# Mobility and Bandwidth Prediction as a Service in Virtualized LTE Systems

Morteza Karimzadeh[1], Zhongliang Zhao[3], Luuk Hendriks[1], Ricardo de O. Schmidt[1], Sebastiaan la Fleur[1], Hans van den Berg[1,2], Aiko Pras[1], Torsten Braun[3], and Marius Julian Corici[4]

[1]*University of Twente,* [2]*Netherlands Organization for Applied Scientific Research (TNO), The Netherlands*
[3]*University of Bern, Switzerland,* [4]*Fraunhofer, Germany*

***Abstract*—Recently telecommunication industry benefits from infrastructure sharing, one of the most fundamental enablers of cloud computing, leading to emergence of the Mobile Virtual Network Operator (*MVNO*) concept. The most momentous intents by this approach are the support of on-demand provisioning and elasticity of virtualized mobile network components, based on data traffic load. To realize it, during operation and management procedures, the virtualized services need be triggered in order to scale-up/down or scale-out/in an service instance. In this paper we propose an architecture called MOBaaS (*Mobility and Bandwidth Availability Prediction as a Service*), comprising two algorithms in order to predict user(s) mobility and network link bandwidth availability, that can be implemented in cloud based mobile network structure and can be used as a support service by any other virtualized mobile network service. MOBaaS can provide prediction information in order to generate required triggers for on-demand deploying, provisioning, disposing of virtualized network components. This information can be used for self-adaptation procedures and optimal network function configuration during run-time operation, as well. Through the preliminary experiments with the prototype implementation on the OpenStack platform, we evaluated and confirmed the feasibility and the effectiveness of the prediction algorithms and the proposed architecture.**

## I. Introduction

Mobile operators are currently focusing on providing technological solutions for the issues arising from the significant growth of data traffic due to the continuous increase of mobile users, devices and applications. Long Term Evolution (*LTE*) as the fourth generation (*4G*) cellular system, standardized by the 3rd Generation Partnership Project (*3GPP*), is the most promising approach to cope with this challenge. Providing high data rates as well as supporting high-speed mobility are LTEs momentous peculiarities. Even though LTE promises a faster and more efficient data network, its core network architecture still highly centralized and hierarchical, leading to high bandwidth requirement and processing load on core network nodes. Further, it increases delay and network resources consumption [1],[2]. The huge appreciation received by cloud computing technologies in latest years pushed mobile network operators to plan the adoption of virtualization in their future network aiming at the possibility to share a common infrastructure among them. The cloud computing model could be applied in mobile network systems in order to offer decentralized computing, smart storage, on-demand, elastic and pay-as-you-go services to third party operators and users.

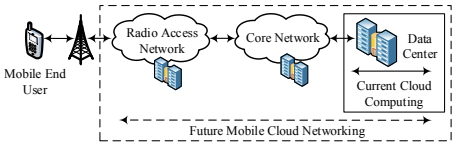



Fig. 1: A view of future LTE mobile cellular network.

The EU FP7 Mobile Cloud Networking (*MCN*) [3] project integrates the use of cloud computing concepts in LTE system with the objective of increasing LTEs performance by building a shared distributed mobile network and optimizing the utilization of computation, storage and networking resources. This vision, as shown in Fig. 1, can be realized by; (*i*) extending the cloud computing concept beyond the typical (*macro*) data centers towards new smaller (*micro*) data centers that are distributed within the E-UTRAN (*Evolved UMTS Terrestrial Radio Access Network*) and the EPC (*Evolved Packet Core*), and (*ii*) deploying and running cloud-based E-UTRAN, and EPC denoted as RAN as a Service (*RANaaS*) and EPC as a Service (*EPCaaS*), respectively. The most important intents by this approach are the support of on-demand deploying, provisioning, disposing of virtualized LTE system components, the support of resource allocation on-demand and dynamically. As the duration of scaling or virtual machine (*VM*) migration procedures are higher than the delay acceptable for the communication of a subscriber (*e.g., booting up of a VM may take 10-20s while the connectivity of a handed-over subscriber is lost after 30-50ms in LTE due to too large buffering required*), novel mechanisms apart from reacting to metered values by the monitoring system are required [4]. In order to give enough time for the system to adapt, one possible mechanism is adjusting the threshold limits far enough from the trigger points, to spot early during decision making procedures. However, this mechanism is highly prone to errors especially when the duration of a specific trend is short. Additionally, it may bring a high-level of instability to the system by not being able to detect ping-pong effects. Another alternative way is prediction. Utilizing an appropriate prediction mechanism in order to estimate the system change state in the future, can enable to place the thresholds at more extreme values (*referring to a future state*), and thus to tune a more accurate threshold value during the decision making process.

