The edge cloud: A holistic view of communication, computation and caching

Sergio Barbarossa[,a,∗], Stefania Sardellitti[∗], Elena Ceci[∗] and Mattia Merluzzi[∗]

[∗]*Sapienza University of Rome, Dept. of Information Engineering, Electronics, and Telecommunications, via Eudossiana 18, 00184, Rome, Italy.*

[a]*Corresponding: sergio.barbarossa@uniroma1.it*

ABSTRACT

The evolution of communication networks shows a clear shift of focus from just improving the communications aspects to enabling new important services, from Industry 4.0 to automated driving, virtual/augmented reality, Internet of Things (IoT), and so on. This trend is evident in the roadmap planned for the deployment of the fifth generation (5G) communication networks. This ambitious goal requires a paradigm shift towards a vision that looks at communication, computation and caching ($3C$) resources as three components of a single holistic system. The further step is to bring these $3C$ resources closer to the mobile user, at the *edge* of the network, to enable very low latency and high reliability services. The scope of this chapter is to show that signal processing techniques can play a key role in this new vision. In particular, we motivate the joint optimization of $3C$ resources. Then we show how graph-based representations can play a key role in building effective learning methods and devising innovative resource allocation techniques.

Keywords: 5G networks, wireless communications, graph-based learning

0.1 INTRODUCTION

The major goal of next generation (5G) communication networks is to build a communication infrastructure that will enable new business opportunities in diverse sectors, or *verticals*, such as automated driving, e-health, virtual/augmented reality, Internet of Things (IoT), smart grids, and so on [1], [2]. These services have very different specifications and requirements

2   Chapter Title

in terms of latency, reliability, data rate, number of connected devices, and so on. Thinking of enabling such diverse services using a common communication platform might then look like a crazy idea. But, in reality, if the system is properly designed, reusing a common infrastructure for different purposes might induce a significant economic advantage. The key idea for making this possible is to use *virtualization* [3] and implement *network slicing* [4]. Through virtualization, many network functionalities are implemented in software through virtual machines that can be instantiated and moved upon request [5]. Building on virtualization, network slicing partitions a *physical* network into multiple *virtual* networks, each matched to its specific requirements and constraints, thus enabling operators to provide networks on an as-a-service basis, while meeting a wide range of use cases in parallel.

This new reality, sometimes called fourth industrial revolution, can be realized by a new architecture able to meet advanced requirements, especially in terms of latency (below 5 ms), reliability (around 0.99999), coverage (up to 100 devices/m^2), and data rate (more then 10 Gbps). At the physical layer, 5G builds on a significant increase of system capacity by incorporating massive MIMO techniques, dense deployment of radio access points, and wider bandwidth. All these strategies are facilitated by the introduction of millimeter wave (mmWave) communications [6], [7], [8]: mmWaves make possible the reduction of the antenna size, thus enabling the use of array with many elements, as required in massive MIMO; dense deployment is also facilitated because mmWaves give rise to a stronger intercell attenuation; finally, increasing the carrier frequency facilitates the usage of wider bandwidths. However, the significant improvement achievable at the physical layer could be still insufficient to meet the challenging and diverse requirements of very low latency and ultra reliability. A further improvement comes from a paradigm shift that puts applications at the center of the system design. Network Function Virtualization (NFV) and Multi-access Edge Computing (MEC) [9] are the key tools of this application-centric networking. In particular, MEC plays the key role of bringing cloud-computing resources at the edge of the network, within the Radio Access Network (RAN), in close proximity to mobile subscribers [9], [10]. MEC is particularly effective to deliver context-aware services or to enable computation offloading from resource-poor mobile devices to fixed servers or to perform intelligent cache pre-fetching, based on local learning of the most popular contents across space and time.

Given this perspective, the goal of this chapter is to show that graph-based methods can play a significant role in optimizing resource allocation or deriving new learning mechanisms. The organization of this chapter is the following. In Section 0.2 we present the edge-cloud architecture and we motivate the holistic approach that looks at $3C$ resources as a common pool of resources to be handled jointly with the goal of achieving, on the user side, a satisfactory quality of experience and, on the network side, a balanced and efficient use
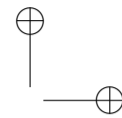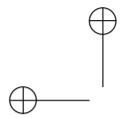
of resources. Then, in Section 0.3, we will focus on the joint optimization of computation and communication resources, with specific attention to computation offloading in the edge-cloud. In Section 0.4, we will concentrate on the joint optimization of caching ad communication. Differently from storage, which is fundamentally *static*, caching is inherently *dynamic*, so that cache memories are pre-fetched when and where needed, and then released. In both cases of joint optimization, the goal is to bring resources, either computation (virtual machines) or cache, as close as possible to the end user, to enable truly low latency and low energy consumption services. After presenting this holistic view, we will move in Section 0.5 to present some learning mechanisms based on graph signal processing. In particular, we show how to reconstruct the radio environment map (REM), which enables a cognitive usage of the radio resources. Then, building again on graph representations, in Section 0.6, we show how to achieve an optimal resource allocation across a network while being robust to link failures. The proposed approach is based on a small perturbation analysis of network topologies affected by sporadic edge failures. Finally, in Section 0.7 we draw some conclusions and suggest some possible further developments.

0.2 HOLISTIC VIEW OF COMMUNICATION, COMPUTATION AND CACHING

The new infrastructure provided by next communication networks can be seen as a truly distributed and pervasive computer that provides very different services to mobile users with sufficiently good quality of experience. The physical resources composing this pervasive computer are cache memories, computing machines, and communication channels. The system should serve the end user, either a mobile subscriber or a car or the component of a production process with, ideally, zero latency, which means an end-to-end latency smaller than the user perception capability or than the maximum value ensuring proper control, like breaking time in automated driving. To enable this vision, at the physical layer, the network will support a much higher system (or area) capacity (bits/sec/km^2). In 5G systems, a $1,000$-fold increase of system capacity is planned, exploiting mmWave communications, massive MIMO, and dense deployment of access points. However, in spite of this enormous improvement in system capacity, the zero-latency ideal could still be far to be obtained because it is very complicated, if not impossible, to control latency over a wide area network. For this reason, the next step is to bring computation and cache resources as close as possible to the end user, where proximity is actually measured in terms of service time. This creates a new eco-system, called edge-cloud, whose architecture is sketched in Fig. 0.1. In this system, within a macro-cell served by one base station, we have multiple millimeter-wave
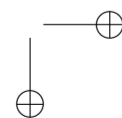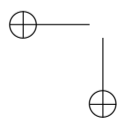
4 Chapter Title

access points (AP), covering much smaller areas. Each AP is endowed with computation and caching capabilities, to enable mobile users to get proximity access to cloud functionalities. This makes possible to provide cloud services with very low latency and high data rate, while at the same time keeping data traffic and computation as local as possible. Of course, the computing and caching capabilities of local MEC servers are significantly lower than a typical cloud, but they also serve a limited number of requests and, whenever their resources are insufficient, they may interact with nearby MEC servers, under the supervision of a MEC orchestrator. In this system, mobile applications
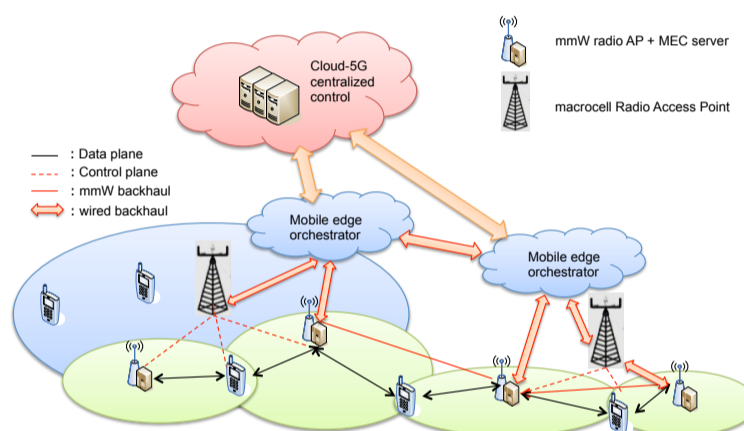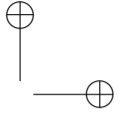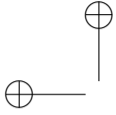

