Mobile Network Resource Optimization under Imperfect Prediction

Nicola Bui[12], Joerg Widmer[1]
[1]IMDEA Networks Institute, Leganes (Madrid), Spain
[2]UC3M, Leganes (Madrid), Spain

***Abstract*—A highly interesting trend in mobile network optimization is to exploit knowledge of future network capacity to allow mobile terminals to prefetch data when signal quality is high and to refrain from communication when signal quality is low. While this approach offers remarkable benefits, it relies on the availability of a reliable forecast of system conditions. This paper focuses on the reliability of simple prediction techniques and their impact on resource allocation algorithms. In addition, we propose ICARO, a resource allocation technique that is robust to prediction uncertainties. The algorithm combines autoregressive filtering and statistical models for short, medium, and long term forecasting. We validate our approach by means of an extensive simulation campaign based on real measurement data collected in Berlin. We show that our solution performs close to an omniscient optimizer and outperforms a limited horizon omniscient optimizer by $10-15\%$. Our solution provides up to 30% saving of system resources compared to a simple solution that always maintains a full buffer and is close to optimal in terms of buffer under-run time.**

I. Introduction

"Every form of behavior is compatible with determinism. One dynamic system might fall into a basin of attraction and wind up at a fixed point, whereas another exhibits chaotic behavior indefinitely, but both are completely deterministic" [1]. In his provocative essay, Ted Chiang is suggesting that unpredictability is just a consequence of the limitedness of human comprehension. While we do not assume that mobile networks are deterministic, in this paper we take resource optimization in mobile networks one step further by exploiting the predictability of future network capacity.

Mobile networks are increasingly constrained by limited spectral resources, while at the same time user traffic demands are growing steadily [2]. Researchers are addressing this challenge from a variety of perspectives including massive multiple-input multiple-output communications, heterogeneous networks combining femto, micro and macro cells, device-to-device communication, and the concept of exploiting knowledge about user behavior and the network itself for performance optimization.

Recent studies [3] highlight how network dynamics [4] can be understood, predicted and linked to human mobility patterns [5]. This ability to predict user and network behavior allows optimizing resource allocation [6], [7]. As such, it is a highly interesting approach to increase the efficiency of mobile networks and deal with future traffic growth.

In this paper, we propose a resource allocation algorithm for mobile networks that leverages link quality prediction and prediction reliability. Our solution exploits simple autoregressive (AR) filters to compute short term prediction [8], [9] and the analytical data rate model we developed in [10]. Thus, we do not assume a perfect knowledge of future evolution of the network as in [6], [11], and we are able to extend the prediction horizon from tens of seconds [12] to the order of minutes.

We develop an optimal resource allocation algorithm that assumes perfect forecast and a general prediction framework that combines short and medium/long term prediction. Subsequently, we introduce our Imperfect Capacity prediction-Aware Resource Optimization (ICARO) algorithm, which iteratively uses the optimal algorithm on a predicted data rate sequence. Finally, we validate our approach on data traces derived from measurements performed in Berlin by the MOMENTUM project [13] and we show that ICARO achieves almost optimal outage performance and outperforms solution with shorter prediction horizon.

The rest of the paper is structured as follows: Section II provides a summary of the related work. Section III describes the system model and assumptions. In Section IV we present the omniscient resource allocation algorithm. Section V analyzes future prediction feasibility and its limits: Section V-A gives details about the filtering technique used to obtain short term predictions and Section V-B discusses the statistical tools for medium to long term forecasting. Section VI provides our solution for resource allocation under imperfect prediction and the performance of this algorithm is analyzed in Section VII. Section VIII provides our final considerations.

II. Related work

Several recent papers, for example [6], [7], [10]–[12], optimize mobile network resources by exploiting future knowledge in order to save both energy and cost. The main idea is that it is better to communicate when the signal quality is good and refrain from doing so when the signal quality is bad: better signal quality results in higher spectral efficiency and fewer resources are needed to send the same amount of data.

For instance, the authors of [6] provide an optimal resource allocation algorithm exploiting perfect future knowledge, while the authors of [7], [11] provide a linear programming (LP) formulation of the resource allocation problem and solve it with optimal LP solvers. In [12], actual mobility prediction tools have been used to validate proportionally fair scheduling algorithms for cellular networks.

978-1-4799-8461-9/15/$31.00

This paper considers a general formulation of the resource allocation problem which is not limited to video delivery. As in other works, it assumes that it is always possible to know in advance what content the user will be interested in [14] in order to be able to be able to prefetch data. Also, we relax the assumption of perfect knowledge of future system conditions, taking into consideration prediction techniques and their reliability.

Network capacity prediction has been studied, for example, in [8], [9] and [10]. These papers evaluate ARMA and GARCH filtering techniques that account for sequences of random variables that have the same (homoscedastic) or different (heteroscedastic) finite variance respectively. Other papers [15], [16] investigate mobility prediction using either Markovian estimators or trajectory-based forecasting techniques. Margolies at al. [12] propose an advanced map-based solution to extend the network forecast to tens of seconds.

A key aspect of our solution is that it accounts for the statistic model we developed in [10], which extends those proposed in [4], [17] to account for imprecise information. This approach allows us to extend the prediction horizon to the order of minutes, without requiring very complex computations.

III. System model

In this paper we address the downlink from a base station of a mobile network (eNodeB) to a single receiver (UE). To simplify the description of the problem, we consider slotted time with slot duration t and thus the quantities discussed in the paper are discrete time series. We use i, j, and k to refer to slot indices. The quantities of interest are:

• Position $P = \{p_i \in [0, P_{\max}],\ i \in \mathbb{N}\}$, where p_i is the distance between UE and eNodeB and $P_{\max}$ is the coverage range.

• Active users $N = \{n_i, i \in \mathbb{N}\}$, where n_i is the number of active users that are in the same cell as the UE. It reflects the congestion level of the cell in slot i.

• Signal to interference plus noise ratio (SINR) $S = \{s_i \in \mathbb{R},\ i \in \mathbb{N}\}$, where s_i is obtained from p_i as follows:

$$s_i = s_0 p_i^{-\alpha} f_F. \tag{1}$$

Here, s_0 is a system constant, α is the path loss exponent and f_F is a random multiplicative term to account for fast fading.

• User cell capacity $C = \{c_i \in [0, C_{\max}],\ i \in \mathbb{N}\}$, where c_i represents the average capacity obtained by the user during slot i. $C_{\max}$ is the maximum capacity allocable to the UE, given the specific mobile technology. We compute c_i as a function of s_i and n_i through

$$c_i = c_0 g_c(s_i, n_i), \tag{2}$$

where c_0 is a system constant and g_c is a technology dependent function which models system level variables such scheduling policy, congestion, spectral efficiency, etc. In the rest of the paper we consider LTE as the mobile network technology and we adopt the model in [17], which provides a closed form expression for f_F and g_c for a user at a given distance from the base station, when another $n-1$ users are uniformly distributed in the cell area and proportionally fair scheduling is used.