This paper proposes a prediction system architecture, entitled MOBaaS, which can be implemented in cloud based

978-1-4673-9501-4/15/$31.00 ©2015 IEEE

infrastructures and can be used as a support service by any other virtualized LTE component. MOBaaS encompasses two algorithms that can provide prediction information about the user(s) mobility and link bandwidth availability in the particular future. The prediction information provided by MOBaaS can be interpreted as an estimation on geographic and temporal traffic distribution. In particular, this paper focuses on addressing following challenges associated to the realization of MOBaaS in cloud based LTE system, which are in a close relationship with the requirements to be fulfilled by the MOBaaS system in [4]:

- Proper mechanism to estimate (*with higher accuracy*) the location of user(s).
- Proper mechanism to estimate (*with higher accuracy*) network link bandwidth availability.
- Some of the most important design considerations for MOBaaS, so that it can easily be instantiated, deployed and disposed in cloud computing platform and be scalable when the number of requests and overall computational load increases.
- A reference architecture for MOBaaS that can be simply integrated with any other virtualized LTE component to provide prediction information.

The remainder of the paper is organized as follows. Section II presents the motivation behind MOBaaS, introduces its architecture and components and describes design considerations. Section III and IV discuss the proposed mobility and bandwidth prediction algorithms respectively, demonstrate the obtained results and present some example use cases. Finally, section V concludes the paper and makes recommendations for the future work.

## II. MOBaaS Overview

### A. Motivations and Service Requirements

Different telecommunication services could benefit from the prediction results to optimize network performance. For example, Information Centric Network (*ICN*), which provides distributed storage, caching and content relocation features, see Fig. 2, could optimize the distribution of content according to the user mobility prediction results, such that the user content will be available/stored in the location he/she will visit in the future [5]. This information could be used by RAN in order to instantiate, deploy, upgrade and scale resources on-demand where and when they are required, and releasing them when not needed anymore [6], or during mobility management procedure to support service continuity and seamless mobility [7], as well. In addition to user mobility prediction, estimation of used and available bandwidth will also bring benefits to mobile telecommunication applications. EPC, for instance, could allocate network resources and bandwidth in a certain area in a more targeted manner [8].

MOBaaS, as a support service, can be used by any MCN service such as RANaaS, ICNaaS and EPCaaS [3], in order to provide prediction information or generate necessary triggers for self-adaptation procedures, *e.g.,* scaling of service instance components and optimal network function placement. This is realized by defining one entry point, called Frontend, which

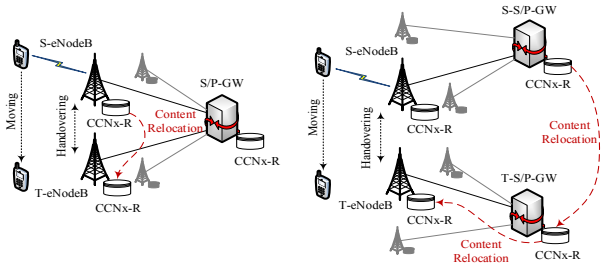



Fig. 2: ICNaaS content relocation based on MOBaaS output.

can either respond with prediction results or issue a trigger when certain thresholds are met, see Fig. 3. MOBaaS targets at providing prediction information of: (*i*) the estimated locations (*cells*) of an individual/a group of end-users, and (*ii*) predicted information about the bandwidth used/available at a certain network link in the future. In order to make prediction, MOBaaS requires a significant amount of users mobility and network link bandwidth usage information, which could be provided by another MCN support service, called Monitoring as a Service (*MaaS*). Given the above description, MOBaaS defines the following design requirements:

- An entry point for receiving requests of prediction (*request-based and trigger-based*)
- A connection to MaaS for retrieving history data.
- A mobility prediction algorithm block.
- A bandwidth prediction algorithm block.

Based on the proposed architecture, we propose a mobility prediction algorithm that provides prediction information about the next location(s) that user(s) may visit in the future. Mobility prediction can be requested and performed for an individual/a group of end-user(s) and may be utilized by any other MCN service. As an example, ICNaaS could utilize the user mobility prediction information to decide on content relocation. Based on the network topology design and implementation, the content may be relocated on the CCNx routers/repositories which are placed on the eNodeBs and/or the S/P-GWs (*Serving/PDN-Gateway*). When a user moves from one cell (*source eNodeB*) to another cell (*target eNodeB*) which are served by the same S/P-GWs, the content may be relocated between the CCNx routers/repositories implemented in the eNodeBs. If the source and target eNodeB are not served by the same S/P-GWs, the content may be moved between the CCNx routers/repositories which are placed in the target S/P-GW, see Fig. 2.

In addition to user mobility prediction, estimation of the used and available bandwidth will also bring benefits to cloud-based mobile network. For instance, allocation of network resources and bandwidth in a certain area could be performed more intelligently, given the knowledge that the bandwidth demands are higher than in the normal case. In this regard, we propose a bandwidth prediction algorithm using a sufficient amount of traffic usage information, to estimate the available bandwidth of a certain network link in a specific time. This information could also be integrated to the results of the mobility prediction algorithm, to estimate the total bandwidth in a particular network coverage area.

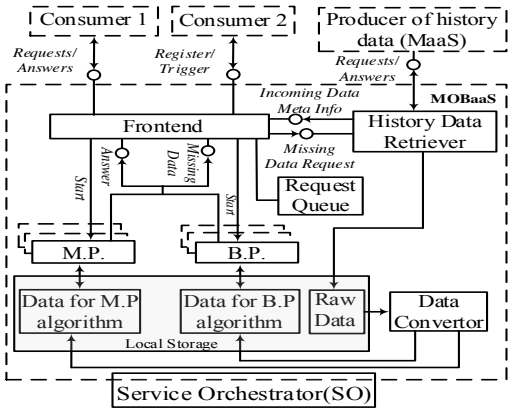



Fig. 3: MOBaaS implementation architecture.