Figure 0.1 Edge-cloud architecture.

are handled by *virtual machines* (or containers) instantiated at the edge of the network, close to the end user. The edge is either the ensemble of network access points, as in Multi-access Edge Computing (MEC) [11], or it might even include the mobile terminals as well, as in *fog computing* [12].

Similarly, contents move dynamically when and where it is more convenient to have them. Caching can in fact be seen as a *non-causal* communication, where content move before they are actually requested, to minimize the downloading time. In this framework, it makes sense to allocate $3C$ resources *jointly*, with the objective of guaranteeing some ultimate user quality of experience.

Assuming such a holistic view perspective, the first important question is why using a *common* platform, call it 5G or the generations to come next, to accommodate services having so different requirements, like IoT, virtual reality or automated driving. This is indeed one of the main challenges faced by 5G systems. The approach proposed in the 5G roadmap is *network slicing* [5]. A network slice is a virtual network that is implemented on top of a physical network in a way that creates the illusion to the slice tenant of operating its

own dedicated physical network.

Optimizing network slicing is a first important application of graph-based representations, at a high level. In fact, a mathematical formulation of network slicing has been recently proposed in [13], where the communication network is represented as a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V}$ is the set of nodes and $\mathcal{E}$ is the set of directed links. There is a subset of function nodes, enabled with NFV functionalities, that can provide a service function f. In general, there are K flows, each requesting a distinct service. The requirement of each service k is represented as a service function chain $\mathcal{F}(k)$ consisting of a set of functions that have to be performed in the predefined order throughout the network. Zhang et al. in [13] formulated the slicing problem as the optimal allocation of service functions across the NFV-enabled nodes, while minimizing the total flow in the network. The problem is a mixed binary linear program, which is NP-hard. Nevertheless, the authors of [13] proved that the problem can be relaxed with performance guarantees. This is indeed a very interesting application of a graph-theoretic formulation of a very high-level problem.

In the following two sections, we will focus on the joint optimization of pairs of $3C$ resources, namely communication and computation in Section 0.3 and communication and caching in Section 0.4.

0.3 JOINT OPTIMIZATION OF COMMUNICATION AND COMPUTATION

Smartphones have really exploded in their usage and capabilities, placing significant demand upon battery usage. Unfortunately, advancements in battery technology have not kept pace with the demands of users and their smartphones. One approach to overcome the battery energy limitations is to offload computations from mobile devices to fixed devices. Computation offloading may be convenient for the following reasons [14], [15]: i) to save energy and then prolong the battery lifetime of hand-held devices; ii) to enable simple devices, like inexpensive sensors, to run sophisticated applications; iii) to reduce latency. From a user perspective, one of the parameters mostly affecting the quality of experience is the end-to-end (E2E) latency, i.e. the time necessary to get the result of running an application. In case of offloading, this latency includes: i) the time to send bits from the mobile device to the fixed server to enable the program; ii) the time to run the application remotely; iii) the time to get the result back. It is precisely this E2E latency that couples communication and computation resources and then motivates the joint allocation of these resources. We recall now the approach proposed in [16] and later expanded in [14] and [17].

We first consider the case where multiple users are served by a single AP/MEC pair. Then, we will move to the more challenging case where
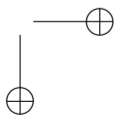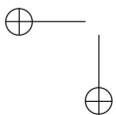
6 Chapter Title

multiple users are served by multiple AP's and MEC servers. In the first case, the assignment of each UE to a pair of AP and MEC is supposed to be given; in the second case, the assignment is part of the optimization problem. In both cases, for economical reasons associated to promoting their capillary deployment, the computational capabilities of MEC servers are enormously smaller than a typical cloud. This implies that the number of cores per server is very limited or, in other words, that the available cores in a MEC server must operate in a multi-tasking mode to accommodate the requests of multiple users. This means that a server running K applications for as many mobile users will allocate a certain percentage β_k of its CPU time to the users that are being served concurrently. If F_S denotes the number of CPU cycles/sec that the server can run, the percentage of CPU cycles/sec assigned to the k-th user is then $f_k = \beta_k F_S$.

Multiple users served by a single AP/MEC pair

We start by considering K user equipments (UE) assigned to a single AP and a single MEC. The decision to offload a computation from the mobile device to the MEC server depends on the characteristics of the application to be offloaded. Not all applications are equally amenable to offloading. The decision should take into account all sources of energy consumption in a smartphone, like display, network, CPU, GPS, camera, and so on. Profiling energy consumption of applications running on smartphones, rather than on a general purpose computer, is not an easy task because of *asynchronous power behavior*, where the effect on a component's power state due to a program entity lasts beyond the end of that program entity [18]. The signal processing community could provide a significant contribution to this research field by optimizing app developments taking into account the associated energy profiling for a class of smartphone operating systems, e.g. OS, Android, and so on, and a class of applications. In this chapter, we do not dig into these aspects. We rather concentrate on the joint optimization of radio and computational resources associated to computation offloading, in a multiuser context. From this point of view, we simplify the classification of applications by identifying a few most significant parameters, as relevant for computation offloading. For each user k, we consider: i) the number b_k of bits to be transmitted from the mobile user to the server to transfer the program execution; ii) the number of CPU cycles w_k necessary to run the application to be offloaded. We denote by L_k the E2E latency requested from UE k. The overall latency T_k experienced by the k-th UE for offloading an application is the sum of three terms: i) the time $T_k^{\texttt{tx}}$ necessary to transmit all bits to the server to enable the transfer of program execution; ii) the time $T_k^{\texttt{exe}}$ for the server to run the application; iii) the time $T_k^{\texttt{rx}}$ to get the result back to the UE. In formulas,

$$T_k = T_k^{\texttt{tx}} + T_k^{\texttt{exe}} + T_k^{\texttt{rx}}. \tag{0.1}$$

This equation, in its simplicity, shows that enforcing an E2E latency constraint induces a coupling between communication and computation resources.

From a user-centric perspective, the goal might either be to minimize the E2E latency, under a maximum transmit power constraint or, by duality, to minimize the transmit power necessary to guarantee a desired latency. We follow this latter approach, but clearly the two strategies can be interchanged. Let us now express the single contributions in (0.1) in terms of the parameters to be optimized.