• Receive rate $R = \{r_i \in [0, c_i],\ i \in \mathbb{N}\}$: this is the rate at which the base station sends data to the UE in slot i.

• Download requirement $D = \{d_i \in [0, D_{\max}], i \in \mathbb{N}\}$, where $D_{\max}$ is the maximum data consumption rate. In slot i, the user consumes d_i bytes of data if they are available. If at any time the user receives more data than required, the excess can be stored in a buffer for later use.

• Buffer state $B = \{b_i \in [0, B_M],\ i \in \mathbb{N}\}$, where b_i is the buffer level and B_M is the buffer size in bytes.

• Buffer under-run time $U = \{u_i \in [0, 1],\ i \in \mathbb{N}\}$ is the fraction of slot i for which no data was available to satisfy the download requirements.

The aforementioned quantities are linked as follows:

$$b_{i+1} = \min\{\max\{b_i + r_i - d_i, 0\}, B_M\} \tag{3}$$

$$u_i = \begin{cases} \max\{d_i - r_i - b_i, 0\}/d_i & d_i > 0 \\ 0 & d_i = 0 \end{cases}. \tag{4}$$

The buffer fills (up to the full buffer B_M) whenever the download rate is higher than the consumption rate, $r_i > d_i$. In case $r_i < d_i$, the algorithm empties the buffer and accumulates buffer under-run time whenever $b_i + r_i < d_i$.

In what follows, we refer to function $y = g_y(x)$ as g_y. Similarly, we refer to the probability density function and the cumulative density function (CDF) of a random variable X as $f_X(x)$ and $F_X(x) = \int_{-\infty}^{x} f_X(y)\mathrm{d}y$ and with μ_X and σ_X to its mean and standard deviation.

IV. Resource allocation optimization with perfect forecast

The resource allocation problem aims at finding the optimal rate time series R that satisfies the download requirements D by using the available capacity C in the most efficient way. We define the following objective function:

$$O = \{o_i = r_i/c_i \in [0, 1],\ i \in \mathbb{N}\}, \tag{5}$$

where o_i is the fraction of the available capacity used in slot i and represents a cost. Note that the same rate r has a different cost $o_i > o_j$ if the available capacity $c_i < c_j$. We obtain the following optimization problem:

$$\begin{aligned} \underset{R}{\text{minimize}} \quad & \sum_i o_i \\ \text{subject to:} \quad & \sum_i u_i = \sum_i u_i^*, \\ & b_i \le B_M,\ \forall i \in \mathbb{N}, \end{aligned} \tag{6}$$

where $\sum_i u_i^*$ is the minimum feasible buffer under-run time. To minimize this cost function, the base station should send more data when the available capacity is high and use just the minimum rate required to avoid a buffer under-run when the capacity is low.

The solution of Eq. 6 is the optimal resource allocation strategy R^* that achieves the minimum buffer under-run time $\sum_i u_i^*$ at the lowest cost $\sum_i o_i^*$. If the sequence C is known a priori, various offline algorithms can be used to determine the optimal resource allocation. We propose a simple algorithm

that we call Split & Sort (S&S), which splits the optimization horizon into windows so that allocation decisions belonging to two different windows can be made independently. Within each window slots are used in descending order of predicted capacity. The last slot of each window is called a break-point.

S&S computes the optimal solution of Eq. 6 by using the following rules: *i*) define the break-point e_l as the last slot for which all previous rates are finalized (i.e., no more rate can be used in slots up to e_l) which requires that either $b_{e_l} = B_M$ or $r_k = c_k, \forall e_{l-1} < k \leq e_l$; *ii*) define the optimization window $[e_l + 1, m]$, where e_l is the last break-point slot and the rate allocated in all slots in $e_{l-1} < k \leq e_l$ is finalized; *iii*) starting from $l = 0$, $e_l = 0$ and $m = 1$ the algorithm accounts for the slots in the set $\{e_l+1, \ldots, e_l+m\}$ to satisfy the requirements up to slot $e_l + m$; the algorithm chooses a slot if it has the highest capacity among the unused ones in the set. *iv*) the algorithm either increments l, updates e_l and resets $m = 0$ if a break-point is found or increments m. The complete Split & Sort algorithm is given in Algorithm 1. $\mathrm{sAdd}(X, x)$ adds the element x to the sorted list X in the correct position, $\pi(c_i)$ gives the position in C of the element c_i and $\mathrm{shift}(D, u_j, j)$ is a shift function that recomputes the requirements sequence D accounting for a buffer under-run event u_j in slot j. The following conditions are used:

- $\mathrm{I}_1 := \exists\ e_l < j \leq e_l + m \mid b_j = B_M$ to verify whether a full buffer state is reached,
- $\mathrm{I}_2 := \sum_{j=e_l+1}^{e_l+m} c_j - r_j = 0$ to verify whether all of the available capacity is used, and
- $\mathrm{I}_3 := \sum_{j=e_l+1}^{e_l+m} r_j - d_j = 0$ to verify whether all of the download requirements have been satisfied.

In the following we prove the optimality of Algorithm 1 and discuss the behavior of the algorithm when knowledge of the future capacity is not perfect.

Theorem 1 (Split & Sort Optimality): If R is a solution of Algorithm 1 with C and D as inputs and it achieves a buffer under-run time $\sum_i u_i$ and cost $\sum_i o_i$, then there exists no other allocation strategy $R' \neq R$ for C and D that obtains performance $\sum_i u'_i$ and $\sum_i o'_i$, for which $(\sum_i u'_i < \sum_i u_i) \vee (\sum_i u' = \sum_i u_i \wedge \sum_i < \sum_i o_i)$, i.e., it has either a lower buffer under-run time or the same buffer under-run time and a lower cost.

In the following we will prove the theorem by contradiction: we show that is impossible that a solution exists which is both different from that provided by S&S and achieves better performance, due to either the stopping conditions of the algorithm or the ordering of the decisions within an optimization window.