### B. MOBaaS Reference Architecture Model

Fig. 3 demonstrates different components of MOBaaS and their roles are listed below:

- *M.P./B.P.*: These blocks represent the mobility prediction and bandwidth prediction algorithms, respectively.
- *History Data Retriever*: The thread to continually retrieve raw monitored data from MaaS. It also maintains a copy of a users' mobility and network links' traffic usage data history in the local data storages by deleting old or unused history.
- *Data Converter* : A logic that defines rules about how to process raw monitored data and convert them to the format usable by the algorithms.
- *Data for M.P./B.P. algorithm*: The processed history data in a format usable by the algorithms.
- Request Queue: A library on how to queue a request when specific data is missing in the history.
- *Frontend*: It is the main thread which ties all internal components of MOBaaS together. It handles prediction requests and starts algorithm threads, as well. If the Frontend receives a prediction request for which its required history information is currently not available, it will store the request to the "Request Queue", and that request will be postponed until the missing data becomes available. The Frontend also handles the trigger-based prediction, which means it periodically calculates the future states of users' movement and network links' bandwidth and decides if the registered consumers need to be informed.
- *Service Orchestrator (SO)* : The SO is responsible for managing the service instances of MOBaaS, which includes service initialization, disposal, run-time scaling operations, etc. It also calls the Frontend program to start the internal logics of MOBaaS.

### C. Mobility and Bandwidth Prediction As a Service

As a cloud-based supporting service, MOBaaS should provide requested information on-demand. Accordingly, several cloud computing principles have to be supported, in which on-demand service provisioning and scalability are the most important two issues.

The proposed architecture fits well for virtualization in a cloud environment and the different MOBaaS components can be provisioned on-demand, which includes Frontend, mobility prediction and bandwidth prediction algorithms, data storage, and other related functions. The configuration of internal components and the adapters related to the specific services can also be easily modified and updated if it is required. Furthermore, SO can release all the related resources after the service's life-cycle.

Scalability is another critical concern for MOBaaS, which should be able to automatically scale-up/down and scale-in/out depending on the number of prediction requests. The current proposed architecture can only handle vertical scaling for both processing power and storage. The architecture is not suitable for horizontal scaling due to the dependence on a local data storage. One of the responsibilities of the Frontend is to restrict the number of calculation threads. It communicates with the SO when there is need more processing power due to the increased amount of requests. Whenever the SO observes that the computing overheads for making prediction is getting high and are above certain threshold values, a scale-up decision will be made. Requesting history mobility and bandwidth data requires a lot of storage. Depending on the number of requests coming in and the amount of storage available, SO can scale-up the storage to have more resources available.

## III. Mobility Prediction

A large number of different algorithms have been proposed in the literature for predicting the future position of mobile user(s) in mobile networks. Most often, the prediction mechanisms have been developed to address a prediction problem for a specific type of scenario, which mainly are based on location, movement history, movement patterns, velocity, and etc., see *e.g.,* [9],[10]. Generally speaking, propounded schemes carry out forecasting based on mobility models that can be categorized in three main classes: *Temporal Dependency*, *Spatial Dependency*, and *Geographic Restriction* [10]. The mobility models represent the movement of mobile nodes, and how their location, velocity and acceleration change over time. Prediction schemes based on *Temporal Dependency* mobility model assume that mobile node trajectories may be constrained by some physical characteristics such as acceleration, velocity, direction, and also affected by their movement history, see *e.g.,* [11],[12]. In case of the latter, estimation is performed based on the assumption that mobile nodes desire to travel in a correlated manner and mobility of one node is affected by the mobility pattern of other neighboring nodes, see *e.g.,* [11],[13]. The solutions relying on *Geographic Restriction*, assume that node trajectories are subject to the environment and motion of mobile nodes are bounded by geographic restrictions such as freeways, local streets, buildings and other obstacles, see *e.g.,* [14],[15].

One of the intuitive methods to determine a mobile user's movement pattern, leading to predict its future behavior, is the attempt to trace and capture some sort of regularity in the user mobility. Many studies in the field, show that most often a mobile user has the tendency to regularly behave the same way.

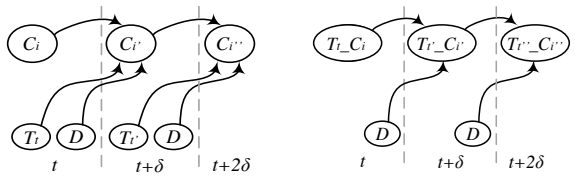
Such behaviour regularity could be considered as user's profile and utilized to estimate places a user may visit in the future. The prediction schemes, relying on history of mobile node trajectories, are classified in *Temporal Dependency* category. The mobility prediction algorithm proposed in this paper falls into this category.

In the context of the LTE system, the Mobility Management Entity (*MME*) is one of the key components of the EPC. It is responsible for variety of important key functions (*e.g.,* subscribers authentication, keeping location information of users, gateway, roaming and handovers). By monitoring and tracing information from MME, it would be possible to derive movement history of users.