The first contribution is the time $T_k^{\mathtt{tx}}$ to transmit b_k bits from the UE to the AP:

$$T_k^{\mathtt{tx}}(p_k) = \frac{c_k}{r_k(p_k)} \tag{0.2}$$

where $c_k = b_k/B$, B is the bandwidth and $r_k(p_k)$ is the spectral efficiency over the channel between UE and AP, which is equal to

$$r_k(p_k) = \log_2 \ (1 + \alpha_k p_k) \tag{0.3}$$

where p_k is the transmit power of UE k; $\alpha_k = |h_k|^2/(d_k^\gamma \sigma_n^2)$ is a an equivalent channel coefficient that incorporates the channel coefficient h_k, the noise variance σ_n^2, the distance d_k between UE and AP, and the channel exponent factor γ. The second contribution in (0.1) is the execution time at the server, which is equal to $T_k^{\mathtt{exe}} = w_k/f_k$. From the user perspective, the third term in (0.1) does not imply a transmit power, but only the energy to process the received data. This term is typically much smaller than the first term and in the following derivations we will assume it to be a fixed term incorporated in the overall latency.

We are now ready to formulate the computation offloading optimization problem in terms of the transmit powers p_k and the CPU percentages f_k, $k = 1, \ldots, K$:

$$
\begin{aligned}
\min_{\boldsymbol{p}, \boldsymbol{f_s}} \quad & \sum_{k=1}^{K} p_k, && [\mathbf{P.1}] \\
\text{s.t.} \quad & \frac{c_k}{\log_2 (1 + p_k \alpha_k)} + \frac{w_k}{f_k} \leq L_k, \ k = 1, \ldots, K \\
& 0 < p_k \leq P_T, \quad f_k > 0, \quad k = 1, \ldots, K \\
& \sum_{k=1}^{K} f_k \leq F_S
\end{aligned}
\tag{0.4}
$$

where $\mathbf{p} = (p_1, \ldots, p_K)$ and $\mathbf{f_s} = (f_1, \ldots, f_K)$.

This is a convex problem that can be easily solved. In particular, the

8   Chapter Title

optimal computational rates can be expressed in closed form as [19]:

$$f_k = \frac{\sqrt{w_k\, \eta_k}}{\sum_{k=1}^{K} \sqrt{w_k \eta_k}} F_S, \qquad (0.5)$$

where η_k are coefficients that depend on the channel coefficients. This simple formula shows how the allocation of computational resources depends not only on computational aspects, but also on the channel state. Note also that the above formula contrasts with the proportional allocation of computational rates that would have been performed in a conventional system, i.e.

$$f_k = \frac{w_k}{\sum_{k=1}^{K} w_k} F_S. \qquad (0.6)$$

A further substantial improvement to computation offloading comes from the introduction of mmWave links. Merging MEC with an underlying mmWave physical layer creates indeed a unique opportunity to bring IT services at the mobile user with very low latency and very high data rate. This merge is indeed one of the main objectives of the joint Europe/Japan H2020 Project called 5G-MiEdge (Millimeter-wave Edge Cloud as an Enabler for 5G Ecosystem) [20]. The challenge coming from the use of mmWave links is that they are more prone to blocking events [21], which may jeopardize the benefits of computation offloading. A possible way to counteract blocking events in a MEC system using mmWave links was proposed in [22], [19].

Multiple users served by multiple AP's and multiple MEC servers

Let us consider now a more complex scenario, where multiple users may get radio access through multiple AP's and multiple MEC's. Besides resource allocation, our goal now is to find also the optimal association between UE's, AP's and MEC servers. We consider a system composed of N_b small cell access points, N_c MEC servers, and K mobile UE's. Within the edge-cloud scenario depicted in Fig. 0.1, the association of a mobile user to an access point does not necessarily follow the same principles of current systems, where a mobile user gets access to the base station with the largest signal-to-noise ratio. In the edge-cloud scenario depicted in Fig. 0.1, the association of a UE to a pair of AP and MEC server depends not only on radio channel parameters, but also on the availability of computational resources at the MEC server. Furthermore, a UE can get radio access from a certain AP, but its application can run elsewhere, not necessarily on the nearest MEC, depending on the availability of computational resources. Actually, since the applications run as virtual machines (VM), we can think of migrating these VM's in order to follow the user. The orchestration of MEC servers in order to provide seamless service continuity to mobile users is an item that has been recently included in the standardization activities of ETSI, within the MEC study

group [23]. Migrating VM's is not an easy task, because the instantiation of a VM requires times that are too large with respect to some of the latency requirements foreseen in 5G. This has motivated significant research efforts in investigating light forms of virtual machines, named *containers*, that do not need the instantiation of the whole operating system, but only of a restricted kernel [24].

Here, we do not consider the migration of VM's, but we do consider the possibility of letting a UE get access under one AP, while having its application run in an MEC located elsewhere. In this case, we need to incorporate in the E2E latency the delay along the backhaul link connecting AP and MEC. In particular, we denote by T_{Bnm} the latency between access point n and MEC server m.

Following an approach similar to what we proposed in [25], we generalize now the resource allocation problem by incorporating binary variables $a_{knm} \in \{0, 1\}$ that assume a value $a_{knm} = 1$ if user k gets radio access through AP n to have its application running on MEC server m, and $a_{knm} = 0$ otherwise. For the sake of simplicity, we assume that each user is served by a single base station and a single cloud. Our goal now is to find the optimal assignment rule, together with the optimal transmit powers p_k and the computational rates f_{mk} assigned by MEC server m to UE k. As in the previous section, our goal is to minimize the overall UE power consumption, under a latency constraint.

The resulting optimization problem is:

$$
\begin{aligned}
\min_{\mathbf{p},\mathbf{f},\mathbf{a}} \quad & f(\mathbf{p},\mathbf{a}) \triangleq \sum_{k=1}^{K} \sum_{n=1}^{N_b} \sum_{m=1}^{N_c} p_k a_{knm} \qquad\qquad (\mathcal{P}) \\
\text{s.t.} \quad & \text{i) } g_{knm}(p_k, f_{mk}, a_{knm}) \le L_k, \forall\, k, n, m \\
& \text{ii) } p_k \le P_k, \quad p_k \ge 0, \,\forall\, k \\
& \text{iii) } h_m(\mathbf{f},\mathbf{a}) \triangleq \sum_{k=1}^{K} \sum_{n=1}^{N_b} a_{knm} f_{mk} \le F_m, \,\forall\, m, \, \mathbf{f} \ge \mathbf{0} \\
& \text{iv) } \sum_{n=1}^{N_b} \sum_{m=1}^{N_c} a_{knm} = 1, \, a_{knm} \in \{0, 1\}, \quad \forall\, k, n, m
\end{aligned}
\qquad (0.7)
$$

where $\mathbf{f} := (f_{mk})_{\forall m,k}$, $\mathbf{a} := (a_{knm})_{\forall k,n,m}$, and

$$
g_{knm}(p_k, f_{mk}, a_{knm}) \triangleq a_{knm} \left(\frac{c_k}{r_{kn}(p_k)} + \frac{w_k}{f_{mk}} + T_{Bnm} \right)
$$

with $r_{kn}(p_k) = \log_2 \left(1 + \alpha_{kn} p_k\right)$ denoting the spectral efficiency of UE k accessing AP n and α_{kn} the equivalent channel coefficient between UE k and AP n.

