238–2250.
[11] H. jun Yang, X. Hu, Wavelet neural network with improved genetic algorithm for traffic flow time series prediction, Optik - Int. J. Light Electron Optics 127 (19) (2016) 8103–8110. http://dx.doi.org/10.1016/j.ijleo.2016.06.017.
[12] Y. Chen, B. Yang, Q. Meng, Y. Zhao, A. Abraham, Time-series forecasting using a system of ordinary differential equations, Inf. Sci. 181 (1) (2011) 106–114. http://dx.doi.org/10.1016/j.ins.2010.09.006.
[13] W. Lu, Parameters of network traffic prediction model jointly optimized by genetic algorithm, JNW 9 (2014) 695–702.
[14] D. Thilakawardana, K. Moessner, Traffic modelling and forecasting using genetic algorithms for next-generation cognitive radio applications, Ann. Telecommun. - Annales des télécommunications 64 (7–8) (2009) 535–543.
[15] J.-H. Hwang, C. Yoo, Formula-based TCP throughput prediction with available bandwidth, Commun. Lett. IEEE 14 (4) (2010) 363–365.
[16] P. Loiseau, P. Gonçalves, J. Barral, P.V.-B. Primet, Modeling TCP throughput: An elaborated large-deviations-based model and its empirical validation, Perform. Eval. 67 (11) (2010) 1030–1043.
[17] M. Mirza, J. Sommers, P. Barford, X. Zhu, A machine learning approach to tcp throughput prediction, IEEE/ACM Trans. Netw. 18 (4) (2010) 1026–1039, doi:10.1109/TNET.2009.2037812.
[18] Q. He, C. Dovrolis, M. Ammar, Prediction of TCP throughput: formula-based and history-based methods, ACM SIGMETRICS Perform. Eval. Rev. 33 (1) (2005) 388–389.
[19] Q. He, C. Dovrolis, M. Ammar, On the predictability of large transfer TCP throughput, Comput. Netw. 51 (14) (2007) 3959–3977.
[20] T. i Huang, J. Subhlok, Fast pattern-based throughput prediction for TCP bulk transfers, in: CCGrid 2005. IEEE International Symposium on Cluster Computing and the Grid, 2005, vol. 1, IEEE, 2005, pp. 410–417.
[21] P. Cortez, M. Rio, M. Rocha, P. Sousa, Multi-scale internet traffic forecasting using neural networks and time series methods, Expert Syst. 29 (2) (2012) 143–155.
[22] S. Gowrishankar, P. Satyanarayana, A time series modeling and prediction of wireless network traffic., iJIM 3 (1) (2009) 53–62.
[23] R.P. Karrer, Tcp prediction for adaptive applications, in: 32nd IEEE Conference on Local Computer Networks (LCN 2007), IEEE, 2007, pp. 989–996.
[24] B. Zhou, D. He, Z. Sun, Traffic predictability based on arima/garch model, in: 2006 2nd Conference on Next Generation Internet Design and Engineering, 2006. NGI'06., IEEE, 2006, pp. 8–pp.
[25] O. Khattab, O. Alani, Algorithm for seamless vertical handover in heterogeneous mobile networks, in: Science and Information Conference (SAI), 2014, pp. 652–659, doi:10.1109/SAI.2014.6918256.
[26] E. Zola, P. Dely, A.J. Kassler, F. Barcelo-Arroyo, Robust association for multi-radio devices under coverage of multiple networks, in: 11th International Conference Wired/Wireless Internet Communication (WWIC), Proceedings, 2013, pp. 70–82, doi:10.1007/978-3-642-38401-1_6.
[27] Y. Sun, X. Yin, N. Wang, J. Jiang, V. Sekar, Y. Jin, B. Sinopoli, Analyzing TCP throughput stability and predictability with implications for adaptive video streaming, CoRR abs/1506.05541 (2015).
[28] M. Franceschinis, M. Mellia, M. Meo, M. Munafo, Measuring TCP over WiFi: A real case, 1st Workshop on Wireless Network Measurements (Winmee), Riva Del Garda, Italy, 2005.
[29] R. Bruno, M. Conti, E. Gregori, Performance modelling and measurements of TCP transfer throughput in 802.11-based WLAN, in: Proceedings of the 9th ACM International Symposium on Modeling Analysis and Simulation of Wireless and Mobile Systems, ACM, 2006, pp. 4–11.