The next section presents the mobility prediction scheme proposed in this paper. This approach utilizes the trace of mobile user trajectories, to predict the next location that may be visited.

### A. Description of Mobility Prediction Algorithm

The proposed mobility prediction algorithm benefits from the Dynamic Bayesian Network (*DBN*) model presented in [16]. The rationale behind using DBN is that, the next location (*cell*) visited by a user depends on: (*i*) it's current location, (*ii*) the current time, and (*iii*) the day of the week that the user is in the movement. The proposed DBN model is illustrated in Fig. 4.



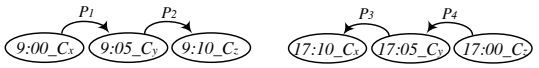

(a) DBN model (b) Modified DBN model

Fig. 4: The model used to drive the Markov Chain states.

In Fig. 4, $C_i$ represents a cell with ID *i*, $T_t$ defines the time of day $D$, $D$ shows the day of the week (*e.g.,* Monday), and $\delta$ determines the future time. As it is shown in Fig. 4a, the conditional distribution of the next location (*cell*) visited by a user comprises the current location, time, and the week day.

$$P(C(t+\delta) = C_{i'} \mid C(t) = C_i,\ T(t) = T_t,\ D) \tag{1}$$

Expression (1) can be considered as a location dependent distribution, providing a given time and day, and can be modeled as a simple first order Markov Chain (*MC*) that encodes the frequency (*probability*) of transitions between the cells. The DBN model can be simplified by integrating the transition time step (*e.g., each minute*) and the cell ID to derive a customized MC model, see Fig. 4b. In order to derive the transition probability matrix of the MC, we calculate the probability of moving from one cell to other(s) for each individual user, by counting the number of transitions from one cell to another, in regular tunable intervals (*e.g., each minute*) on the given days of the week (*e.g.,* all Mondays, all Tuesdays, and etc.) in the trace files. Fig. 5, as an example, shows two various transitions from one cell ($C_y$) to other cells ($C_x$ or $C_z$) derived in two different times in the customized MC model.




(a) The direction from home to work (*e.g., at morning)* (b) The return direction (*e.g., at evening)*

Fig. 5: Examples of customized MC states.

The spatial granularity of this algorithm is at the cell level, which means the algorithm outputs the possible future cell(s) of a user with particular probabilities. The temporal granularity of the algorithm is dependent on the application requirements and could be tuned in scale of minutes. In order to evaluate performance of the proposed algorithm, we used the mobility data trace provided by Nokia for academic research [17]. From this set we picked data from 100 users ranging over 2-6 months of time. For each user, we separated available data into two parts, the learning data set (*L*), and the testing data set (*T*). Data set *L*, is the 70% of user's data trace and utilized in order to derive the MC states and to calculate its transition probability matrix. Data set *T* contains the rest 30% of data trace, which is used to test and evaluate the proposed algorithm. For instance if the length of a data trace is 2.5 months (*i.e., it includes trace data for 10 Mondays*), we use the data trace of the first seven Monday during the learning phase and the rest for the test phase.

### B. Evaluation of Mobility Prediction Algorithm

The number of valid states (*with a time step and cell ID*) in the derived MC for each user depend on the quality of data trace in each day. In order to evaluate accuracy of the proposed algorithm, for each user we selected randomly $10\%$ of states, out of the MC states derived for each particular day of a week from the data set of *L*. Afterwards, for each of the selected states prediction calculation is performed to find the future possible cell(s) in the next $\delta$ minutes (*e.g., $\delta$=20*). These states have been chosen as the random test points (*times and ID of the cells that user was there*) during evaluation process. We checked the probability of possible transitions for the same selected states in the data set *T* as well. Afterward, the Mean Absolute Error (*MAE*) for the possible transitions in the corresponding test points, chosen from the learning and testing data sets, is computed in order to obtain accuracy of the prediction for each user in a particular day of the week.

$$MAE = \frac{1}{M}\sum_{i=1}^{M} |P_{iL} - P_{iT}|\ ,\ Accuracy = (1 - MAE) \times 100 \tag{2}$$

In Eq. (2), $M$ represents the number of all possible transitions

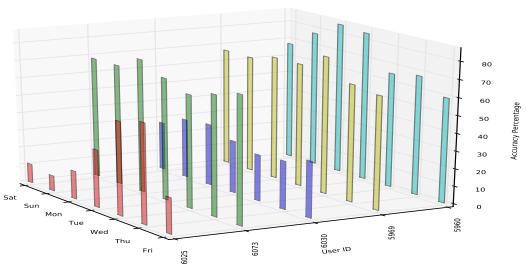



Fig. 6: Accuracy of algorithm for some users per day.

in the selected states, $P_{iL}$ and $P_{iT}$ define the calculated probability for each possible transition in the test points chosen from data set *L*, and the checked probability of possible transitions for the same selected states in the data set *T*, respectively. Fig. 6 shows accuracy of the prediction calculated per day, for some users as an example.

Fig. 7 displays the overall accuracy for 100 users. For each user it represents the average of accuracy calculated for each single day of the week.