10   Chapter Title

The objective function is the total transmit power consumption from the mobile users. The constraints have the following meaning: i) the overall latency for each user k must be less than the maximum value L_k; ii) the total power spent by each user must be lower than a fixed total power budget P_k; iii) the sum of the computational rates f_{mk} assigned by each server cannot exceed the server computational capability F_m; iv) each mobile user should be served by one AP/MEC pair; this is enforced by imposing $\sum_{n=1}^{N_b} \sum_{m=1}^{N_c} a_{knm} = 1$, for each k, together with $a_{knm} \in \{0,1\}$.

Unfortunately, problem $\mathcal{P}$ is a mixed-binary problem and is, in general, NP-hard. To overcome this difficulty, as we suggested in [26], we relax the binary variables a_{knm} to be real variables in the interval $[0,1]$ and adopt a sub-optimal successive convex approximation strategy [27], [25], able to converge to local optimal solutions. Additionally, to drive the assignment variables a_{knm} to contain only one value equal to one and all others to zero, for each k, we incorporate a further constraint recently suggested in [13]. The penalty method in [13] is based on the fact that the following problem

$$
\begin{aligned}
\min_{\mathbf{a}_k} \quad & \| \mathbf{a}_k + \epsilon \mathbf{1} \|_p^p \triangleq \sum_{n=1}^{N_b} \sum_{m=1}^{N_c} (a_{knm} + \epsilon)^p \\
\text{s.t.} \quad & \| \mathbf{a}_k \|_1 = 1, \\
& a_{knm} \in [0,1], \quad \forall n, m
\end{aligned}
\tag{0.8}
$$

with $\mathbf{a}_k = (a_{knm})_{\forall n,m}$ and $p \in (0,1)$, $\epsilon > 0$, admits an optimal solution that is binary, i.e. only one element is one and all the others are zero. The optimal solution is $c_{\epsilon,k} = (1+\epsilon)^p + (N_b N_c - 1)\epsilon^p$. Therefore, by relaxing the binary variables a_{knm} so that they belong to the following convex set

$$
\mathcal{A} = \{ (\mathbf{a}_k)_{k \in \mathcal{I}} \; : \; a_{knm} \in [0,1], \sum_{n=1}^{N_b} \sum_{m=1}^{N_c} a_{knm} = 1, \forall k, n, m \},
$$

where $\mathcal{I}$ denotes the set of K users, we formulate the following relaxed optimization problem [26]:

$$
\begin{aligned}
\min_{\mathbf{p},\mathbf{f},\mathbf{a}} \quad & f_{P_\sigma}(\mathbf{p}, \mathbf{a}) \triangleq f(\mathbf{p}, \mathbf{a}) + \sigma P_\epsilon(\mathbf{a}) && (\mathcal{P}_\sigma) \\
\text{s.t.} \quad & \text{i) } g_{knm}(p_k, f_{mk}, a_{knm}) \leq L_k, \forall k, n, m \\
& \text{ii) } h_m(\mathbf{f}, \mathbf{a}) \triangleq \sum_{k=1}^{K} \sum_{n=1}^{N_b} a_{knm} f_{mk} \leq F_m, \; \forall m, \; \mathbf{f} \geq \mathbf{0} \\
& \text{iii) } p_k \leq P_k, \quad p_k \geq 0, \forall k \in \mathcal{I}, \; \mathbf{a} \in \mathcal{A}
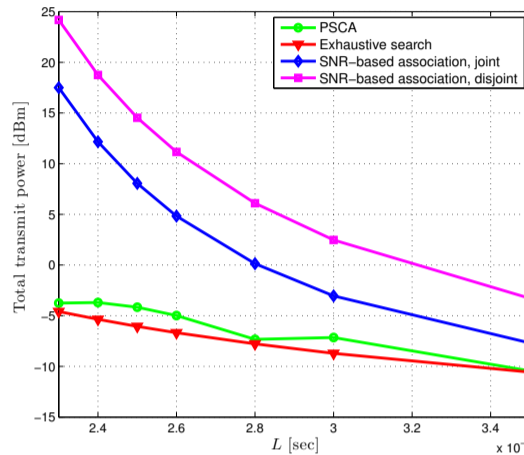\end{aligned}
\tag{0.9}
$$

Figure 0.2   Overall UE transmit power consumption vs. L.

where $\sigma > 0$ is the penalty parameter, and

$$P_\epsilon(\mathbf{a}) \triangleq \sum_{k=1}^{K} \parallel \mathbf{a}_k + \epsilon \mathbf{1} \parallel_p^p - c_{\epsilon,k}. \tag{0.10}$$

It is important to emphasize that this penalty is differentiable with respect to the unknown variables $\mathbf{a}$. Even by relaxing the binary variables $\mathbf{a}$, problem in (0.9) is still non-convex, since the objective function and the constraints i), ii) are non convex. In [26], we proposed a Successive Convex Approximation (SCA) technique, inspired by [27], to devise an efficient iterative penalty SCA approximation algorithm (PSCA) converging to a local optimal solution of (0.9). We omit the details here, but we report some numerical results.

To test the effectiveness of the proposed offloading strategy, in Fig. 0.2 we report the optimal total transmit power consumption vs. the maximum latency L_k. We consider a network composed of $K = 4$ users, a number of base stations equal to the number of clouds, i.e. $N_b = N_c = 2$. The other parameters are set as follows: $F_1 = 2.7 \cdot 10^9$, $F_2 = 6 \cdot 10^8$, $P_k = 2 \cdot 10^{-1}$, $p = 0.025$. From Fig. 0.2, we may observe that the PSCA algorithm provides results very close to the exhaustive search algorithm whose complexity is exponential. Additionally, we consider as a comparison term the SNR-based association method, in both cases where the radio and computational resources are optimized jointly or disjointly. It can be noted that the PSCA algorithm yields considerable power savings compared to methods based on SNR only, since it takes advantage of the optimal assignment of each user to a cloud through the most convenient base station.

0.4 JOINT OPTIMIZATION OF CACHING AND COMMUNICATION

Caching popular contents in storage disks distributed across the network yields significant advantages in terms of reduction of downloading times and limitation of data traffic. Caching can be seen as a *non-causal* communication, where popular contents move throughout the network in the off-peak hours to anticipate the users' requests. Clearly an effective caching strategy builds significantly on the ability to learn and predict users' behaviors. This capability lies at the foundation of *proactive caching* [28] and it motivates the need to merge future networks with big data analytics [29]. An alternative approach to proactive caching based on reinforcement learning to learn file popularity across time and space was recently proposed in [30].