Proof: Theorem 1 can be proven by contradiction on the following hypotheses:
Assume a solution $R' \neq R$ exists so that

1) either $\sum_i u'_i < \sum_i u_i$ (shorter buffer under-run time)
2) or $\sum_i u'_i = \sum_i u_i \Rightarrow \sum_i o'_i < \sum_i o_i$ (cheaper)

For 1) R cannot satisfy the requirements D in all the slots, thus $\sum_i u'_i < \sum_i u_i \Rightarrow \exists\ j$ s.t. $r_j + b_j < r'_j + b'_j < d_j$. Since $R' \neq R$, they must differ before or on slot j in order to cause the larger under-run time, because any variation later than that cannot decrease $\sum_i u'_i$. Since R is obtained using Algorithm 1 and must result in $u_j > 0$, then for all the slots belonging to the analysis window $[e_{l-1} + 1, e_l]$, where $e_l = j$ the whole available capacity must have been used, which means R' cannot use more capacity there to avoid the buffer under-run. R' cannot use more capacity before slot e_{l-1} either, since that would impact a window already completed (ended because of condition I_1). Thus, $\sum_i u'_i \geq \sum_i u_i$ if the two strategies are different, which contradicts the first hypothesis.

For 2) it is $(R \neq R') \wedge (\sum_i u_i = \sum_i u'_i) \wedge (\sum_i o_i < \sum_i o'_i)$, thus the two strategies must differ in at least two slots j, k, where $c_j > c_k$ and $(r_j < r'_j) \wedge (r_k > r'_k)$. The two slots j, k cannot belong to the same window, because Algorithm 1 uses the slots from a sorted list and finishes either with a full buffer or when the whole capacity has been used. The two slots j, k cannot belong to different windows either, because if $j < k$, it would have been possible to use more capacity earlier in the allocation which is not possible due to the stopping conditions of the algorithms, whereas if $j > k$, a cheaper slot later in the sequence could have been used instead of a more expensive one earlier in the sequence. However, this is not possible due to

Algorithm 1 Split & Sort Algorithm (S&S)

Input: the knowledge of the future capacity availability C, the future download requirements D and the initial buffer level B_0.
Output: $R = \mathrm{SS}(C, D, B_0)$

```
l = 0, e_l = 0 // set the starting point
b_{e_l} = B_0 // set the starting buffer
r_{e_l} = 0, R = ∅ // set the starting allocation
while |R| < |D| do
  m = 1 // set the initial window size
  S = ∅ // sorted capacity vector initialization
  while ¬I_1 ∧ ¬I_2 ∧ |R| + m < |D| do
    S = sAdd(S, c_{e_l+m}) // add an element to the sorted capacity list
    i = 1
    while i ≤ m ∧ ¬I_3 do
      r_{π(s_i),old} = r_{π(s_i)} // store previous allocation
      r_{π(s_i)} = min{r_{π(s_i)} + d_{e_l+m}, c_{h_i}, B_M − b_{π(s_i)}}
      b_{π(s_i)+j} = b_{π(s_i)+j} + r_{π(s_i)} − r_{π(s_i),old}, ∀ 1 ≤ j ≤ m − π(s_i)
      i = i + 1
    end while
    m = m + 1 // update the window size
  end while
  if I_1 then
    l = l + 1, e_l = j // new break-point
  else
    l = l + 1, e_l = e_{l−1} + m // new break-point
    if I_2 then
      u_j = max{d_j − r_j − b_j, 0}/d_j
      D = shift(D, u_j, j) // shift of requirements
    end if
  end if
  R = {R, r_{e_{l−1}}, ..., r_{e_l}} // update the allocation
end while
return R
```

either the fact that the more expensive slot must have been used in order not increase $\sum_i u_i$ (stopping condition I_2) or because of the ordered selection of the slots (stopping conditions I_1 or I_3). Thus, $\sum_i o'_i \geq \sum_i o_i$ if the two strategies are different, which contradicts the second hypothesis.

Thus, assuming that an allocation strategy R' provides a better solution than that obtained using Algorithm 1 violates the hypotheses of the theorem, which is therefore proved. ■

Algorithm 1 will be later used in Section VI in an iterative procedure to compute the resource allocation when the knowledge of future capacity is inaccurate.

V. General forecast model

In this section we propose a general model describing the forecasting reliability of a system. In particular, we split our model in three time periods based on the prediction horizon:

The **short term** period considers the near future and predicts capacity through time-series filtering techniques [8], [9]. It is characterized by the reliability time τ_p, which defines how many slots of the sequence can be predicted. We discuss this in Section V-A.

The **medium term** period describes the evolution of the system in terms of available capacity statistics. During this period one or more network cells can be accounted to in the mobility predictor: Markovian predictors [15] can usually compute the likelihood of visiting a given cell, while trajectory-based predictors [16] provide a more accurate estimate by computing the actual distribution of the user position over time.

The **long term** period provides an overall statistical evaluation of the available capacity availability based on the steady state distribution of the user position in the network. Both the medium and the long term periods are discussed in Section V-B.

A. Short term forecast with filters

This section addresses the reliability time τ_p achievable by filtering techniques applied to available capacity time series. In particular, we study autoregressive-moving average (ARMA) filters and their setup according to the system dynamics defined by the slot time t and the user speed v. We opted for ARMA instead of GARCH [8], since capacity elements belonging to the short term period are characterized by the same finite variance.

For each $(t \in [0.5, 5], v \in [0.5, 5])$ tuple we consider a set of 100 capacity traces computed using Eqns. (1) and (2) as per [17], starting from the mobility paths of a user moving at constant speed in a random network deployment. We apply the Box-Jenkins [18] method to determine the type and the order of the filter to be used with each sequence. Through the analysis of autocorrelation and partial autocorrelation plots, we find that the best technique for our sequences consists of simple autoregressive (AR) filters of order τ_F, and that τ_F is inversely proportional to the tv product.

Subsequently, for each of the sequences we estimate filter coefficients by means of the linear least squares procedure [19] and we use the obtained filter to forecast the values of the other sequences with the same (t, v) parameters. We refer

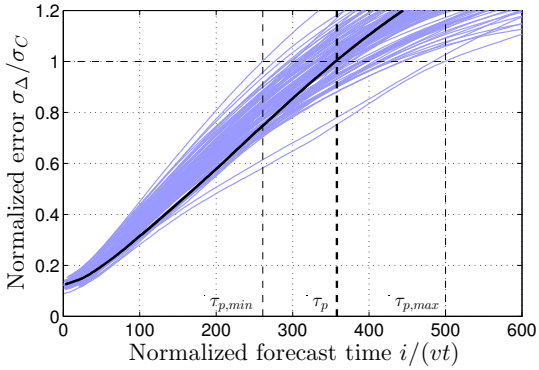

Fig. 1. The shaded area represents how the standard deviation of the short term prediction error increases with increasing prediction distance varying the user speed v and the slot time t. τ_p represents the time after which $\sigma_\Delta \geq \sigma_C$.

to a forecast sequence as $\tilde{C} = \{\tilde{c}_i \in [0, C_{\max}],\ i \in \mathbb{N}\}$, obtained from C and to the corresponding error $\Delta = \{\delta_i = \tilde{c}_i - c_i \in [-C_{\max}, C_{\max}],\ i \in \mathbb{N}\}$. We consider a prediction to be reliable as long as the standard deviation of the error is lower than that of the capacity, $\sigma_\Delta = \sigma_C$.