[30] R. Bruno, M. Conti, E. Gregori, Throughput analysis and measurements in IEEE 802.11 WLANs with TCP and UDP traffic flows, Mob. Comput. IEEE Trans. 7 (2) (2008) 171–186.
[31] W. Yoo, A. Sim, Network bandwidth utilization forecast model on high bandwidth networks, in: Computing, Networking and Communications (ICNC), 2015 International Conference on, 2015, pp. 494–498, doi:10.1109/ICCNC.2015.7069393.
[32] M. Mirza, K. Springborn, S. Banerjee, P. Barford, M. Blodgett, X. Zhu, On the Accuracy of TCP throughput prediction for opportunistic wireless networks, in: 2009 6th Annual IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks, 2009, pp. 1–9, doi:10.1109/SAHCN.2009.5168952.
[33] A. Eswaradass, X.H. Sun, M. Wu, A neural network based predictive mechanism for available bandwidth, 19th IEEE International Parallel and Distributed Processing Symposium, 2005, doi:10.1109/IPDPS.2005.51. 33a–33a
[34] J. Song, J. Li, C. Li, A cross-layer WiMAX scheduling algorithm based on genetic algorithm, in: Communication Networks and Services Research Conference, 2009. CNSR'09. Seventh Annual, IEEE, 2009, pp. 292–296.
[35] R. Zhang, P.-L. Hong, J.-S. Li, L.-P. Guo, A modified mechanism of TCP congestion control over wireless network, J. Circ. Syst. 11 (6) (2006) 001.

[36] P.M. Ruiz, A.F. Gomez-Skarmeta, Using genetic algorithms to optimize the behaviour of adaptive multimedia applications in wireless and mobile scenarios, in: Wireless Communications and Networking, 2003. WCNC 2003. 2003 IEEE, vol. 3, IEEE, 2003, pp. 2064–2068.
[37] F.D. Sanchez Vizcaino, C. Hernandez Benet, Study of TCP Available Bandwidth Using NS2 and Its Forecasting Based on Genetic Algorithm, Karlstad University (Sweden) and Universitat Politècnica de Catalunya (Spain), 2014 Master's thesis.
[38] L. Junior, A. Rodrigues, A study for multi-objective fitness function for time series forecasting with intelligent techniques, in: Proceedings of the 10th Annual Conference Companion on Genetic and Evolutionary Computation, ACM, 2008, pp. 1843–1846.
[39] A. Alvarez, A. Orfila, J. Tintore, Darwin: An evolutionary program for nonlinear modeling of chaotic time series, Comput. Phys. Commun. 136 (3) (2001) 334–349.
[40] G.G. Szpiro, Forecasting chaotic time series with genetic algorithms, Phys. Rev. E 55 (3) (1997) 2557–2568.
[41] S. Fu-Ke, Z. Wei, C. Pan, An engineering approach to prediction of network traffic based on time-series model, in: Artificial Intelligence, International Joint Conference on (JCAI'09), IEEE, 2009, pp. 432–435.
[42] H. Yin, C. Lin, B. Sebastien, B. Li, G. Min, Network traffic prediction based on a new time series model, Int. J. Commun. Syst. 18 (8) (2005) 711–729.
[43] J. Lv, X. Li, T. Li, Network traffic prediction and applications based on time series model, in: Advanced Intelligent Computing Theories and Applications. With Aspects of Artificial Intelligence, Springer, 2007, pp. 1306–1315.
[44] F. Takens, Detecting Strange Attractors in Turbulence, vol. 898, Springer, 1981.
[45] S.R. Garcia, M.P. Romo, J. Figueroa-Nazuno, Characterization of ground motions using recurrence plots, Geofisica Internacional 52 (3) (2013) 209–227.
[46] H. Kim, R. Eykholt, J. Salas, Nonlinear dynamics, delay times, and embedding windows, Physica D 127 (1) (1999) 48–60.
[47] L. Cao, Practical method for determining the minimum embedding dimension of a scalar time series, Physica D 110 (1) (1997) 43–50.
[48] M. Casdagli, Nonlinear prediction of chaotic time series, Physica D 35 (3) (1989) 335–356.