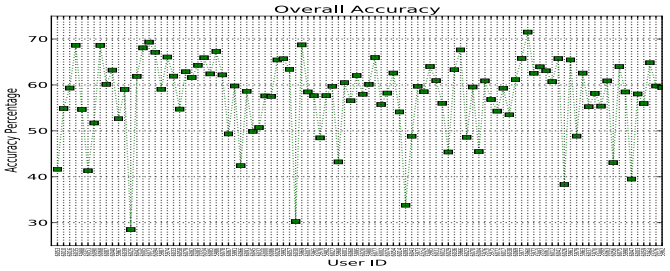
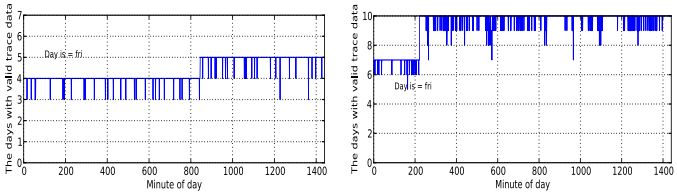



Fig. 7: The overall accuracy of prediction for 100 users.

Accuracy of prediction effectively pertains to the quality of data trace used to derive the transition probability matrix. Fig. 8, as an example, illustrates two users' data trace, leading to low (*for user 6025*) and high (*for user 5960*) prediction accuracy, see Fig. 7.




(a) user 6025 traced data (b) user 5960 traced data

Fig. 8: Quality of data trace for two different users.

### C. Mobility Prediction use cases

In the context of MCN project, mobility prediction information can be generated and provided in the following ways.

- *Single user and multiple users prediction*: Performing a prediction in this case is relying on a request-based approach, in that a HTTP server running at MOBaaS constantly waits for requests from consumers, see Fig. 3. For the single user request, prediction estimation is performed based on the user's current location (*cell ID*) at a given time (*time and day*). For the multiple users prediction, by providing a given time and day in the request message, the algorithm makes the prediction of which cell the users may be located at a certain future time. Moreover, it can provide details about where (*the cell IDs*) the users were moving.
- *Group users prediction*: Notification mechanism for the generated information in this case is based on the trigger-based approach. Group users prediction targets at providing probability distribution of the number of users in a certain cell at the given times (*e.g., t+δ, t+2δ, ...*), knowing the probability distribution of location(s) that users were at time (*t*). Eq. (3), defines the probability that *M* users may visit cell $C_i$ at time *(t+δ)*. The parameters used in this equation are listed in Table I.

TABLE I: Group users prediction parameters

| Parameter Name | Parameter Definition |
|---|---|
| $C = \{C_1, C_2, .., C_i\}, \lvert C\rvert = I$ | Set of cells |
| $U = \{U_1, U_2, .., U_j\}, \lvert U\rvert = J$ | Set of users |
| $N_{C_i}(t)$ | Number of users in cell $C_i$ at time *(t)* (*e.g.,* m) |
| $N_{C_i}(t+\delta)$ | Number of users in cell $C_i$ at time *(t+δ)* (*e.g.,* M) |
| $n_1$ | Number of users that may move to $C_i$ at time *(t+δ)* |
| $n_2$ | Number of users that may move from $C_i$ at time *(t+δ)* |
| $\triangle m = M - m = n_1 - n_2$ | The difference between the number of users in cell $C_i$ at time *(t)* and *(t+δ)* |
| $F_{C_{i'}(t)}$ | Subset of all users out of $C_i$ at time *(t)* |
| $F_{C_i(t)}$ | Subset of all users in $C_i$ at time *(t)* |
| $P_{j_1}$ , $U_{j_1}$ ∈J | $P\{U_{j_1}\ is\ in\ C_{i'}\ at\ time\ (t)\} \times P\{U_{j_1}\ moves\ to\ C_i\ at\ time\ (t+\delta)\}$ |
| $P_{j_2}$ , $U_{j_2}$ ∈m | $P\{U_{j_2}\ is\ in\ C_i\ at\ time\ (t)\} \times P\{U_{j_2}\ moves\ from\ C_i\ at\ time\ (t+\delta)\}$ |

$$P\left\{N_{C_i}(t+\delta)=M\right\} = \sum_m P\left\{N_{C_i}(t+\delta)=M \mid N_{C_i}(t)=m\right\} \times$$
$$P\left\{N_{C_i}(t)=m\right\} = \sum_m \Big( \sum_{n_1,n_2,n_1-n_2=\triangle m} P\left\{N_{in,C_i}(t+\delta)=n_1\right\} \times P\left\{N_{out,C_i}(t+\delta)=n_2\right\}\Big) \times P\left\{N_{C_i}(t)=m\right\} \quad (3)$$

In Eq. (3), $P\left\{N_{in,C_i}(t+\delta)\right\}$ and $P\left\{N_{out,C_i}(t+\delta)\right\}$, describe the probability of $n_1$ users that may move into cell $C_i$ and $n_2$ users that may move out from $C_i$ at time $(t+\delta)$, respectively. These probabilities can be calculated using Eq. (4).

$$P\left\{N_{in,C_i}(t+\delta)=n_1\right\} = \sum_{A_1 \in F_{C_{i'}(t)}} \prod_{j_1 \in A_1} P_{j_1} \prod_{j_1' \in A_1^c} \left(1 - P_{j_1'}\right)$$
$$P\left\{N_{out,C_i}(t+\delta)=n_2\right\} = \sum_{A_2 \in F_{C_i(t)}} \prod_{j_2 \in A_2} P_{j_2} \prod_{j_2' \in A_2^c} \left(1 - P_{j_2'}\right) \quad (4)$$

The group users prediction information could be integrated to network link bandwidth estimation and could be utilized by EPCaaS to optimize placement of the components in the cloud, or to dynamically adapt those components, while taking into account number of users and traffic loads.