Thus, we compute μ_Δ and σ_Δ as the average and the standard deviation of all the error sequences with the same (t, v) parameters. Fig. 1 shows on the abscissa the prediction time index normalized on t and on the ordinate σ_Δ/σ_C the standard deviation of the prediction normalized on the standard deviation σ_C of the original series C.

While the actual steepness of the curves varies with the parameters, for all of them the normalized error standard deviation σ_Δ/σ_C approaches 1 almost linearly. Hence, we set $\tau_p = \mathrm{argmin}_i$ s.t. $\sigma_{\Delta_i}/\sigma_C > 1$. In addition, we observe that both τ_p and the filter order can be approximated with simple linear models with the inverse of the tv product and that τ_p is usually 10 times as large as the order of the AR filter.

Finally, it is sufficient to tune a set filters for varying t and v and select the one to use according to the actual user mobility. Also, since filters can be normalized on σ_C it is not needed to have different filters for different numbers of active users in the cell, but it is sufficient to rescale the constant and the variance parameters of the filter.

B. Statistical models and uncertainties

For medium and long term prediction we base the model of distribution of per user capacity we started on [17], since to the best of our knowledge it is the only one which takes into account the scheduler impact and thus is able to model user contention.

To account for the impact of uncertainties on the user position and/or the number of active users in the cell we modify the expression of the capacity distribution $f_C(x)$ obtained for a specific position p_i and number of users n_i to the actual distribution of the user position $f_P(x)$ and the probability mass function $f_N(n)$ of the number of active users in the cell, as follows:

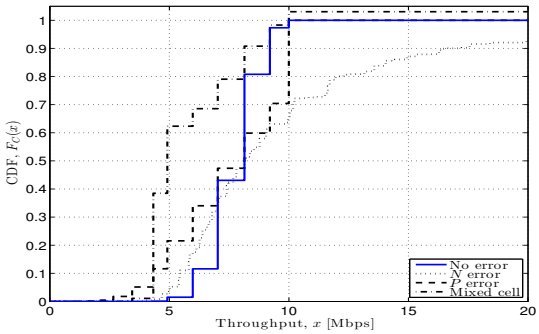

Fig. 2. Examples of the impact of uncertainties on the capacity distribution.

$$f_C(x) = \sum_{i \in \mathcal{N}} f_N(i) \int_0^{\infty} f_{F,P|i}(g_C^{-1}(x,p),p|i) \left| \frac{\partial g_C^{-1}(x,p)}{\partial x} \right| \mathrm{d}p, \quad (7)$$

where $f_{F,P}$ is the joint distribution of fading and position, g_C is the function linking the per user capacity to p and n and $\mathcal{N}$ is the support of $f_N(n)$. Since fading and user position are statistically independent, their joint distribution $f_{F,P}(x,y) = f_F(x)f_P(y)$ is the product of their distributions. Eq. (7) modifies the original capacity distribution weighting it through the active user probability mass function $f_N(i)$ and the user position probability $f_P(y)$; the partial derivative normalizes the integrand.

For what concerns our analysis, it is sufficient to be able to compute the per user capacity distribution by accounting for limited knowledge of the user position and traffic in the cell by means of their respective distributions.

So far, our model describes capacity only for the case when the cell the user is connected to is known perfectly. To account for different cells, it is sufficient to consider the weighted sum of the capacity distributions of single cells,

$$f_C(x) = \sum_{i \in \mathcal{C}} \rho_i f_{C,i}(x), \quad (8)$$

where $\mathcal{C}$ is the set of cells that can be visited in the next time period with some probability, $f_{C,i}(x)$ is the capacity distribution related to cell i and ρ_i is the probability of visiting cell i in the next time period.

Fig. 2 provides a few examples of the CDF obtained using the model. The solid line is representative of the capacity CDF $F_C(x)$ when both the active user number $n = 5$ and the user position $p = 500$ meters are exactly known so that the distribution is equal to the fading distribution. The dotted line accounts for an error in the number of active users in the cell so that $f_N(x) = \{0.2, 0.6, 0.2\}$ for $x = \{4, 5, 6\}$ respectively. Conversely the dashed line is obtained by accounting for an error in the user position which has a normal distribution with parameters $\mu_P = 500$ meters and $\sigma_P = 100$ meters. Finally, the dash-dotted line is obtained by mixing together two cells with 5 and 10 users with 20% and 80% of visiting probability respectively. The piecewise-constant shape of the curves is due to the discrete relationship between SINR and bitrates.

While practical implementations of this solution can use different methodologies, in our evaluation campaign we proceed as follows. From the measurements of the MOMENTUM project and the channel model in [17] we derive the model for the capacity distribution for each cell of the network (cells are defined so that each point of the area is associated to the base station which has the strongest average SNR). We assume the user position statistic $f_P(x)$ to be uniform in the area (i.e., we do not leverage any auxiliary information such as street topology). Similarly, we computed the cell traversal time $\tau^{(i)}$ as the ratio between the average cell width and the average user speed. Thus, we are able to define the statistical model for each cell of the network, while, for the long term period we use Eq. (8) over the whole area and assume a uniform distribution among cells.

VI. Resource allocation optimization under uncertainties

The objective of this section is leveraging the concepts of the previous ones to design a network resource allocation algorithm which takes into account imperfect forecast, called Imperfect Capacity prediction-Aware Resource Optimization (ICARO). ICARO aims at minimizing the communication cost while avoiding buffer under-runs. In particular, we use Algorithm 1 (S&S) of Section IV in an iterative way. At each iteration, Algorithm 1 makes a single decisions about which rate r to use by exploiting both the AR predictor described in Section V-A and the statistical models designed in Section V-B.

Before describing the new algorithm, we describe how to obtain a single general capacity prediction to use with Algorithm 1. In order to account for the three time periods described in Section V we proceed as follows:

1) The short term prediction $\tilde{c}_i^{(F)}$ with $i \in [0, \tau_p]$ is obtained from the known past capacity information [20] and choosing the filter order τ_F and coefficients based on the user speed v.

2) The medium term model $f_{C,i}(x)$ is computed as the superposition of the cells $j \in \mathcal{C}$ that the user is likely to visit in the i-th time period, each of them accounted for according to their user position $f_{P,j}(y)$ and active user number $f_{N,j}(z)$ statistics by Eq. (8). Similarly, the duration of the i-th time period $\tau_i - \tau_{i-1}$, is obtained as a weighted sum of cells traversal time $\tau^{(j)}$ related to cell $j \in (C)$.

