Capacity Planning of Fog Computing Infrastructures under Probabilistic Delay Guarantees

Ioanna Stypsanelli, Olivier Brun, Samir Medjiah, Balakrishna Prabhu

Capacity Planning of Fog Computing Infrastructures under Probabilistic Delay Guarantees

I. Stypsanelli, O. Brun, S. Medjiah, B. Prabhu

LAAS/CNRS,
7 Av. du Colonel Roche,
31077 Toulouse Cedex 4, France.

Abstract

Fog Computing infrastructures are deployed in the immediate vicinity of users in order to meet the stringent delay requirements of some emerging IoT applications, which cannot be achieved with traditional Cloud Computing infrastructures. The latency gains of Fog Computing come however at the cost of a potentially larger total capacity. The duplication of ressources in many micro data centres may also lead to an explosion of energy and operations costs. In this paper, we consider the optimal capacity planning of Fog Computing infrastructures under probabilistic delay guarantees. Despite the non-linearity of the delay constraints, we show that the problem can be formulated as a Mixed Integer Linear Programming (MILP) problem. We first present a MILP formulation of the problem assuming that the infrastructure cost depends linearly on the capacities. To account for economies of scale in favour of large data centres, we then extend this MILP formulation to arbitrary concave objective functions. Empirical results show that the optimal capacity-planning solution can be determined efficiently even for large-size problem instances, and that it can results in significant gains with respect to the solution in which user requests are always processed in the nearest data centre.

1 Introduction

In sharp contrast to traditional cloud services, many emerging applications require low and predictable latency. This calls for the extension of the classical centralized cloud computing architecture towards a more distributed architecture that includes computing and storage nodes installed close to users. As shown in Fig. 1, a Fog Computing (FC) architecture is a highly virtualized platform that provides a multitude of compute, storage, and networking resources at the edge of the network, allowing applications that depend on time-critical data to use nodes in their vicinity to meet the delay requirements [6,7,19]. The FC infrastructure may correspond for instance to an in-network distributed cloud built by telecom operators by distributing the cloud inside their network points of presence [2].

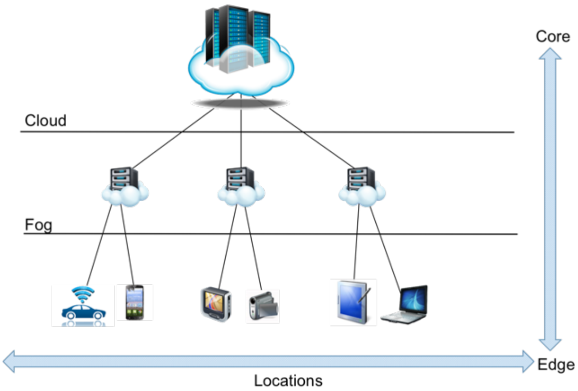

Figure 1: Fog Computing infrastructures.

There are a number of benefits expected from the transition to FC, including reduced latency and better quality of experience for users, preservation of network resources, greater security, privacy and resilience, as well as easier scalability [3,18,25]. FC is not intended to replace Cloud Computing, but rather to complement it by handling data managment, the cloud handling data analytics for its part. FC is also expected to enable a new breed of applications, in as diverse application domains as smart buildings and cities, health care and transportation, among others.

The Fog is however a non-trivial extension of the Cloud. Latency gains come at the cost of potentially larger total capacity because geo-distribution forfeits statistical multiplexing of demands that a single large data centre could benefit from. The duplication of distributed resources may also lead to an explosion of energy and operation costs. In practice, the design of a FC infrastructure has to balance two conflicting factors. The first one is the importance of geographic diversity: the goal is to place micro data centres close enough to users to meet delay requirements. The other one is the size of the data centre: the goal here is to amortize the fixed costs of the site while benefiting from statistical multiplexing gains by serving the workload generated by the local population. Finding an optimal trade off between geographic diversity and data-centre sizes is usually a challenging problem.

As an example, consider the simple scenario depicted in Fig. 2, in which two different base stations can route their traffic to two different micro data centres. The minimum latency is achieved with a distributed solution, in which the traffic of each base station is routed to the closest micro data centre, so that base station B_1 (resp. B_2) routes all its traffic to data-centre D_1 (resp. D_2).

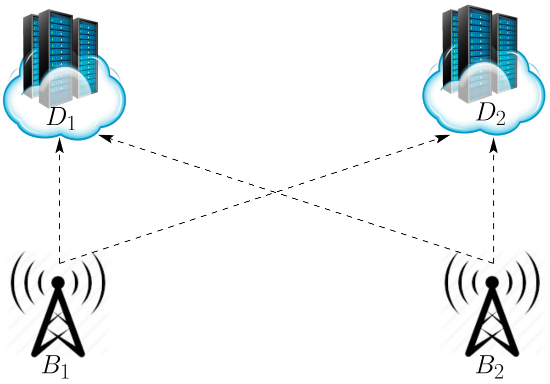

Figure 2: A simple scenario in which two base stations can route their traffic to two different micro data centres.

Now, assume that the daily pattern of the offered traffic at each base station is as shown in Fig. 3. In the distributed solution described above, the minimum amount of capacity to be provisioned at data-centre D_1 (resp. D_2) corresponds to the peak hour of traffic for base station B_1 (resp. B_2), so that the total capacity to be provisioned is for $240 + 240 = 480$ jobs/s. In contrast, in the centralized solution where only one data-centre is used, the total capacity to be provisioned should be only for 282 jobs/s. Thus the ratio of total capacities between the centralized and the distributed solutions is roughly $\frac{1}{2}$. In turns , this translates into an even lower ratio in terms of costs in favor of the centralized solution due to economies of scale which impact not only capacity costs, but also operating and energy costs. The issue is that, for some services with stringent delay requirements, a centralized solution might not be feasible.