Another important pillar of future networks is Information-Centric Networking (ICN), a relatively novel paradigm concerning the distribution of contents throughout the network in a manner much more efficient than conventional Internet [31]. Different from what happens in the Internet, where contents are retrieved through their address, in ICN, information is retrieved by *named* contents [31]. In the ICN framework, network entities are equipped with storage capabilities and contents move throughout the network to serve the end user in the best possible way [32]. The content placement problem, incorporating number of content copies and their locations in order to minimize a cost function capturing access costs (delay, bandwidth) and/or storage costs, has been formulated as a mixed integer linear program (MILP), shown to be NP-Hard [33]. In the case where global knowledge of user requests and network resources is available, an Integer Linear Programming (ILP) formulation was given in [32], yielding the maximum efficiency gains. In this section we recall and extend the formulation of [32] to incorporate the cost of inefficient storage of non-popular contents. Consider an information network $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{K})$, composed of a set of nodes $\mathcal{V}$, a set of links $\mathcal{E}$, and a set of information objects $\mathcal{K}$, as depicted in Fig. 0.3. A content file can be stored (permanently or temporarily) over the nodes of this graph or travel through its edges. Some contents reside permanently over some repository nodes (e.g., the disks in Fig. 0.3). In all other nodes (e.g., the circles in Fig. 0.3), contents may appear and disappear, according to users' requests and network resource allocation. We suppose, for simplicity, that all contents are subdivided into objects of equal size. Each object is then identified by an index $k \in \mathcal{K}$. Each node is characterized by a storage capability and every edge is characterized by a transport capacity. Time is considered slotted and every slot has a fixed duration $\Delta\tau$. At time slot n, each node $u \in \mathcal{V}$ hosts, as a repository, a set of information objects $K_u[n] \in \mathcal{K}$ and requests, as a consumer, a set of informa-
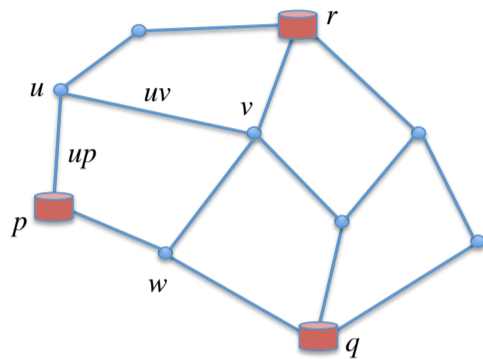
Figure 0.3   Information network.

tion objects $Q_u[n] \in \mathcal{K}$. Let $\mathbf{q}[n] \in \{0,1\}^{|\mathcal{V}||\mathcal{K}|}$ be the request arrival process such that $q_u[k,n] = 1$ if node u requests object k at time n, and $q_u[k,n] = 0$ otherwise.

Given this graph, we define a vertex signal over its nodes and an edge signal over its edges. The vertex signal $s_u[k,n]$ is a binary signal defined as:

$$s_u[k,n] = \begin{cases} 1, & \text{if content } k, \text{ at time } n, \text{ is stored on node } u \\ 0, & \text{otherwise} \end{cases} , u \in \mathcal{V}.$$

The amount of content stored on node u, at time n, is then $S_u[n] := \sum_k s_u[k,n]$. Similarly, we can define an edge signal as a binary signal, defined on each edge, as

$$t_{uv}[k,n] = \begin{cases} 1, & \text{if content } k, \text{ at time } n, \text{ is transported over link } uv \\ 0, & \text{otherwise} \end{cases} , uv \in \mathcal{E}.$$

The amount of content transported over link uv at time n, is then $T_{uv}[n] := \sum_k t_{uv}[k,n]$. Typically, each content may be host on every node and moved whenever useful. The storage and capacity constraints limit the variability of both $S_u[n]$ and $T_{uv}[n]$ as

$$0 \le S_u[n] \le S_u, \ \ 0 \le T_{uv}[n] \le T_{uv}, \tag{0.11}$$

where S_u is the storage capability of node u, whereas T_{uv} is the transport capacity of link uv. The state of the network, at time slot n, is represented by the vector $\mathbf{x}[n] := [\mathbf{s}[n]; \mathbf{t}[n]]$, with $\mathbf{s}[n] := (s_u[k,n])_{\forall u,k}$ and $\mathbf{t}[n] := (t_{uv}[k,n])_{\forall k, uv \in \mathcal{E}}$.

In principle, a content $k \in \mathcal{K}$ may be cached, at any time slot n, in more then one location. However, there is a cost in keeping a content in one place, if is not utilized. The goal of *dynamic caching* is to find the state vector $\mathbf{x}[n]$ that

14   Chapter Title

minimizes an overall cost function that includes the cost for caching and the cost for transportation, under constraints dictated by the storage capability, the transport capacity, and the users' requirements in terms of latency to get access to their desired contents.

The fundamental difference between *caching* and *storage* is that storage is intrinsically *static*, whereas caching is fundamentally *dynamic*. This means that cached contents move throughout the network, appear in some nodes and disappear from others. There are only some *repository* nodes (e.g., nodes p, q, and r in Fig. 0.3) that keep a permanent record or have fast access to a content delivery network. The assumption is that each content is host in at least one repository node.

The basic question about caching is then to decide, dynamically, depending on the users' requests, when and where to place all contents, how to move them, and when to drop contents to save memory. The decision for caching an object k at node u, at time slot n, must result from a trade-off between the cost for storing for a certain amount of time and the cost for transporting the content from its current location to the network access point nearest to the user who requested it.

The cost associated to storing a content k on node u during T consecutive time slots, in the time window $[n' - T + 1, n']$, is

$$E_{st} = \sum_{n=n'-T+1}^{n'} \sum_{k \in \mathcal{K}} \sum_{u \in \mathcal{V}} s_u[k,n] c_u[k], \qquad (0.12)$$

where $c_u[k]$ is the energy cost for keeping content k on node u per unit of time. This unit time cost depends on the popularity of content k in a neighborhood of node u. For instance, we can set

$$c_u[k] = \frac{c_0}{1 + P_u[k]/P_0} \qquad (0.13)$$

where $P_u[k]$ is the popularity of content k at node u and c_0 is the (energy) cost for keeping a content object with zero popularity and P_0 is the popularity level that justifies halving the cost for caching per unit of time, with respect to zero-popularity contents. The introduction of the cost coefficients $c_u[k]$ is what makes the formulation *context-aware*. In fact, the popularity $P_u[k]$ may vary across the network.

The cost associated to content transportation is

$$E_{tr} = \sum_{n=n'-T+1}^{n'} \sum_{k \in \mathcal{K}} \sum_{uv \in \mathcal{E}} t_{uv}[k,n] c_{uv}[k], \qquad (0.14)$$

where $c_{uv}[k]$ is the energy cost for transporting object k over link uv. In general, when user u makes a request of content k, we may associate to that request a maximum delivery time, which we call $D_u[k]$. We also denote by

$\mathcal{N}_u$ the neighborhood of node u, i.e., the set of nodes that are one hop away from node u, and by $\mathbf{x}_T := [\mathbf{x}[n' - T + 1]; \ldots; \mathbf{x}[n']]$ the state vector during T consecutive time slots.

The dynamic caching optimization problem can then be formulated as

$$\hat{\mathbf{x}}_T = \arg \min_{\mathbf{x}_T} (E_{st}(\mathbf{x}_T) + E_{tr}(\mathbf{x}_T)) \qquad (0.15)$$

subject to the following constraints

$$(a) \qquad q_u[k, n] \le s_u[k, n] + \sum_{v \in \mathcal{N}_u} \sum_{j=0}^{D_u[k]} t_{vu}[k, n + j]$$

$$(b) \qquad s_u[k, n] \le s_u[k, n - 1] + \sum_{v \in \mathcal{N}_u} t_{vu}[k, n - 1]$$

$$(c) \qquad t_{vu}[k, n] \le s_v[k, n - 1] + \sum_{w \in \mathcal{N}_v} t_{wv}[k, n - 1]$$

$$(d) \qquad s_u[k, n] = 1, \forall k \in K_u[n], \ s_u[k, 0] = 0, k \notin K_u[n]$$

$$(e) \qquad S_u[n] \le S_u$$

$$(f) \qquad T_{uv}[n] \le T_{uv}$$

$$(g) \qquad s_u[k, n] \in \{0, 1\}, \ t_{uv}[k, n] \in \{0, 1\}, \qquad (0.16)$$

$\forall u \in \mathcal{V}, vu \in \mathcal{E}, k \in \mathcal{K}, n \in [n' - T + 1, n']$.