3) During the i-th time period $D_i = \sum_{j=\tau_{i-1}}^{\tau_i} d_j$ bytes have to be downloaded to avoid a buffer under-run. The maximum cell efficiency is achieved when only the slots with the highest capacity are used.

4) The highest threshold $c_{T,i}$ is computed so that the average amount of data obtained by selecting only the slots with larger capacity than c_T is larger than $D_i/(\tau_i - \tau_{i-1})$:

$$c_{T,i} = \max_y \ \text{s.t.} \int_y^{\infty} x f_{C,i}(x) \mathrm{d}x \geq D_i/(\tau_i - \tau_{i-1}). \quad (9)$$

5) The i-th time period is modeled as a sequence of $\tau_i - \tau_{i-1}$ values

$$\tilde{c}_j^{(M,i)} = \begin{cases} c_{T,i} & j > (1 - F_{C,i}(c_{T,i}))(\tau_i - \tau_{i-1}) \\ 0 & \text{otherwise} \end{cases}, \quad (10)$$

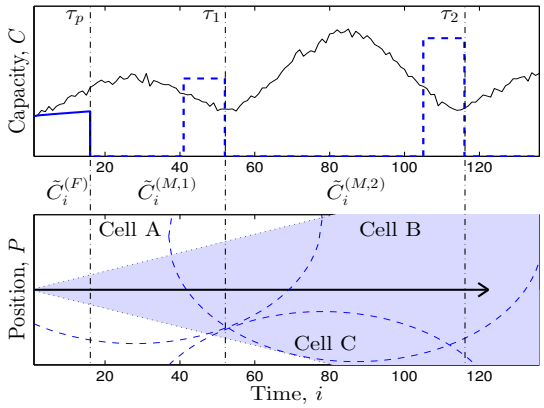

Fig. 3. Example of the general prediction model and related user position.

where $F_{C,i}(c_{T,i})$ is the probability of the capacity being lower than $c_{T,i}$, thus $(1-F_{C,i}(c_{T,i}))(\tau_i-\tau_{i-1})$ is the average number of slots with larger capacity than the threshold.

6) Steps 2 to 5 are repeated and new time periods are added in the sequence if their reliability is sufficient. In our evaluation campaign we consider two periods only, each related exactly to one cell: the current and the following which we choose according to the current mobility direction.

7) Compute τ_o as the offset time when the user first entered in the cell.

8) Obtain the predicted capacity sequence as the concatenation of the previously computed time period sequences:

$$\tilde{c}_i = \begin{cases} c_0 & i = 0 \\ \tilde{c}_i^{(F)} & 0 < i \leq \tau_p \\ \tilde{c}_i^{(M,1)} & \tau_p < i \leq \tau_1 \wedge \tau_1 > \tau_p + \tau_o \\ \tilde{c}_i^{(M,2)} & \max(\tau_o + \tau_p, \tau_1) < i \leq \tau_2 \\ \cdots & \\ \tilde{c}_i^{(M,n)} & \tau_{n-1} < i \leq \tau_n \end{cases}, \qquad (11)$$

where τ_n is the duration of the whole sequence, c_0 is the known present capacity, and $\tilde{c}_i^{(M,1)}$ is the current period capacity distribution. $\tilde{c}_i^{(M,1)}$ is modified by accounting for the time passed from when the user first entered the cell in τ_o: for each passed time slot one sample is removed either from the beginning if $c_0 < cT, 1$ (higher capacity can be found later, since the current capacity is lower than the current capacity threshold) or from the end otherwise (capacity is sufficiently high).

Fig. 3 shows an example of a mixed model sequence: the upper part compares the ground truth (C as a thin solid line) to the short term ($\tilde{C}^{(F)}$ as a thick solid line) and the medium-long term ($\tilde{C}^{(M,1)}$ and $\tilde{C}^{(M,2)}$ as dashed line) predictions respectively. The lower part is a map of the user movement (central horizontal arrow) and the coverage areas of different cells (dashed circles). The shaded area highlights the uncertainties in future user position. Dash-dotted lines crossing the figures mark τ_p, τ_1 and τ_2 instants.

In every time slot, ICARO (Algorithm 2) computes the mixed forecast sequence of Eq. (11) and uses Algorithm 1 (S&S) to allocate the rate of the current slot. The algorithm iterates until the requirements are completely satisfied.

Algorithm 2 Imperfect Capacity prediction-Aware Resource Optimization (ICARO)

Input: the future download requirement D, user speed v and position p, τ_F past values of the capacity sampled with t period, the capacity statistics $f_{C,i}(x)$ and time period traversal time τ_i for the next predictable time periods.
Output: R, O, U
  $s = 0$ *// set the starting point*
  $b_s = B_0$ *// set the starting buffer*
  $r_s = 0$, $R = \emptyset$ *// set the starting allocation*
  while $\sum_{i=s}^{|D|} d_i \geq b_s$ **do**
    compute $\tilde{C}$ as per Eq. (11)
    run $\hat{R} = \mathrm{SS}(\tilde{C}, D, b_s)$ *// allocation is computed using Algorithm 1 on the predicted sequence of Eq. 11*
    $r_s = \min(\hat{r}_1, c_s, B_M - b_s)$ *// rate to be used*
    compute next buffer state b_{s+1} according to Eq. (3)
    compute buffer under-run u_s according to Eq. (4)
    compute cost o_s according to Eq. (5)
    $s = s + 1$
    $D = \{d_i, s < i \leq |D|\}$ *// remove the first element from the requirements sequence*
  end while
  return R, O, U

The rationale for using the S&S algorithm on the mixed forecast sequence is that its operational principle, that selects which slot to use in descending order, still works under uncertainties and provides a solution which is conservative (as the highest capacity slots are placed last) to avoid under-runs, and aggressive (as the allocation priority is given to the most reliable slots) to optimize allocation costs. In the following, we provide a few examples of the algorithm:

Ordering the short term forecast: the elements of the short term prediction sequence can be assumed to have the same order of those of the actual sequence. In fact, as we showed in Section V-A, $\sigma_{\delta,i}$ is increasing with i, thus if $\tilde{c}_i^{(F)} > \tilde{c}_j^{(F)}$ and $j > i$, then the probability of having the same ordering is larger than that of opposite order ($\mathrm{P}[c_i > c_j] > \mathrm{P}[c_i \leq c_j]$). Thus, the S&S algorithm can be used on the short term prediction, because its order is likely to match that of the actual sequence.