In this paper, we address the capacity planning of micro data-centers used in Fog Computing. Three types of decisions have to be made. We need to decide where to install data-centres, how user-generated requests are routed to these data-centres, and the amount of capacity to be installed in each data-centre. We assume that the goal is to minimize an arbitrary concave function of the capacities installed in the data-centres under probabilistic delay guarantees for user requests. Despite the non-linearity of the delay constraints, we show that the problem can be formulated as a Mixed Integer Linear Programming (MILP) problem. We first present a MILP formulation of the problem assuming that the infrastructure cost depends linearly on the capacities. To account for economies of scale in favour of large data-centres, we then extend this MILP formulation to arbitrary concave objective functions. Empirical results show that the optimal capacity-planning

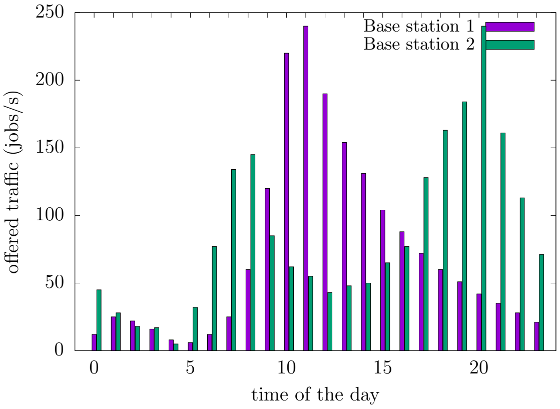

Figure 3: Daily pattern of the offered traffic by the two base stations, as a function of the hour of the day.

solution can be determined efficiently even for large-size problem instances. These results also show that significant gains can be obtained with respect to the solution in which user requests are always processed in the nearest data-centre, and with respect to the minimum-cost centralized solution.

The paper is organized as follows. In section 2, we motivate the specific FC architecture and the latency requirements considered in this paper. Section 3 introduces the notations and assumptions used in the paper, and formulates the problem as a mixed integer optimization problem with non-linear constraints. In Section 4, we consider the case where the system designer provisions resources for each service separately. We prove that the optimal capacities are the minimum solution of a system of linear inequalities. This result is then used to formulate the global problem as a MILP problem. Numerical results are then presented in Section 5. Finally, related works are discussed in Section 6 and some conclusions are drawn in Section 7.

2 Fog computing for connected vehicles

The main motivation for the work in this article comes from connected vehicles for which a large number of applications with varying needs of quality of service have been defined. Applications for connected vehicles can be classified into three categories [1]:

1. Active road safety applications: applications used to reduce the probability of traffic accidents and loss of lives. Examples include: intersection collision warning, head on collision warning, emergency vehicle warning, wrong way driving warning, signal violation warning, etc.

2. Traffic efficiency and management applications: these applications are employed for improving the traffic flow, traffic assistance, traffic co-ordination, updated local information, etc. Example include speed management, and cooperative navigation applications.

3. Infotainment applications: these are less constrained applications such as application that collect and disseminate information about locally based services such as points of interest (restaurants, hotels, etc.) or global internet services: multimedia services, parking management, etc.

Based on [1], these applications have different requirements, ranging from the most constrained: periodic messages, 10Hz frequency, and 100ms critical latency for active road safety applications to 1Hz frequency and 500ms critical latency for co-operative services.

2.1 Fog Computing Infrastructure Planning

The business model behind FC for connected vehicles applications in not clear yet. Indeed, multiple actors may be involved; car manufacturers, telco operators, road infrastructures operators, or even cloud operators. All of these different actors may have interest on building and operating a FC infrastructures for connected vehicles applications. For example, a telco operator (or a cloud operator) may be interested to operate such an infrastructure to optimize its primary services and augment its products catalogue by services especially tailored for connected vehicles. In this article, we propose a model that can be used by any one of the previously mentioned actors to build the Fog infrastructure from scratch by planning the number and the capacity of the future micro data-centers. This model can also be used to extend an already existing computing infrastructure. Indeed, existing data centers can be fixed into the model and the optimization problem will be solved for planning the capacity of the additional nodes to be built. This second configuration is likely more realistic since the Fog operator will have to deal with already existing resources. In other cases, existing (and specific) data centers must obligatorily be part of the final infrastructure for privacy, performances or cost requirements.

2.2 Network latency

In this article, it is assumed that vehicles are connected to the micro data-centres through a cellular network such as 4G/5G. Other options exist for connected vehicles such as DSCRC. In the case of cellular networks, it is necessary to consider two types of communications latencies: between the connected vehicle and the base station (i.e. the radio network) and the latency between the base station and the micro data-centre.

The first latency is strongly dependent on parameters such vehicle velocity, distance between the vehicle and the base station, density of users attached to the same base station, users' data traffics, etc. This latency can range from 10 to 100 ms [4]. For the second latency, we can distinguish two cases. In the case of the telco operator is also the Fog operator, the data centres are part of the core network. This latency can be up to 20 ms [16]. However, if the micro data-centre is not part of the core network of the telco operator, this latency can be very high depending on how the data centre network is interconnected with other networks. In this article, no assumption is made about whether the micro data-centres are part of the telco's core network or not. We have assumed application latencies of 60-150ms.

3 Problem statement

We are given as input a set $\mathcal{D}$ of potential sites for installing micro-datacenters, as well as a set $\mathcal{B}$ of base stations. The geographical locations of the micro-datacenters and of the base stations are known. The problem amounts to deciding what amount of capacity to install in each micro-datacenter and how to route the traffic originating from the base stations so as to obtain a minimum-cost infrastructure satisfying a number of performance requirements. We assume that the traffic generated by base stations varies over time, and that they can change the routing of their traffic from one time slot to the other. However the capacities of the datacenters have to be decided once and for all.

The infrastructure supports a set $\mathcal{S}$ of S job classes. Let $\lambda_i^{k,t}$ be the class-k traffic originating from base station $i \in \mathcal{B}$ at time $t = 1, 2, \ldots, \tau$. We define $x_{ij}^{k,t}$ as the amount of class-k traffic sent by base station i to micro-datacenter j at time t. These variables, which define the routing strategy at time t, have to satisfy the following constraints

$$
\begin{aligned}
\sum_{j \in \mathcal{D}} x_{ij}^{k,t} &= \lambda_i^{k,t}, & (1)\\
x_{ij}^{k,t} &\geq 0. & (2)
\end{aligned}
$$

We also define the binary variable $a_{i,j}^{k,t}$, which indicates whether base station i sends class-k jobs to datacenter j at time t, and the binary variable $u_j^{k,t}$, which indicates whether class-k jobs are routed to datacenter j, by imposing the following constraints on the feasible values of these variables

$$\sum_i a_{i,j}^{k,t} \leq |\mathcal{B}| u_j^{k,t}, \tag{3}$$

$$u_j^{k,t} \leq \sum_i a_{i,j}^{k,t}, \tag{4}$$

$$x_{i,j}^{k,t} \leq \lambda_i^{k,t} a_{i,j}^{k,t}, \tag{5}$$

$$u_j^{k,t} \in \{0,1\}, \tag{6}$$

$$a_{i,j}^{k,t} \in \{0,1\}. \tag{7}$$

Finally, the binary variable u_j defined by the following constraints

$$u_j^{k,t} \leq u_j, \quad \forall k, t, \tag{8}$$

will be used to determine whether data centre j has to be opened. Note that there is no need to enforce that $u_j = 0$ if $u_j^{k,t} = 0$ for all k and t because the objective functions that we consider are non-decreasing in u_j. In the following, the variables $x_{i,j}^{k,t}$, $a_{i,j}^{k,t}$, $u_j^{k,t}$ and u_j shall be referred to as the routing variables of the problem. A feasible routing strategy is a set of values for these variables satisfying (1)-(8).