The above constraints reflect the storage and flow constraints [32]:
(a) ensures that if object k is requested by node u at time slot n, then k either is in the cache of node u at time n or needs to be received by node u from a neighbor node $v \in \mathcal{N}_u$ within $D_u[k]$ time slots;
(b) assures that if k is being cached at node u at time n, then k either was in the cache of u at time $n - 1$ or was received by node u from a neighbor node $v \in \mathcal{N}_u$ at time $n - 1$;
(c) assures that if object k is received by node u from a neighbor node $v \in \mathcal{N}_u$ at time n, then k either was in the cache of v at time $n - 1$ or was received by node v from a neighbor node $w \in \mathcal{N}_v$ at time $n - 1$;
(d) describes the initial condition constraints that assure that each node u always stores the objects that it hosts as a repository, $\mathcal{K}_u[n]$, and at $n = 0$ nothing else;
(e) and (f) define the storage and transport capacity constraints;
(g) states the binary nature of the network configuration (storage and transport) variables.

To simplify the solution of the above problem, we let the entries of vector $\mathbf{x}_T$ to be real variables in $[0, 1]$. A numerical example resulting from our relaxed formulation is shown in Fig. 0.4 where we illustrate the optimal transport energy vs. the arrival request rate. We consider a network composed of
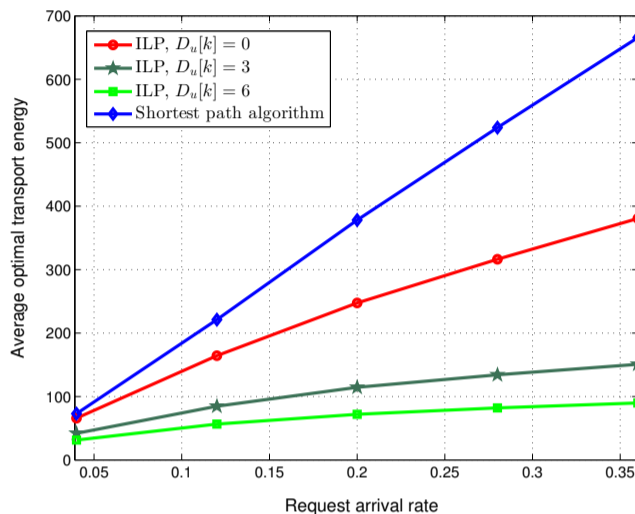
Figure 0.4   Average optimal transport energy vs. the request arrival rate.

$|\mathcal{V}| = 10$ nodes and $|\mathcal{K}| = 4$ information objects to be transported, by setting $T = 25$, $\triangle \tau = 1$s, $T_{uv} = 2$Mb, and $S_u = 4$. We considered, for simplicity, no knowledge of popularity and same transportation costs over all links. To better evaluate the effect of the transport energy, we neglected the storage energy E_{st} term in the integer linear program (ILP) (0.15), by assuming that only three repository nodes store the information objects for all time. As a benchmark method, we consider the shortest path algorithm, which at each request forwards the desired content along the shortest path. It can be noted that the relaxed ILP method yields a considerable performance gain with respect to the shortest path algorithm: moreover, the improvement grows as the maximum delivery time $D_u[k]$ (set equal for each k) increases, due to the greater degrees of freedom of the algorithm.

0.5 GRAPH-BASED RESOURCE ALLOCATION

Enabling proactive resource allocation strategies is a key feature of 5G networks. Proactivity is rooted on the capability to predict users' behavior. Proactive caching is one example where the prediction is based on learning the popularity matrix. But of course caching is not the only network aspect that can benefit from learning. Radio coverage is one more case where learning maps of the radio environment may be useful to ensure seamless connectivity to moving users, possibly keeping the smallest number of access points active to save energy. This requires prediction of users' mobility and the capability

to build Radio Environment Maps (REM) [34]. Building a REM is also a key step to enable cognitive radio [34], [35], [36]. Balancing data traffic across the network is another problem that could take advantage of the capability to predict data flows exploiting spatio-temporal correlation (low-rank) [37], [38].

0.5.1 RADIO ENVIRONMENT MAP

In this section, we show how graph-based representations can be useful to build a REM from sporadic measurements. Graph-based representations play a key role in many machine learning techniques, as a way to formally take into account all similarities among the entities of an interconnected system. In the signal processing community, there is a growing interest in methods for processing signals defined over a graph, or graph signal processing (GSP), for short [39]. We show now an application of GSP to recovering the REM in a urban environment from sporadic measurements collected by mobile devices. The goal is to reconstruct the field over an ideal grid, built according to the city map, starting from observations taken over a subset of nodes. We use a graph-based approach to identify patterns useful for the ensuing reconstruction from sparse observations. More specifically, given a set of N points in space, whose coordinate vectors are $\mathbf{r}_i$ and denoting with E_i the field measured at node i, we define the coefficients of the adjacency matrix $\mathbf{A}$ as $a_{ii} = 0$ and

$$a_{i,j} = \begin{cases} e^{-\frac{|E_i - E_j|^2}{2\sigma^2}}, & \text{if } ||\mathbf{r}_i - \mathbf{r}_j||^2 \leq R_0 \\ 0, & \text{otherwise} \end{cases}, i \neq j,$$

where σ and R_0 are two parameters used to assess the similarity of two nodes: σ is a variable used to establish the interval of values in the e.m. field within which two nodes are assumed to sense a *similar* value; R_0 is the distance within which two nodes are assumed to be neighbors. Building matrix $\mathbf{A}$ requires some prior information on the field that can be either acquired through time from measurements or it may be inferred from ray-tracing tools. From the adjacency matrix $\mathbf{A}$, we build the Laplacian matrix

$$\mathbf{L} = \mathbf{D} - \mathbf{A} \tag{0.17}$$

where $\mathbf{D}$ is the diagonal matrix whose i-th entry is the degree of node i: $d_i = \sum_{j=1}^{N} a_{ij}$. Taking the eigendecomposition of $\mathbf{L}$

$$\mathbf{L} = \mathbf{U}\mathbf{\Lambda}\mathbf{U}^T \tag{0.18}$$

we have a way to identify the principal components of the field. It is well known from spectral graph theory [40], in fact, that the eigenvectors associated to the smallest eigenvalues of $\mathbf{L}$ identify clusters, i.e., well connected components. Hence, the eigenvectors associated to the smallest eigenvalues of the Laplacian matrix built according to the above method are useful to identify patterns in the e.m. field. Denoting with $\mathbf{u}_k$ the eigenvector associated to the k-th

18   Chapter Title

eigenvalue, the useful signal $\mathbf{x}$ can then be modeled as the superposition of the K principal eigenvectors:

$$\mathbf{x} = \sum_{k=1}^{K} \mathbf{u}_k\, s_k := \mathbf{U}_K\, \mathbf{s}, \tag{0.19}$$

with $K < N$ to be determined from measurements and $\mathbf{U}_K := [\mathbf{u}_1, \ldots, \mathbf{u}_K]$.