Comparing short and medium term forecast: the i-th medium term period is represented as a sequence of $(\tau_i - \tau_{i-1})F_{C,i}(c_{T,i})$ zero capacity slots while the remaining slots are equal to $c_{T,i}$, which represent a worst case scenario computed on the known capacity distribution. Hence, if the short term prediction is lower than $c_{T,1}$ only the minimum rate is used, since from the statistical model slots of higher capacity are expected to come later. Conversely, if the short term prediction is larger than $c_{T,1}$, then it is more likely that the remaining slots will be lower than the threshold (see also step 8 of the sequence creation). In other words, running the S&S algorithm on this sequence ensures that it buffers enough data to avoid using the zero-capacity slots by exploiting those with a capacity larger than $c_{T,1}$.

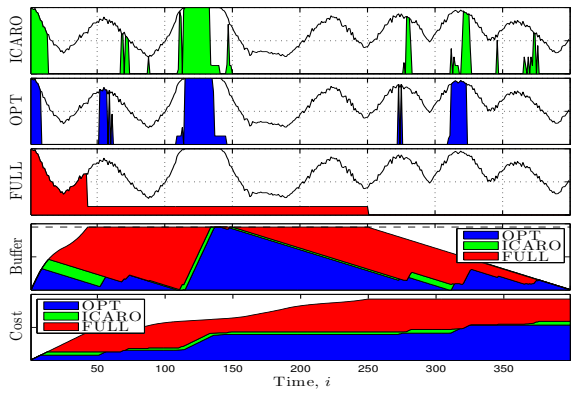

Fig. 4. Comparison among the three main algorithms.

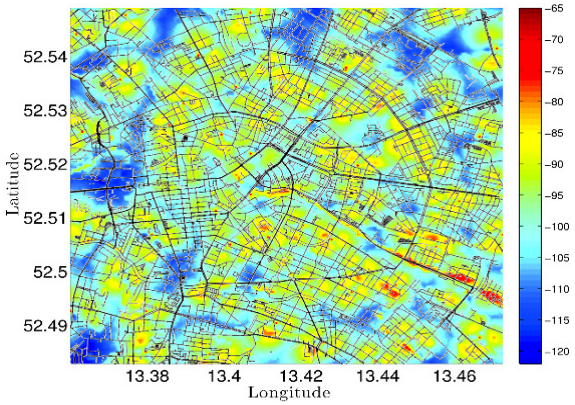

Fig. 5. Pathloss map of Berlin as measured by the MOMENTUM project [13]

Buffering: the algorithm will always try to use the slots above threshold in each time period and bridge the gaps between those by using the buffer. By positioning the slots with highest capacity at the end of each time periods we ensure that the algorithm is conservative. Finally, the maximum buffer size B_M limits the optimization horizon of the algorithm: in fact, the maximum time that the system can last without using any capacity is given by $B_M/(\sum_i d_i/|D|)$. Hence, the buffer size has a significant impact on the algorithm's performance which we analyze in the next section.

Fig. 4 shows an example of ICARO's performance compared to the optimal boundary (OPT) obtained with perfect forecast and to the trivial (FULL) solution which maintains the buffer as full as possible at all times. The top three plots show the used rate R of the three algorithms: ICARO, OPT and FULL from the top. The shaded areas represent the used part of the total available capacity (solid line). While FULL continues to fill the buffer during the low quality period ($i = 25$), OPT just uses the needed quantity to harness the best part of the second cell ($i = 50$). ICARO's decisions, even though slightly more conservative (i.e.: $i = 80$), are very similar to OPT's. The last two plots show the buffer and the cumulative cost variation respectively.

VII. Results

In this section we provide an analysis of the performance of our algorithm. In particular we compare ICARO against the following algorithms:

- OPT: the optimum offline allocation computed with the optimal S&S algorithm on the exact capacity time series.
- FULL: the most conservative approach which just fills up the buffer as soon as possible and maintains it as full as possible until the download requirements are satisfied.
- OPT(x): an optimal algorithm that iteratively makes the decision on the current slot by running the S&S algorithm on the first x samples of the exact future capacity. This algorithm targets a full buffer (or the sum of the remaining requirements if it is lower than the buffer size) at the end of the optimization window. This algorithm represent the performance upper bound for any solution using at most x samples of prediction.

Our main performance metrics are the objective function O and the buffer under-run time U. To compare the results of every tested configuration, we adopt the average cost $\xi = \sum_i o_i/|O|$, the average cost saving $\eta = (\sum_i o_{i,\mathrm{FULL}} - o_{i,\mathrm{ICARO}})/\sum_i o_{i,\mathrm{FULL}}$ obtained by our algorithm, and the average buffer under-run time increase $\zeta = \sum_i u_{i,\mathrm{ICARO}} - \sum_i u_{i,\mathrm{OPT}}$. In addition, we study the impact of the parameter x on OPT(x) and compare it to our solution.

Our evaluation campaign considers an LTE network scenario based on the pathloss data provided by the MOMENTUM project [13] and accounts for both vehicular and pedestrian mobility ($\mu_v = 5$ and 1 meters per second respectively). For each evaluation round we generate a random mobility trace in a 12×6 square kilometer area of Berlin (centered at latitude $52.52°$ North and longitude $13.42°$ East. From the mobility trace, we generate a pathloss trace computed on the pathloss map of Fig. 5. Finally, we account for fast fading as per in the analysis of proportionally fair scheduling in [17] to obtain the capacity trace. In all experiments we consider an average number of active users $N = 5$ uniformly distributed.

Each of the trace represents the ground truth for one of our experiments and consists of 4000 capacity samples. From the whole sequence we estimate the parameters of the AR filters that we use for ICARO (see Section V-A above), while we run the four algorithms on 10% of the samples only, chosen starting from a random starting point of the trace. In order to provide ICARO with the medium and long term parts of the prediction we assume the following:

- a user stays in a cell for an average time equal to the ratio between the average cell width (that we computed numerically for each cell from the MOMENTUM data) and the user speed;
- the capacity distribution of a given cell is computed as per Section V-B assuming the position to be uniformly distributed in the area of the cell;
- the current and the next cells are known;
- the long term distribution is the combination of all the cells visited during the whole trace.