[49] H. Cheng, P.-N. Tan, J. Gao, J. Scripps, Multistep-ahead time series prediction, in: Advances in Knowledge Discovery and Data Mining, Springer, 2006, pp. 765–774.
[50] W. Ming, Y. Bao, Z. Hu, T. Xiong, Multistep-ahead air passengers traffic prediction with hybrid arima-svms models, The Scientific World Journal 2014 (2014) 1–14.
[51] C. Leadbetter, R. Blackford, T. Piper, Cambridge International as and a Level Computing Coursebook, Cambridge University Press, 2013.
[52] M.R. Noraini, J. Geraghty, Genetic algorithm performance with different selection strategies in solving TSP, in: 2011 International Conference of Computational Intelligence and Intelligent Systems, 6–8 July, 2011.
[53] O. Al Jadaan, C. Rao, L. Rajamani, Improved selection operator GA(2008) 269–277.
[54] T. Blickle, Theory of Evolutionary Algorithms and Application to System Synthesis, Computer Engineering and Communication Networks Lab Report, vdf Hochschul Verlag AG, 1997.
[55] S.F. Galan, O.J. Mengshoel, R. Pinter, A novel mating approach for genetic algorithms, Evol. Comput. 21 (2) (2013) 197–229.
[56] S. Patil, Designing optimal network topologies under multiple efficiency and robustness constraints, International Institute of Information Technology Bangalore, 2013 Ph.D. thesis.
[57] M. Srinivas, L.M. Patnaik, Adaptive probabilities of crossover and mutation in genetic algorithms, Syst. Man Cybern. IEEE Trans. 24 (4) (1994) 656–667.
[58] K.-Y. Chen, C.-H. Wang, Support vector regression with genetic algorithms in forecasting tourism demand, Tourism Manage. 28 (1) (2007) 215–226.
[59] A.M. Foley, P.G. Leahy, A. Marvuglia, E.J. McKeogh, Current methods and advances in forecasting of wind power generation, Renew. Energy 37 (1) (2012) 1–8.
[60] D. Akdemir, J.I. Sanchez, J.-L. Jannink, Optimization of genomic selection training populations with a genetic algorithm, Gene. Selection Evol. 47 (1) (2015) 38.
[61] L.D. Chambers, Practical Handbook of Genetic Algorithms: Complex Coding Systems, vol. 3, CRC press, 1998.

Cristian Hernandez is a Ph.D student in Computer Networking at Karlstad University, Sweden. He completed the M.Sc. in Applied Telecommunications and Engineering Management at UPC, in 2014, Barcelona and the B.Sc. in Telecommunications Engineering at UPC. His current research is focused on software defined networking and data centers networking.

Andreas J. Kassler is Professor of Computer Science at Karlstad University, Karlstad, Sweden, that he joined in 2005. From 2003 to 2004, he was Assistant Professor at the School of Computer Engineering, Nanyang Technological University, Singapore. His research interests are in the area of Wireless Meshed Networks, Ad-Hoc Networks, Quality of Service, and Software Defined Networking. He has published over 100 conference and journal papers, and several book chapters. He received the Docent title in Computer Science from Karlstad University in 2006, the Ph.D. degree in Computer Science from Universität Ulm, Germany, in 2002 and an M.S. degree from Universität Augsburg, Germany. Prof. Kassler is a Member of the IEEE.

Enrica Zola received the double M.Sc. degree in Telecommunications Engineering from both Politecnico di Torino (Italy) and Universitat Politècnica de Catalunya (UPC, Spain), in 2002 and 2003, respectively. In 2011, she earned a Ph.D. from the UPC. From September 2001 to August 2002, she collaborated with the Radio Department of the Spanish teleoperator Amena. From March 2003 to February 2006, she has been working at UPC as a full-time Lecturer. From March 2006, she serves as an Assistant Professor at the Department of Telematics Engineering at UPC. She has been teaching design and planning of communication networks and wireless networks. Dr. Zola has been involved in a number of research projects supported by the Spanish Government and the European Commission on performance modeling of wireless systems and networks (IST Emily, RUBI, IST Liaison, COST Winemo, COST290). Her research interest areas encompass wireless networking in general, with special attention to mobility management and radio resource management. Recently, her interest has focused on performance optimization modeling and robust optimization techniques, and on the design of the 5G network.