## IV. Bandwidth Prediction

An important task for network operators is to properly provision the capacity of their links. Under-provisioned links might result in network performance degradation, which can ultimately affect Quality of Service (*QoS*) as perceived by end users. Aiming at adequate QoS, operators continuously monitor link usage. A commonly adopted approach is to read interface counters via Simple Network Management Protocol (*SNMP*) and use this information to roughly estimate required link capacity for current traffic [18]. While easy to use and readily available in most devices, the measured data lacks accuracy at shorter timescales, which leads mostly to loss of resources due to unnecessarily large safety margins. Higher accuracy for the estimation of required capacity at any timescale typically comes at the cost of more expensive measurements, such as packet-level. Operators however, tend to not deploy such approaches due to operational and financial challenges of such measurements. A compromise between easiness of

use and accuracy is achieved by approaches that use packet sampling or flow-level measurements for estimating required capacity. Next we describe our proposed solutions to estimate required capacity from commonly found measurement data nowadays, namely flows.

### A. Description of Bandwidth Prediction Algorithm

The starting point of our approaches is the dimensioning formula originally proposed in [19] and further validated in [20], [21]. Our dimensioning approach aims at link transparency, and assures the provided link capacity $C$ satisfies $A(T) \geq CT \leq \varepsilon$. The provided link capacity $C$ satisfy the condition $A(T)$, where $A(T)$ denotes the total amount of traffic arriving in intervals of length $T$, and $\varepsilon$ indicates the probability that the traffic rate $A(T)/T$ is exceeding $C$ at timescale $T$. The dimensioning formula requires that traffic aggregates are Gaussian (*i.e., $A(T)$ are normally distributed*) and stationary. The link capacity $C(T, \varepsilon)$, given by the adopted dimensioning formula, that satisfies the condition above, is calculated by adding to the mean traffic rate $\rho$ a safety margin that depends on the traffic variance $v(T)$ of $A(T)$:

$$C(T, \varepsilon) = \rho + \frac{1}{T}\sqrt{-2\log(\varepsilon) \cdot v(T)} \tag{5}$$

Relying on the variance $v(T)$, this dimensioning formula is able to take into account the impact of possible traffic bursts on the required link capacity. In addition, it is very flexible: network operators can choose $T$ and $\varepsilon$ according to the QoS level they want to provide to their users. Although accurate, the dimensioning approach from [19] originally requires continuous packet capture to calculate $\rho$ and $v(T)$. We developed alternative approaches to estimate these statistics from measurement data largely found at today's networks: flow-level data. Note that our definition of flows follows the typical 5-tuple of NetFlow v5 [22], consisting on source and destination IP, source and destination ports and transport layer protocol. Due to its aggregation nature, this type of data offers less information by nature, and mainly the estimation of $v(T)$ from this data becomes a challenge. Flow data, such as NetFlow [22] or IPFIX [23] flows (*or equivalents such as J-Flow [24]*) aggregates the information of packets belonging to the same connection and, therefore, it lacks information on individual packets size and their inter-arrival times. We developed two approaches to estimate required capacity from flows: a pure flow-based approach and a hybrid flow-based approach.

The pure flow-based approach, initially proposed in [25], estimates required capacity solely from flow-level measured data under the (*naive*) assumption of uniformly distributed bytes within flows. Traffic variance $v(T)$ is calculated from a flow-level time series and then applied to Eq. (5). This approach results in accurate estimations of required capacity at timescales as short as 1s. The hybrid flow-based approach [26], however, is able to accurately estimate required capacity at the millisecond timescale. This approach combines flow data with mathematical models that model the behavior of individual packets within flows. The downside of the hybrid approach is that for tuning the parameters of the models, we need occasional and short term packet captures. Nonetheless, once tuned, parameters remain valid for very long periods (*up to months*), making the hybrid approach measurement-wise lightweight if compared to a fully packet-based approach.

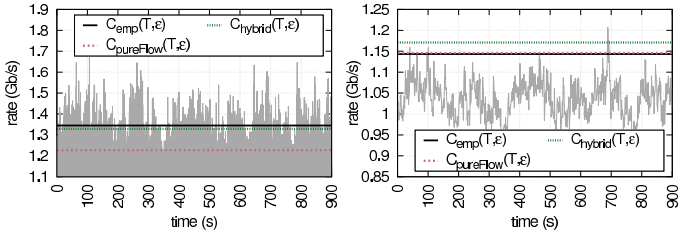
### B. Evaluation of Bandwidth Prediction Algorithm