Fig. 6 and Fig. 7 show the main results of our evaluation

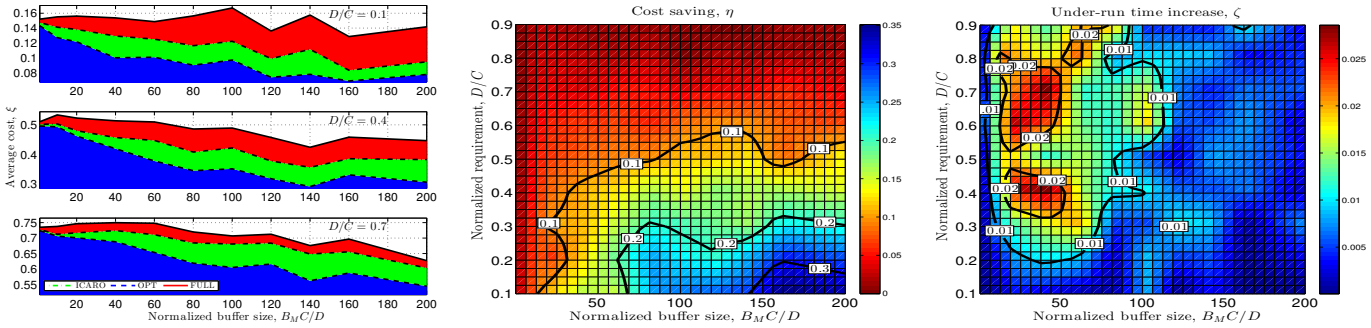

Fig. 6. Performance comparison among ICARO, OPT and FULL with vehicular mobility. ξ, η and ζ are plotted on the left, center and right respectively.

campaign for pedestrian and vehicular mobility respectively: in both cases we vary the requirement over capacity ratio $(\sum_i d_i / \sum_i c_i) \in [0.1, 0.9]$, and the normalized buffer size $(B_M \sum_i c_i / \sum_i d_i) \in [1, 200]$.

Fig. 6 (left) shows the average cost ξ of the three main algorithms (OPT, ICARO and FULL as solid, dashed and dash dotted lines, respectively) varying the buffer size (x-axis) for $\sum_i d_i / \sum_i c_i = \{0.1, 0.4, 0.7\}$ (upper, center and lower plots). The variable horizon algorithm OPT(x) is accounted for in Fig. 8 and Fig. 9 for better readability.

In the upper plot the download requirements are moderate and both OPT and ICARO are able to obtain a normalized cost lower than 0.08 (corresponding to 80% of the $\sum_i d_i / \sum_i c_i$), while FULL often needs more than 100% of the average requirements ($\xi \geq 0.1$). The performance is similar in the other plots and ICARO is always better than FULL and close to OPT. As expected, ICARO performance improves when the buffer is larger and the requirements are lower. Notably, when the buffer is very small the three algorithms perform almost exactly the same as a too small buffer does not allow leveraging forecast information.

The central figure shows contour plots of ICARO's efficiency η using $B_M \sum_i c_i / \sum_i d_i$ as abscissa and $\sum_i d_i / \sum_i c_i$ as ordinate: the curves are labeled according to the cost savings achieved and the area color changes to red where the savings are lower than 10%, while it changes to cyan and blue when it is higher than 15%. Again the best results are obtained for medium-large buffer size and small requirements where ICARO is about $25 - 30\%$ cheaper than FULL. On average, ICARO is 10% worse than OPT.

The figure on the right shows how close ICARO is to the optimal buffer under-run time obtained by both OPT and FULL. We plot ζ using the same coordinates as those of the previous figure. Here the blue part of figure highlights where ICARO is able to achieve almost optimal performance ($\zeta \leq 0.01$), while green and red areas correspond to slightly worse performance ($0.01 < \zeta < 0.04$). Notably, for no parameters the buffer under-run time was larger than 0.05, and the worst performance is obtained for low buffer sizes.

Fig. 7 provides results equivalent to those of the previous set of figures, but obtained for vehicular mobility. Here, all the trends identified above are confirmed and ICARO performs slightly worse than for pedestrian mobility. This is chiefly due to the higher variability of the capacity traces.

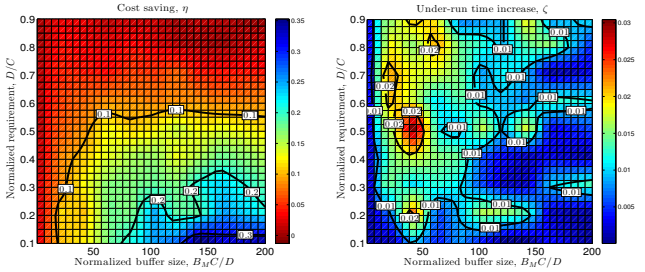

Fig. 7. Performance comparison with pedestrian mobility.

Since ICARO gives priority to avoiding buffer under-runs, it can obtain higher cost savings when the ratio between requirements and available capacity is lower. Thus, since ζ is always lower than 0.05, the algorithm is able to effectively trade off cost efficiency for robustness and it is able to achieve up to 30% cost reduction when the conditions are favorable, but it never behaves too aggressively when the future capacity estimation does not allow to do so.

Fig. 8 compares the results of the last algorithm OPT(x) against the length of the prediction horizon x for both vehicular (left) and pedestrian mobility (right). We plot the results of simulations run for $\sum_i d_i / \sum_i c_i = 0.3$ and $(B_M \sum_i c_i / \sum_i d_i) = 100$. OPT($x$) is plotted as a solid black line and clearly shows that performance improves with increasing x, starting from about the value achieved by FULL (red dash-dotted line) and reaching the OPT (blue solid line) performance at about 2 and 10 minutes prediction horizon for vehicular and pedestrian mobility respectively.

In addition, we plot ICARO performance (green dashed line) and one vertical line to mark the 1 minute horizon, which is often used (e.g.: [12]). ICARO is performs very close to the optimal algorithm with 1 minute horizon for vehicular mobility and outperforms it for pedestrian.

Fig. 9 plots the CDFs of $\Delta_\xi = \xi_{\text{ICARO}} - \xi_{\text{OPT}(x)}$ (left) and $\Delta_\zeta = \zeta_{ICARO} - \zeta_{\text{OPT}(x)}$ (right) for $x = 60$ seconds and considering all the parameters range. We colored in blue the part of the curve where ICARO is achieving better results ($\Delta_\xi < 0$ or $\Delta_\zeta < 0$) and red otherwise. Again, ICARO clearly outperforms the optimal algorithm with limited prediction horizon when

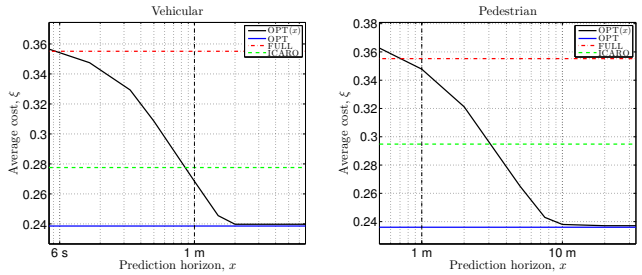

Fig. 8. OPT(x) performance against prediction horizon x compared with the other algorithms for vehicular (left) and pedestrian mobility (right).