In the GSP literature, a signal as in (0.19), with $K < N$, is called a *bandlimited* signal over the graph. In general, a real signal is never perfectly bandlimited, but it can be approximately bandlimited. Having a band-limited model is instrumental to establish the condition for the recovery of the entire signal from a subset of samples [41].

In a real situation, it is typical to have several access points whose radio coverage areas overlap. For each access point, we can build a dictionary using the method described above, using for the e.m. field a ray-tracing algorithm. We denote by $\mathbf{U}_K^{(m)}$ the dictionary built when only AP m is active. At any given time frame, only a few AP's are active. Therefore, the overall map can be written as

$$\mathbf{x} = \sum_{m=1}^{M} \sum_{k=1}^{K} \mathbf{u}_k s_k^{(m)} := \sum_{m=1}^{M} \mathbf{U}_K^{(m)} \mathbf{s}^{(m)} := \mathbf{U}\mathbf{s}, \tag{0.20}$$

where M is the number of AP's covering the area of interest (not all of them necessarily active at the same time), $\mathbf{U} := (\mathbf{U}_K^{(1)}, \ldots, \mathbf{U}_K^{(M)})$ and $\mathbf{s} := (\mathbf{s}^{(1)}; \ldots; \mathbf{s}^{(M)})$ is sparse. The observed signal typically consists in a limited number of measurements collected along the grid. We may write the observed signal as:

$$\mathbf{y} = \mathbf{\Sigma} \sum_{m=1}^{M} \mathbf{U}_K^{(m)} \mathbf{s}^{(m)} = \mathbf{\Sigma}\mathbf{U}\mathbf{s}, \tag{0.21}$$

where $\mathbf{\Sigma}$ is a diagonal selection matrix, whose i-th entry is one if node i is observed, and zero otherwise. The recovery of the overall radio coverage map can then be formulated as a sparse recovery problem. We used Basis Pursuit (BP), which implies solving the following convex problem:

$$\hat{\mathbf{s}} = \arg\min_{\mathbf{s}} \|\mathbf{s}\|_1$$
$$s.t.\ \mathbf{y} = \mathbf{\Sigma}\mathbf{U}\mathbf{s} \tag{0.22}$$

and then we used $\hat{\mathbf{x}} = \mathbf{U}\hat{\mathbf{s}}$.

An example of reconstruction using BP is shown in Fig.0.5. The grid is composed of $N = 547$ nodes and the number M of AP's covering the city area illustrated in the figure is 4. The AP's are located in the south-east, north-
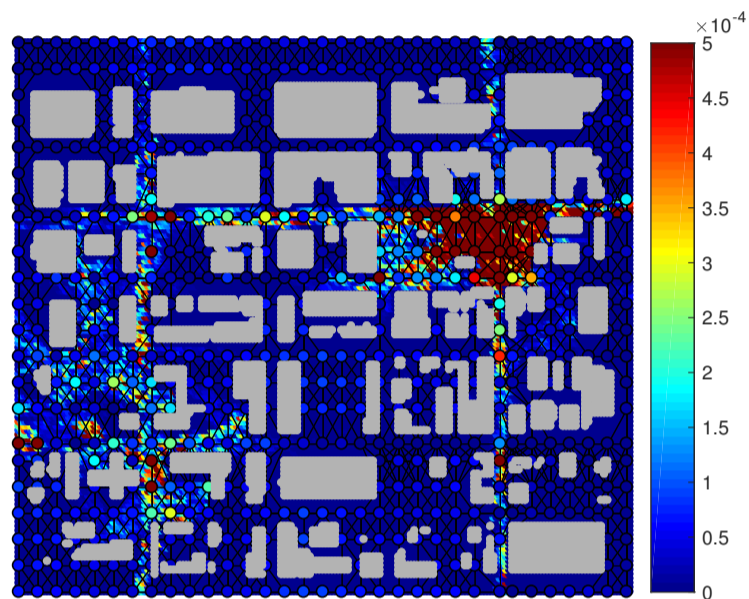
Figure 0.5   Example of reconstructed e.m. field.

east, north-west and south-west side of the examined area. The number of measurements is 115. Measurement noise is considered negligible. We assumed a bandwidth $K = 40$, equal for all AP's. The background (continuous) color is the map ground-truth, obtained using the ray-tracing tool Remcom Wireless InSite 2.6.3 [42]. The colors on each vertex of the grid represent the reconstructed value. Comparing each node color with the background, we can testify the goodness of the method to reconstruct the overall map. The Normalized Mean Square Error (NMSE), measured as the square norm of the error, normalized by the square norm of the true signal, in this example, is $NMSE = 0.018$. The quality of the reconstruction depends on the number of measurements and on the assumption on the bandwidth. Clearly, the larger is the bandwidth, the better is the reconstruction, but the larger is also the number of measurements to be taken to enable the reconstruction. This suggests that the choice of the bandwidth must come from a trade-off between accuracy and complexity.

### 0.5.2 **MATCHING USERS TO** $3C$ **RESOURCES**

In Section 0.3 we motivated the use of a joint allocation of computation and communication resources in computation offloading. We also incorporated the assignment rule between UE, AP, and MEC within the overall optimization problem. The resulting formulation yields better performance than a disjoint formulation, however it is also computationally demanding because it involves the solution of a mixed-integer programming problem.

A possible way to overcome this difficulty is to simplify the rule for associating UE's to AP and MEC. One possibility is to resort to *matching theory*, a low complexity tool used to solve the combinatorial problem of matching players from different sets, based on their preferences. Matching theory can be seen as the problem of finding a bipartite graph connecting two sets, depending on the preference lists. Matching theory has already been proposed in [43] for resource allocation in multi-tiered wireless heterogeneous architectures, with applications to cognitive radio networks, heterogeneous small-cell-based networks and Device-to-Device communications (D2D). In [44], a multi-stage matching game is used in the C-RAN context to assign Radio Remote Heads (RRH), Base Band Units (BBU) and computing resources for computation offloading, aimed at minimizing the refusal ratio, i.e. the proportion of offloading tasks that are not able to meet their deadlines. A well-known matching problem is the college admission game presented in [45], where a Deferred-Acceptance (DA) algorithm is proved to converge to a stable matching with extremely low complexity. The key initial step of matching theory is to establish a preference rule. For instance, in [46] the users' preferences are defined as the R-factor, which captures both Packet Success Rate (PSR) and wireless delay. However, as pointed out in [46], the complexity of this algorithm increases considerably when dealing with interdependent preferences, i.e. when the preference of a user is affected by the acceptance of the others. This is indeed the case of user association in wireless networks, because, continuing in the example defined above, the R factor of a user changes as other users get accepted by the same AP. To overcome this problem, the authors of [46] divide the game into two interdependent subgames:

1. An admission matching game with R-factor guarantees, depending on the maximum delay experienced at each access point;
2. A coalitional game among access points, where the coalitions are sets of AP's and associated users.