The performance requirements are related to the quality of service of jobs processed by the servers of the FC infrastructure. The system designer aims at determining the capacities of the data centres in such a way that most class-k jobs be served in a maximum acceptable processing time T_k. More precisely, let $S_j^{k,t}$ be the processing time of class-k jobs at data centre j and at time $t = 1, 2, \ldots, \tau$, and let $\ell_{i,j}^k$ be the network time, that is, the time it takes to send a job request from base station i to data centre j plus the time it takes to receive the reply. For simplicity, we assume that $\ell_{i,j}^k$ is a fixed communication delay which does not depend on the network load. In contrast, the processing time $S_j^{k,t}$ is a random value whose distribution may depend on the load of the data center and on the class of the job. The term $S_j^{k,t} + \ell_{i,j}^k$ then represents the total time it takes for a class-k request sent by node i at time t to be received and processed by micro data-centre j, plus the time it takes to receive the reply.

The goal is to design the system in such a way that the probability that this time be strictly greater than T_k be lower than a given value δ_i, that is, in such a way that

$$\mathbb{P}\left(S_j^{k,t} + \ell_{i,j}^k \geq T_k\right) \leq \delta_k,$$

for all base stations i sending class-k jobs to micro data-centre j at time t. In other words, the above delay constraints should be enforced only for those i such that $a_{i,j}^{k,t} = 1$ and for those values of j, k and t such that $u_j^{k,t} = 1$. This can be done by imposing that

$$\mathbb{P}\left(S_j^{k,t} \geq T_k - \max_i \ell_{i,j}^k a_{i,j}^{k,t}\right) \leq \delta_k + 1 - u_j^{k,t}. \tag{9}$$

Note that if no class-k jobs are routed to site j at time t (that is, $u_j^{k,t} = 0$), the above constraint is redundant. Otherwise, it imposes that $\mathbb{P}\left(S_j^{k,t} \geq T_k - \max_i \ell_{i,j}^k a_{i,j}^{k,t}\right) \leq \delta_k$, as expected.

We shall assume that micro data-centres are equipped with homogeneous servers. We denote by $\frac{1}{\mu_k}$ the mean processing time of a class-k job on one of these servers. We also denote by c_j be the number of compute servers installed in site $j \in \mathcal{D}$. It has to be big enough so that the latency constraints (9) are satisfied. We assume the following cost structure:

- An opening cost β_j is incurred if capacities are installed in data centre j, that is, if $u_j = 1$.
- The cost of installing c servers in data centre j is $g_j(c)$, where g is a given continuous function, which is often chosen concave to express economies of scale in favour of large data centres. Note that $g_j(c)$ includes the cost of purchasing the capacity c, but can also include energy and maintenance costs for operating it.

The problem can now be formally stated as follows

$$\begin{aligned} \text{minimize} & \sum_{j \in \mathcal{D}} \beta_j \, u_j \; + \; g_j(c_j) \qquad \text{(CAPA)} \\ \text{subject to} & \;\; \text{constraints } (1) - (9). \end{aligned}$$

In this problem, we have non-linear constraints and binary variables which make the problem non-standard. The precise form of these constraints obviously depend on the queueing model which is assumed for data centres. We shall make it explicit in the following section.

4 Separate Resource Provisioning per Service

In this paper, we consider the case where the system designer provisions resources for each service separately. We let c_j^k be the number of compute servers provisioned for handling class-k jobs[1]. Obviously, we have

$$c_j = \sum_k c_j^k \tag{10}$$

[1] In the following, we consider c_j^k as a continuous parameter. In practice, the value that should be used is $\lceil c_j^k \rceil$.

We can analyse separately the optimal capacity to be provisioned for each class. As a consequence, in sections 4.1 and 4.2 below, we consider only one class of jobs, and we drop the index k.

4.1 Assumptions

We assume that job requests arrive according to a Poisson process and that an incoming job is routed with probability $1/c_j$ to any of the servers using what is known as Bernoulli routing, that is jobs are routed to server j with a probability that does not depend upon the number of tasks in the servers. This is different from state-dependent routing policies such as join-the-shortest-queue for which the number of jobs in each of the servers has to known to the dispatcher. The service time of a job on a server will be assumed to be exponentially distributed with mean $1/\mu$. Note that the service time is different from the processing time. Service time is the time it takes to finish a job if it were alone in the system whereas the processing time is the sum of the service time and the waiting time (the time to serve jobs that arrived before). In practice, service times need not be exponentially distributed. In this case, the analysis for the computation of processing time is involved and the expressions for the distribution of the response time are not easy to obtain. As a first work on this topic, we shall assume exponentially distributed service times.

It follows from these assumptions that the servers provisioned at datacentre j for the considered class are modelled as c_j parallel $M/M/1/\infty$ queues [15], and therefore that

$$\mathbb{P}\left(S_j^t \geq z\right) = e^{-(\mu - y_j^t/c_j)\,z}, \tag{11}$$

where $y_j^t = \sum_{i \in \mathcal{B}} x_{i,j}^t$ is the rate at which job requests arrive at data center j at time $t = 1, \dots, \tau$.

4.2 Optimal capacity for a fixed routing strategy

Our first step is to analyze the capacity required at a given data center, say j, for a fixed routing strategy satisfying (1)-(8). We first note that, for stability reasons, we should have $y_j^t/c_j < \mu$, that is, $c_j > y_j^t/\mu$. This condition is however not sufficient, as formally stated below.