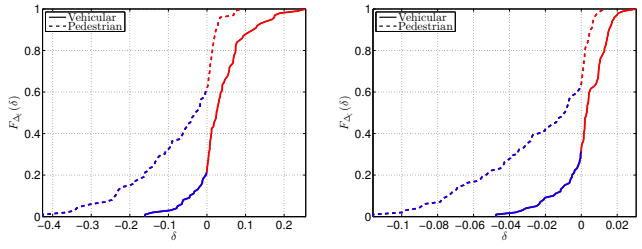

Fig. 9. Performance gap CDF between ICARO and OPT(x) for $x = 60$ seconds (e.g.: [12]). Cost and buffer under-run time gaps on the left and right respectively.

the mobility is pedestrian (average cost gap $E[\Delta_\xi] \approx -7.1\%$ average buffer under-run time gap $E[\Delta_\zeta] \approx -0.02$), while the two solutions performs very close in the case of vehicular mobility ($E[\Delta_\xi] \approx 3.4\%$ and $E[\Delta_\zeta] \approx 0$).

Combining close to optimal performance and low complexity, ICARO shows that combining short term prediction with medium-long term statistical consideration makes for a robust solution in prediction-based resource optimization. Finally, compared to the wide-spread full buffer strategy an ICARO-based system is able to sustain the same quality of service while saving up 30% of the network resources or, analogously, 30% more users can be served with the same capacity.

VIII. Conclusion

In this paper we addressed the problem of resource optimization for mobile networks under imperfect prediction of future available capacity. To this aim we developed a general prediction model that encompass short to medium-long term forecast. This joint estimation technique is the basis for ICARO, a lightweight resource optimization algorithm which achieves close-to-optimal performance. It achieves nearly optimal performance for vehicular mobility and outperforms the state of the art solution under pedestrian models. In addition, ICARO is effective in the robustness/efficiency trade off. As a final remark, ICARO is a practical resource optimization algorithm that does not require highly complex prediction techniques as it is only based on autoregressive filtering and simple statistic considerations. As future work we are extending ICARO to multi-user multi-quality scenarios, where different user prediction and decisions will be intertwined.

Acknowledgment

The research leading to these results was partly funded by the European Union under the project eCOUSIN (EU-FP7-318398) and by the Madrid Regional Government through the TIGRE5-CM program (S2013/ICE-2919).

References

[1] T. Chiang, "What's expected of us," *Nature*, vol. 436, no. 7047, pp. 150–150, 2005.

[2] S. Wang, Y. Xin, S. Chen, W. Zhang, and C. Wang, "Enhancing spectral efficiency for LTE-advanced and beyond cellular networks [Guest Editorial]," *IEEE Wireless Communications*, vol. 21, no. 2, pp. 8–9, April 2014.

[3] U. Paul, A. P. Subramanian, M. M. Buddhikot, and S. R. Das, "Understanding traffic dynamics in cellular data networks," in *IEEE INFOCOM*, Shangai, China, April 2011, pp. 882–890.

[4] M. Z. Shafiq, L. Ji, A. X. Liu, and J. Wang, "Characterizing and modeling internet traffic dynamics of cellular devices," in *ACM SIGMETRICS*, New York, NY, USA, 2011, pp. 305–316.

[5] M. C. Gonzalez, C. A. Hidalgo, and A.-L. Barabasi, "Understanding individual human mobility patterns," *Nature*, vol. 453, no. 7196, pp. 779–782, 2008.

[6] Z. Lu and G. de Veciana, "Optimizing stored video delivery for mobile networks: The value of knowing the future," in *IEEE INFOCOM*, Turin, Italy, April 2013, pp. 2706–2714.

[7] H. Abou-zeid, H. Hassanein, and S. Valentin, "Energy-efficient adaptive video transmission: Exploiting rate predictions in wireless networks," *IEEE Transactions on Vehicular Technology*, vol. 63, no. 5, pp. 2013–2026, June 2014.

[8] N. Sadek and A. Khotanzad, "Multi-scale high-speed network traffic prediction using k-factor gegenbauer arma model," in *IEEE ICC*, vol. 4, Paris, France, June 2004, pp. 2148–2152.

[9] Y. Qiao, J. Skicewicz, and P. Dinda, "An empirical study of the multiscale predictability of network traffic," in *IEEE HDPC*, Honolulu, Hawaii USA, June 2004, pp. 66–76.

[10] N. Bui, F. Michelinakis, and J. Widmer, "A model for throughput prediction for mobile users," in *European Wireless*, 2014.

[11] H. Abou-zeid, H. Hassanein, and S. Valentin, "Optimal predictive resource allocation: Exploiting mobility patterns and radio maps," in *Proc. IEEE GLOBECOM*, 2013.

[12] R. Margolies, A. Sridharan, V. Aggarwal, R. Jana, N. Shankaranarayanan, V. A. Vaishampayan, and G. Zussman, "Exploiting mobility in proportional fair cellular scheduling: Measurements and algorithms," in *Proc. IEEE INFOCOM*, 2014.

[13] H.-F. Geerdes, E. Lamers, P. Lourenço, E. Meijerink, U. Türke, S. Verwijmeren, and T. Kürner, "Evaluation of reference and public scenarios," IST-2000-28088 MOMENTUM, Tech. Rep. D5.3, 2003. [Online]. Available: http://momentum.zib.de/paper/momentum-d53.pdf

[14] M. Ahmed, S. Spagna, F. Huici, and S. Niccolini, "A peek into the future: predicting the evolution of popularity in user generated content," in *ACM WSDM*, Rome, Italy, February 2013, pp. 607–616.

[15] A. J. Nicholson and B. D. Noble, "Breadcrumbs: forecasting mobile connectivity," in *ACM MobiCom*, 2008.

[16] J. Froehlich and J. Krumm, "Route prediction from trip observations," *SAE SP*, vol. 2193, p. 53, 2008.

[17] O. Østerbø, "Scheduling and capacity estimation in lte," in *IEEE ITC*, San Francisco, CA, USA, September 2011, pp. 63–70.

[18] S. Makridakis and M. Hibon, "ARMA Models and the Box-Jenkins Methodology," *Journal of Forecasting*, vol. 16, no. 3, p. 147, 1997.

[19] J. D. Hamilton, *Time series analysis*. Princeton university press Princeton, 1994, vol. 2.

[20] F. Michelinakis, N. Bui, G. Fioravantti, J. Widmer, F. Kaup, and D. Hausheer, "Lightweight mobile bandwidth availability measurement," in *Proc. IFIP Networking*, 2015.