In the following, we present some results on the validation of the developed approaches. Results from more extensive validations can be found at [27]. To validate the accuracy of the flow-based link dimensioning approaches we used packet traces captured from real ISPs networks. The packet traces allowed us to calculate the ground-truth $C_{emp}(T, \varepsilon)$, which is defined by the $(1-\varepsilon)$-quantile of the empirical CDF of the aggregated data rate, given by :

$$C_{emp}(T, \varepsilon) := \min\{C : \#\{A_i(T) \mid A_i(T) > CT\}/n \leq \varepsilon\} \tag{6}$$

$A_1(T), \ldots, A_n(T)$ are the $n$ empirical traffic aggregates on timescale $T$, and $\varepsilon$ is the bandwidth exceedance probability. In these experiments we have set in the dimensioning formula $\varepsilon = 0.01$ and $T$ to 10ms and 1s. We processed the same packet traces using YAF [28] to obtain the flow-level data of the captured traffic. Note that for the flow creation we used active timeout of 60s and inactive timeout of 20s. From the created flow data, we estimated $C_{pureFlow}(T, \varepsilon)$ and $C_{hybrid}(T, \varepsilon)$ using, respectively, the pure flow-based and the hybrid approaches.

Fig. 9 shows the data rate time series at $T = 10$ms and $T = 1$s for an example trace from our dataset, and the estimations of required capacity using the different flow-based approaches. This figure clearly shows that, on the one hand, at very small $T$ the pure flow-based approach is not accurate enough, since $C_{pureFlow}(T, \varepsilon)$ is lower than the minimum required capacity defined by $C_{emp}(T, \varepsilon)$. At larger $T$ though, this straightforward approach does provide $C_{pureFlow}(T\varepsilon) \equiv C_{emp}(T, \varepsilon)$. On the other hand, the hybrid approach supported by the mathematical models with tuned parameters accurately estimated required capacity at $T = 10$ms.




(a) $T = 10$ms (b) $T = 1$ms

Fig. 9: Estimation of required capacity for each of the proposed approaches, at different $T$, using a sample traffic trace from our data set. At any of the considered values of $T$, this sample trace has $\gamma > 0.9$ (*i.e., the traffic is sufficiently Gaussian*).

### C. Bandwidth prediction use cases

Within the context of the MCN project, or cloud infrastructures in general, bandwidth prediction is applicable in various scenarios. Due to the dynamic nature of the RANaaS and EPCaaS, where resources will be scaled up/down, dimensioning links properly in a timely and accurate manner is essential. Not only is bandwidth a resource itself, it is also necessary

to enable other resources to be deployed. For example, VMs that need to be moved to other parts of the network to realize service continuity, have bandwidth requirements on certain links. With a priori known bandwidth capacities of links, the infrastructure is aware of available bandwidth at a specific point in time. This information can then be used to choose certain deployment strategies over others, possibly reducing costs or improving end-user experiences.

## V. Conclusion

Implementing cloud-based LTE systems requires on-demand deploying, provisioning, and disposing of virtulized LTE components. In this paper, we proposed an architecture for MOBaaS, as a service component, which provides user(s) mobility and link bandwidth availability estimation only based on users' trajectory information and network link' bandwidth usage data. The proposed algorithms attempt to capture some sort of regularity in the data trace to predict the future behavior of the mobile user and the network link. The designed architecture have been successfully implemented on the OpenStack platform as proposed and used by the MCN architecture. Extensive experiments have been performed and evaluation results show that the prediction system is feasible and the proposed prediction algorithms achieve high accuracy. In the future, we plan to analyze the performance of MOBaaS in scenarios with other virtulized LTE services, such as ICNaaS and EPCaaS.

## Acknowledgements

The research work presented in this paper is conducted as part of the Mobile Cloud Networking project, and has been funded by the European Union Seventh Framework Program under grant agreement [# 318109].

## References

[1] Philippe Bertin, Servane Bonjour, and J Bonnin. Distributed or centralized mobility? In *Global Telecommunications Conference, 2009. GLOBECOM 2009. IEEE*, pages 1–6. IEEE, 2009.

[2] László Bokor, Zoltán Faigl, and Sándor Imre. *Flat architectures: Towards scalable future internet mobility*. Springer, 2011.

[3] EU FP7 Mobile Cloud Networking project, February 2015. http://www.mobile-cloud-networking.eu/site/.

[4] Georgios Karagiannis, Almerima Jamakovic, Keith Briggs, Morteza Karimzadeh, Carlos Parada, Marius Iulian Corici, Tarik Taleb, Andy Edmonds, and Thomas Michael Bohnert. Mobility and Bandwidth prediction in virtualized LTE systems: Architecture and Challenges. In *Networks and Communications (EuCNC), 2014 European Conference on*, pages 1–5. IEEE, 2014.

[5] Triadimas Arief Satria, Morteza Karimzadeh, and Georgios Karagiannis. Performance evaluation of ICN/CCN based service migration approach in virtualized LTE systems. In *Cloud Networking (CloudNet), 2014 IEEE 3rd International Conference on*, pages 461–467. IEEE, 2014.

[6] Lucio Studer Ferreira, Dominique Pichon, Atoosa Hatefi, Andre Gomes, Desislava Dimitrova, Torsten Braun, Georgios Karagiannis, Morteza Karimzadeh, Monica Branco, and Luis M Correia. An architecture to offer cloud-based radio access network as a service. In *Networks and Communications (EuCNC), 2014 European Conference on*, pages 1–5. IEEE, 2014.