Lemma 4.1. *There exists $c_j > y_j^t/\mu$ satisfying the latency constraint of jobs if and only if the routing strategy is such that*

$$l_{i,j} a_{i,j} < T - \frac{\log(\frac{1}{\delta})}{\mu}, \quad i \in \mathcal{B}, t = 1, \dots, \tau \tag{12}$$

Proof. With (9) and (11), we obtain

$$\frac{y_j^t}{c_j} \leq \mu - \frac{\kappa}{T - \ell_{i,j} a_{i,j}^t}, \tag{13}$$

where $\kappa = \log(\frac{1}{\delta})$. Inequality (13) ha a non-negative solution c_j if and only if the RHS is non-negative, that is, if and only if $l_{i,j} a_{i,j} < T - \frac{\kappa}{\mu}$ □

The condition in Lemma 4.1 merely imposes that $a_{i,j}^t = 0$ whenever $\ell_{i,j} \geq T$. Provided that this condition is met, Lemma 4.2 below gives the optimal number of servers to install in data center j for known values of the other variables.

Lemma 4.2. *Given the values of the routing variables, the minimum number of servers required to satisfy the latency constraint of jobs is*

$$c_j = \max_{t,i} \left\{ \frac{y_j^t}{\mu - d_{i,j}} a_{i,j}^t \right\}, \tag{14}$$

where $d_{i,j} = \log(\frac{1}{\delta}) / [T - \ell_{i,j}]$.

Proof. See Appendix 9. □

It follows from Lemma 4.2 that the optimal capacity at data center j is the minimum value satisfying the following linear inequalities

$$c_j \geq \frac{y_j^t}{\mu - d_{i,j}} - M\left(1 - a_{i,j}^t\right), \tag{15}$$

$$c_j \geq 0, \tag{16}$$

where M is any constant sufficiently large for the RHS of (15) to be negative whenever $a_{i,j}^t = 0$.

4.3 Linear objective function

In this section, we consider the case where $g_j(c) = \alpha_j \, c$, for some constant α_j. It directly follows from Lemma 4.2 that the optimal solution of problem (CAPA) is obtained by solving the following Mixed Integer Linear Programming (MILP) problem

$$\text{minimize} \sum_{j \in \mathcal{D}} (\beta_j \, u_j \; + \; \alpha_j \, c_j) \tag{CAPA-PL}$$

$$\text{subject to constraints } (1) - (8), (10), (12), (15) - (16).$$

4.4 Piecewise linear objective function

As mentioned in the introduction, the function $g_j(c)$ is often a non-linear concave function to express economies of scale in favour of large data centres. The minimization of a non-linear concave objective function is in general a challenging problem, in particular when integer variables are involved. However, with the increasing efficiency of MILP software tools, an interesting alternative is to use a piecewise linear (PWL) approximation of the original non-linear function.

The PWL approximation of a function $f(x)$ over an interval $[x_{min}, x_{max}]$ is obtained by introducing a number n of sampling coordinates $x_1, \ldots, x_n$ such that $x_1 = x_{min}$ and $x_n = x_{max}$. The function $f(x)$ is then approximated by the collection of linear segments $[(x_i, f(x_i)), (x_{i+1}, f(x_{i+1}))]$. Figure 4 illustrates the quality of the approximation obtained for two concave functions, $f_1(x) = \log(1 + x)$ and $f_2(x) = \frac{3}{2} + \frac{1}{4}\sqrt{x}$, over the interval $[0, 50]$.

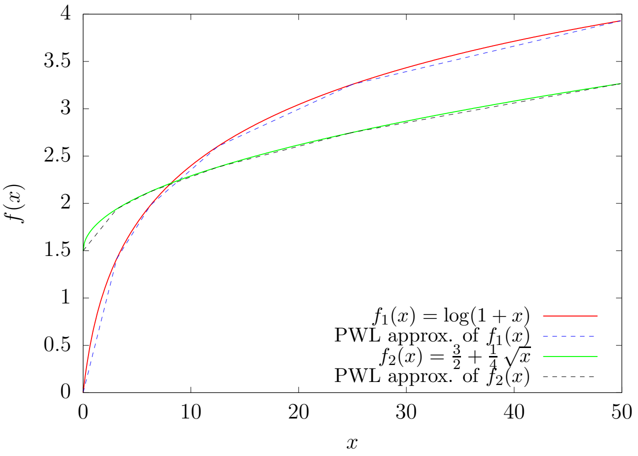

Figure 4: PWL approximations of the functions $f_1(x) = \log(1 + x)$ and $f_2(x) = \frac{3}{2} + \frac{1}{4}\sqrt{x}$ over the interval $[0, 50]$. The number of sampling coordinates is $n = 5$, and they have been generated as follows: $x_1 = 0$ and $x_i = 2^{-(n-i)}50$ for $i = 2, \ldots, n$.

If n is the number of linear segments of the approximation, the above technique can be applied to our problem by introducing n continuous variables and $n - 1$ binary variables, as described in [9]. We note however that most modern MILP solvers are capable of directly handling PWL objective functions, usually using the concept of Special Ordered Set.

5 Experimental Results

We now describe the results that were obtained with the proposed algorithms. We first consider a very simple scenario in Section 5.1. The numerical results obtained with a larger number of base stations are presented in Section 5.2.

5.1 Simple scenario

We first consider a simple scenario with three data centres and two base stations, which are located as shown in Fig. 5. The first two data centres, located in Aulnay-sous-Bois and Corbeil-Essones, in France, are potential data centres. The cost for opening them are $\beta_1 = \beta_2 = 100$, and the cost of one unit of capacity is $\alpha_1 = \alpha_2 = 1$. In contrast, the data centre in London is an existing large public data centre. Therefore, there is no opening cost associated to this data centre ($\beta_3 = 0$) and we assume that, due to economies of scale, the cost of an individual compute server is only $\alpha_3 = \frac{3}{4}$. Note that each base station is in immediate vicinity of a data centre. The distance from the base station 1 in Rosny-sous-Bois to the data centre in Aulnay-sous-Bois (resp. Corbeil-Essonnes) is 8.9 Km (resp. 32.6 Km), and the distance from the base station 2 in Evry to the data centre in Corbeil-Essonnes (resp. Aulnay-sous-Bois) is 4 Km (resp. 35.7 Km).

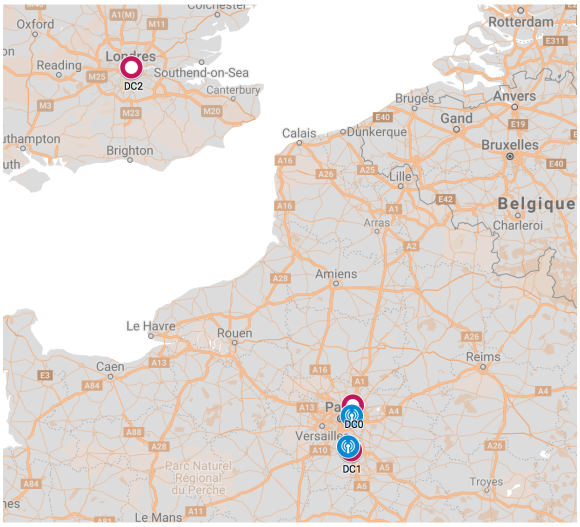

Figure 5: Locations of data centers and base stations.

The workload is composed of two classes of jobs. The first class of jobs correspond to real-time jobs, whereas the other ones are best-effort jobs with far less stringent requirements. The parameter values used in our

Table 1: Characteristics of job classes. Times are given in seconds, and n_k represents the number of packets send by a class-k request.

	$1/\mu_k$	T_k	δ_k	n_k
Class 1	0.01	–	–	2
Class 2.	0.1	2.0	0.1	6

Table 2: Communication times (ms) between base stations and data centres.

	Aulnay	Corbeil	London
Rosny	33 / 43	39 / 52	134 / 178
Evry.	41 / 54	31 / 41	140 / 187

experiments are given in Table 1. Note that the values of the maximum end-to-end latency T_1 and the threshold probability δ_1 for the real-time class are not given in Table 1, because we will vary their values in the following.

We also assume that the communication latency between two points at a distance of d kilometres from each other is $10 + 0.1 \times d$ ms, which, according to the idealized deterministic model in [5], yields the TCP transfer times given in Table 2, where in each cell the first (resp. second) value is the communication time for the first (resp. second) class of jobs. Regarding real-time jobs, the offered traffic of each base station evolves as shown in Fig. 3, but with values which are scaled by a factor 10. or simplicity, we assume that the class-2 offered traffic of each base station is constant over time, and equals to $2,000$ jobs/s.

We first consider the case where the infrastructure cost is linear in the data centre capacities, that is,

$$100 \times (u_1 + u_2) + c_1 + c_2 + \frac{3}{4} \times c_3 \tag{17}$$

Our goal is to compare three different solutions:

- the first one is the optimal solution, which is obtained as the solution of the MILP problem (CAPA-PL),
- the second one is the fully distributed solution in which each base station is assigned to the nearest data centre. This solution is obtained by adding to problem (CAPA-PL), for each base station i, the constraints $a_{i,j} = 1$ if data centre j is the nearest one to base station i, and $a_{i,j} = 0$ otherwise.
- the third one is the minimum-cost centralized solution in which only one data centre is used. This solution is obtained by adding to problem (CAPA-PL) the following constraints:

$$\begin{aligned} \sum_{t} \sum_{(k',i') \neq (1,1)} a_{i',j}^{k',t} &= \tau \left[S|\mathcal{B}| - 1 \right] a_{1,j}^{1,1}, \quad \forall j, \\ \sum_{j} a_{1,j}^{1,1} &= 1. \end{aligned}$$

Note that for low values of the maximum latency T, the problem might become infeasible with these additional constraints.

Using the MILP solver Gurobi [12], we computed the cost of each of the above solution for $\delta_1 = 10^{-2}$ and $\delta_1 = 10^{-4}$, and for different values of the maximum latency T_1 between 69 ms and 300 ms. The results are reported in Fig. 6. As expected, the fully distributed solution is optimal for low values of T, whereas the minimum-cost centralized solution is either infeasible or very expensive. For instance, for $\delta = 0.01$ and $T = 80$ ms, the centralized solution is about 17% more expensive than the optimal one. However, as T increases, the centralized solution quickly becomes the optimal one, whereas the fully distributed one is significantly more expansive. For $\delta = 0.01$, the additional cost is +91% for $T = 300$ ms, but it is already +48% for $T = 100$ ms.

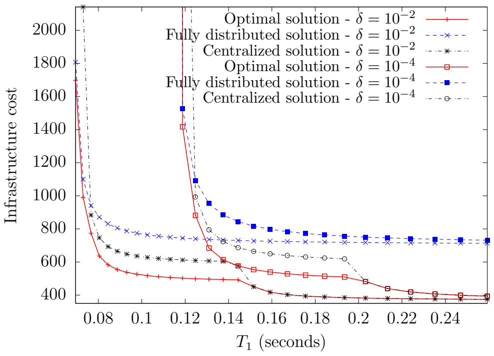

Figure 6: Cost of the FoG infrastructure as a function of the latency requirement of real-time jobs for the cost function given in (17).

Fig. 7 shows the probability that the end-to-end processing delay of jobs be greater than the allowed value as a function of the time of the day when $T = 100$ ms and $\delta = 0.01$. In this case, the optimal solution sends all real-time jobs to the data centre in Corbeil-Essonnes. Note that these probabilities fluctuate over time but never exceed δ.

Let us now evaluate the effect of economies of scale on the structure of the optimal solution. We now assume that the goal is to minimize

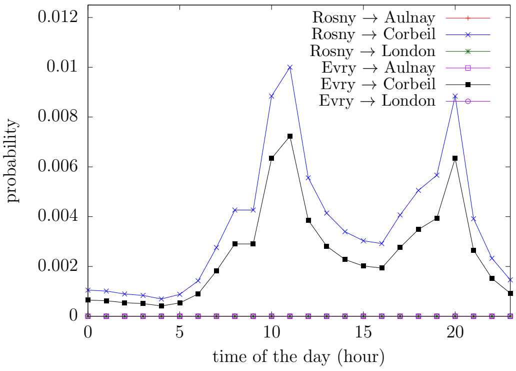

Figure 7: Probability that the end-to-end delay in the optimal solution be greater than the maximum allowed value as a function of time.

$$5 \times (u_1+u_2)+\log(1+c_1)+\log(1+c_2)+\frac{3}{4} \times \log(1+c_3). \qquad (18)$$

The results obtained for this cost function are reported in Fig. 8. Note that the drop in the cost of the optimal solution at $T_1 = 190$ ms correspond to the point where the public cloud in London can host all the traffic. We remark that, as expected, the minimum-cost centralized solution is optimal whenever it is feasible. For $\delta = 0.01$ and $T = 80$ ms (resp. $T = 300$ ms), the cost of the fully distributed solution is 76% (resp. 323%) greater than that of the optimal one.