[7] Luca Valtulina, Morteza Karimzadeh, Georgios Karagiannis, Geert Heijenk, and Aiko Pras. Performance evaluation of a SDN/OpenFlow-based Distributed Mobility Management (DMM) approach in virtualized LTE systems. In *Globecom Workshops (GC Wkshps), 2014*, pages 18–23. IEEE, 2014.

[8] Algorithms and Mechanisms for the Mobile Network Cloud, MCN D4.3. In *European Commission, EU FP7 Mobile Cloud Networking public deliverable, available at[3]*. 2014.

[9] M Abo-Zahhad, Sabah M Ahmed, and M Mourad. Future location prediction of mobile subscriber over mobile network using intra Cell Movement pattern algorithm. In *Communications, Signal Processing, and their Applications (ICCSPA), 2013 1st International Conference on*, pages 1–6. IEEE, 2013.

[10] Fan Bai and Ahmed Helmy. A survey of mobility models. *Wireless Adhoc Networks. University of Southern California, USA*, 206, 2004.

[11] Radhika Ranjan Roy. Autoregressive Individual Mobility. In *Handbook of Mobile Ad Hoc Networks for Mobility Models*, pages 775–789. Springer, 2011.

[12] Ben Liang and Zygmunt J Haas. Predictive distance-based mobility management for PCS networks. In *INFOCOM'99. Eighteenth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE*, volume 3, pages 1377–1384. IEEE, 1999.

[13] Xiaoyan Hong, Mario Gerla, Guangyu Pei, and Ching-Chuan Chiang. A group mobility model for ad hoc wireless networks. In *Proceedings of the 2nd ACM international workshop on Modeling, analysis and simulation of wireless and mobile systems*, pages 53–60. ACM, 1999.

[14] Per Johansson, Tony Larsson, Nicklas Hedman, Bartosz Mielczarek, and Mikael Degermark. Scenario-based performance analysis of routing protocols for mobile ad-hoc networks. In *Proceedings of the 5th annual ACM/IEEE international conference on Mobile computing and networking*, pages 195–206. ACM, 1999.

[15] Jing Tian, Jorg Hahner, Christian Becker, Illya Stepanov, and Kurt Rothermel. Graph-based mobility model for mobile ad hoc network simulation. In *Simulation Symposium, 2002. Proceedings. 35th Annual*, pages 337–344. IEEE, 2002.

[16] Vincent Etter, Mohamed Kafsi, Ehsan Kazemi, Matthias Grossglauser, and Patrick Thiran. Where to go from here? Mobility prediction from instantaneous information. *Pervasive and Mobile Computing*, 9(6):784–797, 2013.

[17] Mobile Data Challenge (MDC) Dataset, December 2014. https://www.idiap.ch/dataset/mdc/.

[18] Cisco Systems Inc. How To Calculate Bandwidth Utilization Using SNMP. http://www.cisco.com/image/gif/paws/8141/calculate_bandwidth_snmp.pdf, 2005. Online, accessed Apr. 2014.

[19] Remco van de Meent. *Network Link Dimensioning: A measurement & modeling based approach*. PhD thesis, University of Twente, 2006.

[20] Hans van den Berg, Michel Mandjes, Remco van de Meent, Aiko Pras, Frank Roijers, and Pieter Venemans. QoS-aware bandwidth provisioning for IP network links. *Elsevier Computer Networks*, 50(5):631–647, 2006.

[21] Aiko Pras, Lambert Nieuwenhuis, Remco van de Meent, and Michel Mandjes. Dimensioning Network Links: A New Look at Equivalent Bandwidth. *IEEE Network*, 23(2):5–10, 2009.

[22] Benoit Claise. Cisco Systems NetFlow Services Export Version 9. RFC 3954, 2004.

[23] Benoit Claise, Brian Trammell, and Paul Aitken. Specifications of the IP Flow Information Export (IPFIX) Protocol for the Exchange of Flow Information. RFC 7011, 2013.

[24] Juniper Networks. Juniper Flow Monitoring. http://www.juniper.net/us/en/local/pdf/app-notes/3500204-en.pdf, 2011. Online, accessed Jun. 2014.

[25] Ricardo de O. Schmidt, Anna Sperotto, Ramin Sadre, and Aiko Pras. Towards Bandwidth Estimation using Flow Measurements. In *Proceedings of the 6th IFIP WG 6.6 International Conference on Autonomous Infrastructure, Management, and Security (AIMS)*, pages 127–138, 2012.

[26] Ricardo de O. Schmidt, Ramin Sadre, Anna Sperotto, Hans van den Berg, and Aiko Pras. A Hybrid Procedure for Efficient Link Dimensioning. *Elsevier Computer Networks*, 67:252–269, 2014.

[27] Ricardo de O. Schmidt. *Measurement-based Link Dimensioning for the Future Internet*. PhD thesis, University of Twente, 2014.

[28] Christopher M. Inacio and Brian Trammel. YAF: Yet Another Flowmeter. In *Proceedings of the 24th Large Installation System Administration Conference*, LISA'10, pages 1–12, 2010.