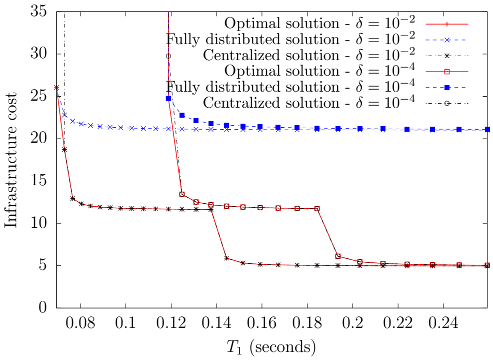

Figure 8: Cost of the FoG infrastructure as a function of the latency requirement of real-time jobs for the cost function given in (18).

5.2 Larger number of base stations

We now build upon the previous scenario to design scenarios in which there is a larger number of base stations. We consider 29 base stations and 3 data centres, which are located as shown in Fig. 9. Note that for convenience, the data centre located in London is not shown in Fig. 9. Most of the base stations are in the area around Paris, but some of them are located a bit farther.

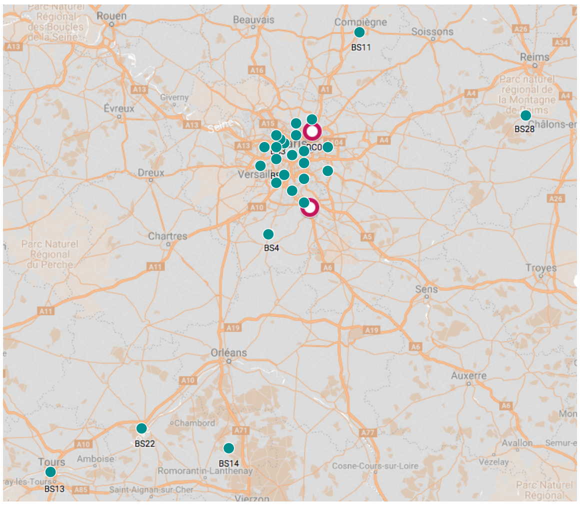

Figure 9: Locations of data centers and base stations for the third scenario.

We consider the same classes of jobs than in the previous scenario (cf. Table 1) with $T_1 = 105$ ms and $\delta_1 = 0.01$. For values of T_1 below 105 ms, the delay requirement of real-time requests sent by the base station located in Tours cannot be met. To generate random problem instances with realistic traffic patterns, we have used a spatio-temporal model inspired from [22], in which a sinusoid superposition model is used to capture the temporal traffic variation, whereas a normal distribution is used for spatial traffic modelling at each epoch.

In a first scenario, we consider only the 5 first base stations, then in a second scenario we consider only the 10 first base stations, etc., until all 29 base stations are included in the sixth and last scenario. We have randomly generated 16 problem instances for each scenario. We have limited the total time expended by the solver gurobi to 5 minutes per problem instance. To avoid spending too much time in proving optimality, we also have set the relative gap of Gurobi to 2%, so that it terminates (with an optimal result) when the gap between the lower and upper objective bounds is less than 0.02 times the the upper bound.

As before, we first consider the case when the infrastructure cost is linear in the capacities. Fig. 10 shows the minimum, maximum and average values of the relative gap in percent between the costs of the optimal solution and the other solutions as a function of the number of base stations. Surprisingly, we observe that the fully distributed and the centralized solutions are always around 30% more expensive than the optimal one. In most cases, the time limit of 5 mn was reached by the solver. We however believe that the optimal solution was found in a few seconds in most cases by Gurobi, which was then not able to prove the optimality of the solution within the allocated timeframe.

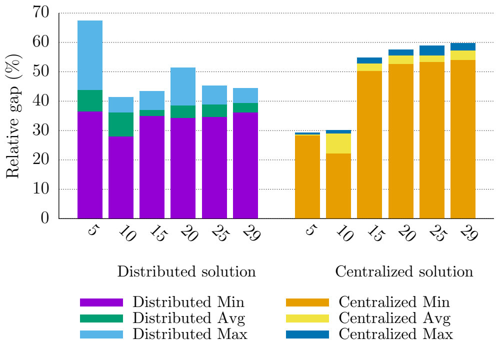

Figure 10: Relative gap in % between the costs of the optimal solution and the other solutions as a function of the number of base stations for a linear cost function.

The results obtained for a logarithmic cost function are reported in Fig. 11. The time limit of 300 seconds was always reached, which means that there is no optimality certificate for the solution obtained. As expected, the relative gap between the optimal cost and the cost of the centralized solution is less important than for a linear objective function. However, significant differences are still observed in Fig. 11 for more than 15 base stations.

6 Related Work

Most of the works on the optimal design of Fog Computing infrastructures focus on the optimal placement of cloudlets and on traffic offloading to the cloud [8] [21] [10] [17] [11] [24] [13] [23]. For instance, the authors in [23] investigate how to optimally select K mobile access points in which to install a cloudlet, assuming that each cloudlet has a fixed capacity and that a fraction of the traffic is offloaded to the cloud. As another example, [21] studies the optimal offloading strategy of mobile devices to fixed-capacity cloudlets with the objective of minimizing the latency.

A number of studies have however been devoted to the capacity planning

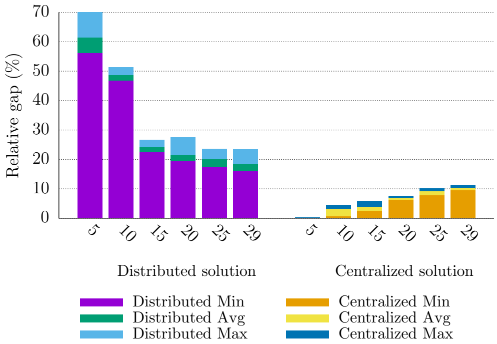

Figure 11: Relative gap in % between the costs of the optimal solution and the other solutions as a function of the number of base stations for the logarithmic cost function.

of Fog infrastructures. In [20] , the authors formulate a mixed integer non-linear programming for the placement and capacity planning of cloudlets, assuming as input a number of potential locations. The objective is to minimize cloudlet installation as well as networking cost. The authors model cloudlets as $M/M/1$ queues, which has the drawback that the processing time can become arbitrary low when the capacity is large. This assumes that the capacity of all the cloudlet servers can be pooled to serve a job. In practice, a job will be allocated a small fraction of the whole capacity and cannot use the capacity of the other servers even if they are idle. Further, this model will only give a lower bound on the capacity, which may not be a good approach for jobs requiring QoS guarantees. In addition, in contrast to the present work, the work in [20] considers only one class of jobs and does not take into account the temporal variations of traffic demands.

Another relevant work is [14]. Given the total capacity, the problem addressed by the authors amounts to distributing it among cloudlets and the cloud. Cloudlets are modeled as discrete-time fluid systems. In contrast to the present work, there is no routing decision from base stations to cloudlets. At each time instant, each cloudlet receives a random amount of traffic and, if the traffic exceeds the capacity of the cloudlet, the excess traffic is routed to the cloud. In this model, there is no infrastructure cost, and the goal is to optimize the quality of service of network flows. The authors discuss variations of the objective function based on the delay or loss probability.

Some other works have addressed the capacity planning problem, but without any queuing model. For instance, the authors in [23] suggest to use profiling and benchmarking tools to determine the resource requirements of applications from their workload. They assume that latency-sensitive applications are always executed in cloudlets, whereas other applications are executed in the cloud. In order to minimize the capacity required in each cloudlet, they solve a Knapsack problem.

7 Conclusions

We have shown that the optimal capacity-planning of micro data centres used in Fog Computing can be formulated as MILP problem, which can be solved efficiently even for large-size problem instances. Numerical results show that significant cost savings can be obtained with respect to the solution in which user requests are always processed in the nearest data centre, and with respect to the minimum-cost centralized solution.

As future work, we plan to extend our approach to situations in which the capacity of individual compute servers are shared among several classes of jobs, using for instance a strict priority mechanism or another more advanced resource sharing mechanism. One challenging problem is to dimension the system when the service times of jobs are not exponentially distributed. Since there are no known formulas for the distribution of the processing time in the general case, we intend to look at analytical approximations that can help dimension the system. Further, we also plan to consider more advanced load-balancing policies than Bernoulli routing for the distribution of jobs inside a data centre, such as for instance policies based on Power of Two Choices or Join the Shortest Queue.

8 TCP transfer times

In this section, we study the transfert time of a file with TCP as a function of the RTT and the file size. To this end, we use a simple deterministic model which was proposed in [5]. The model assumes an ideal environment in which no losses occur and round-trip times are constant. The model can thus be used to estimate lower bounds on the transfer time in most standard TCP implementations.

Let N be the total number of packets transmitted, T be the total transfer time (ms) and RTT be the round-trip time between the sender and the receiver (ms). Let also W_s be the TCP slow-start threshold and W_{max} be the maximum window size, both being measured in packets. Using simple arguments, it is then possible to show that the number of packets transmitted in the TCP slow-start phase is

$$N_{ss} = 2\,W_s - 1,$$

whereas the number of packets transmitted in the congestion-avoidance phase is

$$N_{ca} = (W_{max} - W_s - 1)\,\frac{W_{max} + W_s}{2}.$$

It follows that for short files containing $N \leq N_{ss}$ packets, the transfer time is given by

$$T = (\lceil \log_2(N) \rceil + 1)\ RTT,$$

where $\lceil x \rceil$ denotes the smallest integer greater than x. Files with more than N_{ss} packets but with less than $N_{ss} + N_{ca}$ packets are completely transmitted before the end of the congestion avoidance phase. In this case, the transfer time is given by

$$\begin{aligned} T \;=\; & \{\log_2(W_s) + 1 \\ & + \left\lceil \left(\sqrt{(2W_s + 1)^2 + 8(N - 2W_s + 1)} \right.\right. \\ & \left.\left. -(2W_s + 1) \right) /2 \right\rceil \}\ RTT. \end{aligned}$$

Finally, for files of size strictly greater than $N_{ss} + N_{ca}$ packets, $N - (N_{ss} + N_{ca})$ packets are transmitted in the steady-state phase. In this phase W_{max} packets are transmitted per RTT. A slightly more complex formula can be derived to compute the transfer time of such files (see equation (4) in [5]).

Note that, independently of the file size, the transfer time is linear in RTT. Fig. 12 shows how the transfer time evolves as a function of the file size for two different values of the RTT, 10 ms and 20 ms. It was assumed that $W_s = 16$ packets and $W_{max} = 64$ packets, and that the packet size is 1500 Bytes. Interestingly, we note that when the RTT is 20 ms, the transfer time of only 3 packets (4.5 kB) is already 60 ms. When the RTT is 10 ms, a file of size 13.5 kB (9 packets) is transferred in 50 ms.

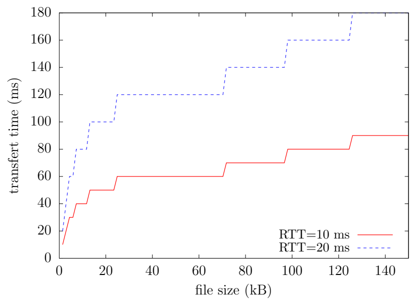

Figure 12: Transfert time with TCP as a function of the file size.

9 Proof of Lemma 4.2

Inequality (13) can equivalently be written as follows

$$c_j \geq \frac{y_j^t}{\mu - \kappa u_j^t / \left[T - \max_i \left(\ell_{i,j} a_{i,j}^t\right)\right]}. \tag{19}$$

Indeed, if $u_j^t = 1$, the RHS of (13) and (19) are obviously equal. If on the contrary $u_j^t = 0$, then it follows from (3) and (5) that $x_{i,j}^t = 0$ for all i and therefore that $y_j^t = 0$, which implies that the equality between the RHS of (13) and (19) holds.

We shall now make use of the following result.

Lemma 9.1. *For any values of the routing variables satisfying* (1)-(7)*, it holds that*

$$\frac{\kappa}{T - \max_i \left(\ell_{i,j} a_{i,j}^t\right)} u_j^t = \max_i \left(\frac{\kappa}{T - \ell_{i,j}} a_{i,j}^t\right) \tag{20}$$

Proof. Noting that the function $x \to \frac{\kappa}{T-x}$ is strictly increasing over $[0, T)$, we obtain

$$\frac{\kappa}{T - \max_i \left(\ell_{i,j} a_{i,j}^T\right)} u_j^t = \max_i \left(\frac{\kappa}{T - \ell_{i,j} a_{i,j}^t}\right) u_j^t.$$

If $u_j^t = 0$, then from constraint (3) we have that $a_{i,j}^t = 0$ for all i, and hence equality (20) is satisfied. If on the contrary $u_j^t = 1$, then constraint (4) implies that there exists k such that $a_{k,j}^t = 1$, and hence that

$$\max_i \left(\frac{\kappa}{T - \ell_{i,j} a_{i,j}^t}\right) \geq \frac{\kappa}{T - \ell_{k,j}} > \frac{\kappa}{T}.$$

Since $\frac{\kappa}{T - \ell_{i,j} a_{i,j}^t} = \frac{\kappa}{T - \ell_{i,j}} a_{i,j}^t$ when $a_{i,j}^t = 1$, and $\frac{\kappa}{T - \ell_{i,j} a_{i,j}^t} = \frac{\kappa}{T}$ when $a_{i,j}^t = 0$, we conclude that equality (20) is also satisfied when $u_j^t = 1$. □

From Lemma 9.1, it follows that

$$\begin{aligned} c_j &\geq \frac{y_j^t}{\mu - \max_i \left(d_{i,j} a_{i,j}^t\right)}, && (21) \\ &= \max_i \left\{ \frac{y_j^t}{\mu - d_{i,j} a_{i,j}^t} \right\}, && (22) \\ &= \max_i \left\{ \frac{y_j^{k,t}}{\mu_k - d_{i,j}^k} a_{i,j}^{k,t} \right\}, && (23) \end{aligned}$$

where the equality between (21) and (22) follows from the fact that the function $z \to \frac{y_j^t}{\mu - z}$ is strictly increasing over $[0, \mu)$. The last equality is proved by considering two different cases:

- If $a_{i,j}^t = 0$ for all i, then it follows from (5) that $x_{i,j}^t = 0$ for all i and therefore that $y_j^t = 0$, which implies that the equality between (22) and (23) holds.
- if $a_{i,j}^t = 1$ for some i, then the maximum value in (22) is obtained for some k such that $a_{k,j}^t = 1$, and this value is equal to the value obtained in (23), which proves that the equality is also valid in this case.

Finally, we conclude the proof by observing that the inequality (22) has to be satisfied for all values of t, which yields (14).

Acknowledgment

We gratefully acknowledge the funding received from Continental Digital Services France.

References

[1] ETSI TR 102 638:Vehicular Communications; Basic Set of Applications; Definitions. Technical report, ETSI Std. ETSI ITS Specification TR 102 638 version 1.1.1, June 2009.

[2] Announcing the ‘edge computing’ concept and the ‘edge accelerated web platform’ prototype to improve response time of cloud. NTT Press release, 2014.

[3] Fog computing. Cisco Technology Radar Trends, 2014.

[4] 3GPP. TR 36.881: Study on latency reduction techniques for LTE. Technical report, 2016.

[5] H. El Aarag and M. Bassiouni. Performance evaluation of tcp connections in ideal and non-ideal network environments. *Computer Communications*, 24:1769–1779, 2001.

[6] F. Bonomi. Cloud and fog computing: Trade-offs and applications. In *International Symposium of Computer Architecture*, 2011.

[7] F. Bonomi, R. Milito, J. Zhu, and S. Addepalli. Fog computing and its role in the internet of things. In *ACM SIGCOMM International Conference on Mobile Cloud Computing*, 2012.

[8] Alberto Ceselli, Marco Premoli, and Stefano Secci. Cloudlet network design optimization. In Rahim Kacimi and Zoubir Mammeri, editors, *Networking*, pages 1–9. IEEE, 2015.

[9] C. D'Ambrosio, A. Lodi, and S. Martello. Piecewise linear approximation of functions of two variables in milp models. *Operations Research Letters*, 38:39–46, 2010.

[10] Qiang Fan and Nirwan Ansari. Cost aware cloudlet placement for big data processing at the edge. In *ICC*, pages 1–6. IEEE, 2017.

[11] Erol Gelenbe, Ricardo Lent, and Markos Douratsos. Choosing a local or remote cloud. In *Second Symposium on Network Cloud Computing and Applications, NCCA 2012, London, United Kingdom, December 3-4, 2012*, pages 25–30, 2012.

[12] LLC Gurobi Optimization. Gurobi optimizer reference manual, 2018.

[13] M. Jia, J. Cao, and W. Liang. Optimal cloudlet placement and user to cloudlet allocation in wireless metropolitan area networks. *IEEE Transactions on Cloud Computing*, 5(4):725–737, Oct.-Dec. 2017.

[14] Abbas Kiani, Nirwan Ansari, and Abdallah Khreishah. Hierarchical capacity provisioning for fog computing. *CoRR*, abs/1807.01093, 2018.

[15] Leonard Kleinrock. *Theory, Volume 1, Queueing Systems.* Wiley-Interscience, New York, NY, USA, 1975.

[16] A. Kurian. Latency analysis and reduction in a 4g network. 2018.

[17] Amardeep Mehta, William Tärneberg, Cristian Klein, Johan Tordsson, Maria Kihl, and Erik Elmroth. How beneficial are intermediate layer data centers in mobile edge networks? In *Workshops on Fog and Mobile Edge Computing at Foundations and Applications of Self* Systems*, pages 222–229. IEEE–Institute of Electrical and Electronics Engineers Inc., 2016.

[18] E. Miluzzo. I'm cloud 2.0, and i'm not just a data center. *IEEE Internet Computing*, 18(3):73–77, May 2014.

[19] C. Mims. Forget 'the cloud': 'the fog' is tech's future. *The Wall Street Journal*, 2014.

[20] Sourav Mondal, Goutam Das, and Elaine Wong. A novel cost optimization framework for multi-cloudlet environment over optical access networks. *GLOBECOM 2017 - 2017 IEEE Global Communications Conference*, pages 1–7, 2017.

[21] Xiang Sun and Nirwan Ansari. Latency aware workload offloading in the cloudlet network. *IEEE Communications Letters*, 21:1481–1484, 2017.

[22] S. Wang, X. Zhang, J. Zhang, J. Feng, W. Wang, and K. Xin. An approach for spatial-temporal traffic modeling in mobile cellular networks. In *2015 27th International Teletraffic Congress*, pages 203–209, Sept 2015.

[23] Yu Xiao, Marius Noreikis, and Antti Ylä-Jääski. Qos-oriented capacity planning for edge computing. In *IEEE International Conference on Communications, ICC 2017, Paris, France, May 21-25, 2017*, pages 1–6, 2017.

[24] Zichuan Xu, Weifa Liang, Wenzheng Xu, Mike Jia, and Song Guo. Efficient algorithms for capacitated cloudlet placements. *IEEE Transactions on Parallel and Distributed Systems*, 27:2866–2880, 2016.

[25] J. Zhu, D. S. Chan, M. S. Prabhu, P. Natarajan, H. Hu, and F. Bonomi. Improving web sites performance using edge servers. In *IEEE International Symposium on Service-Oriented System Engineering*, 2013.