In particular, a user assigned to a certain AP a through the first subgame, could prefer to be matched to another AP b, since the utility functions change as users get admitted. Then, a user k requests to be transferred from a to b if it improves its R-factor. The transfer is accepted if and only if:

1. The access point b does not exceed its quota (maximum number of admitted
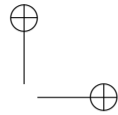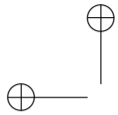
users);

2. The social welfare (sum of the R-factors of the two coalitions) is increased.

Starting from an initial partition (sets of coalitions) obtained with the deferred acceptance algorithm, the algorithm in [46] converge to a final partition that is also Nash-stable. In the holistic view of $3C$ resources, other utility functions can be used to take into account all the three aspects of $3C$: communication, computation, and caching. For instance, additional parameters to be taken into account are the computational load on MEC servers in case of computation offloading and the amount of storage for caching.

One more example where graph theory can be used is load balancing. In fact, especially in view of the dense deployment of access points, there is a high probability that the load, either data rate, computational load or storage, can be highly unbalanced throughout the network [47]. One possibility to balance the situation is to split the networks in many non-overlapping clusters. A cluster head is then elected in each cluster and it enforces a balance within the cluster. Then, balancing across clusters is achieved by repeated clustering and balancing steps. A possible way to do clustering is to use spectral clustering, which starts from the creation of a similarity (adjacency) matrix. In this case, as suggested in [48], it could be useful to include in the construction of the adjacency matrix a *dissimilarity* measure that assesses how much two nodes are unbalanced. In this way, the ensuing clustering tends to put together nodes that are close but unbalanced so that the resulting in-cluster balancing will be more effective.

0.6 NETWORK RELIABILITY

The edge-cloud architecture described in Section 0.2 clearly builds on the reliability of the network connectivity. However, in practice, the presence of a link between a pair of nodes is subject to random changes. In a wireless communication system, for instance, it is typical to have random link failures due to fading. With mmWave communications, link failures are typically even more pronounced because of blocking due to obstacles between transmit and receive devices. The goal of this section is to build on graph-based representations to assess the effect of random failure on a limited number of edge on macroscopic network parameters, such as, for example, connectivity. We build our study on a small perturbation analysis of the eigendecomposition of the Laplacian matrix describing the graph, as suggested in [49]. An outcome of our analysis is the identification of the most critical links, i.e. those links whose failure has a major effect on some network macroscopic features, such as connectivity.

A small perturbation analysis of the eigen-decomposition of a matrix is a classical problem that has been studied since a long time, see, e.g. [50], [51].

22   Chapter Title
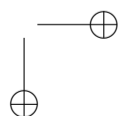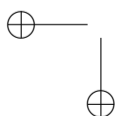
In this section we focus on the small perturbation analysis of the eigendecomposition of a perturbed Laplacian $\boldsymbol{L} + \delta\boldsymbol{L}$, incorporating an original graph Laplacian $\boldsymbol{L}$ plus the addition or deletion of a small percentage of edges. We consider a graph composed of N vertices, so that the dimension of $\boldsymbol{L}$ is $N \times N$. We denote by $\tilde{\lambda}_i = \lambda_i + \Delta\lambda_i$ the perturbed i-th eigenvalue and by $\tilde{\boldsymbol{u}}_i = \boldsymbol{u}_i + \Delta\boldsymbol{u}_i$ the associated perturbed eigenvector. If only one link fails, let us say link m, the perturbation matrix can be written as $\delta\boldsymbol{L}(m) = -\boldsymbol{a}_m\boldsymbol{a}_m^T$, where $\boldsymbol{a}_m = [a_{m_1} \cdots a_{m_n}]^T$ is a column vector of size N that has all entries equal to zero, except the two elements $a_m(i_m) = 1$ and $a_m(f_m) = -1$, where i_m and f_m are the initial and final vertices of the failing edge m. In case of addition of a new edge, the perturbation matrix is simply the opposite of the previous expression, i.e. $\delta\boldsymbol{L}(m) = \boldsymbol{a}_m\boldsymbol{a}_m^T$. It is straightforward to see that the perturbation of the Laplacian matrix due to the simultaneous deletion of a small set of edges is simply $\delta\boldsymbol{L} = -\sum_{m \in \mathcal{E}_p} \boldsymbol{a}_m\boldsymbol{a}_m^T$ where $\mathcal{E}_p$ denotes the set of perturbed edges. The perturbed eigenvalues and eigenvectors $\tilde{\lambda}_i$ and $\tilde{\boldsymbol{u}}_i$, in the case where all eigenvalues are distinct and the perturbation affects a few percentage of links, are related to the unperturbed values λ_i and $\boldsymbol{u}_i$ by the following formulas [50]:

$$\tilde{\lambda}_i \simeq \lambda_i + \boldsymbol{u}_i^T \delta\boldsymbol{L}\, \boldsymbol{u}_i \tag{0.23}$$

$$\tilde{\boldsymbol{u}}_i \simeq \boldsymbol{u}_i + \sum_{j \neq i} \frac{\boldsymbol{u}_j^T \delta\boldsymbol{L}\, \boldsymbol{u}_i}{\lambda_i - \lambda_j} \boldsymbol{u}_j. \tag{0.24}$$

In particular, the perturbations due to the failure of a generic link m on the i-th eigenvalue and associated eigenvector are:

$$\Delta\lambda_i(m) = \boldsymbol{u}_i^T \delta\boldsymbol{L}(m)\boldsymbol{u}_i = -\boldsymbol{u}_i^T \boldsymbol{a}_m\boldsymbol{a}_m^T\boldsymbol{u}_i =$$
$$= -||\boldsymbol{a}_m^T\boldsymbol{u}_i||^2 = -[u_i(f_m) - u_i(i_m)]^2 \tag{0.25}$$

and

$$\Delta\boldsymbol{u}_i(m) = \sum_{j \neq i} \frac{\boldsymbol{u}_j^T \delta L(m)\boldsymbol{u}_i}{\lambda_i - \lambda_j} \boldsymbol{u}_j = -\sum_{j \neq i} \frac{\boldsymbol{u}_j^T \boldsymbol{a}_m\boldsymbol{a}_m^T\boldsymbol{u}_i}{\lambda_i - \lambda_j} \boldsymbol{u}_j$$
$$= \sum_{j \neq i} \frac{[u_j(i_m) - u_j(f_m)][u_i(f_m) - u_i(i_m)]}{\lambda_i - \lambda_j} \boldsymbol{u}_j. \tag{0.26}$$

Within the limits of validity of first order perturbation analysis, the overall perturbation resulting from the deletion of multiple edges is the sum of all the perturbations occurring on single edges:

$$\Delta\lambda_i = \sum_{m \in \mathcal{E}_p} \Delta\lambda_i(m), \tag{0.27}$$

where $\mathcal{E}_p$ denotes the set of perturbed edges. In their simplicity, the above
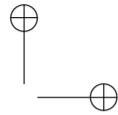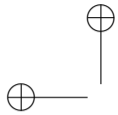
formulas capture some of the most relevant aspects of perturbation and their relation to graph topology. In fact, it is known from spectral graph theory, see e.g., [40], that the entries of the Laplacian eigenvectors associated to the smallest eigenvalues tend to be smooth and assume the same sign over vertices within a cluster, while they can vary arbitrarily across different clusters. Taking into account these properties, the above perturbation formulas (0.23)-(0.26) give rise to the following interpretations:

1. the edges whose deletion causes the largest perturbation are inter-cluster edges;
2. given a connected graph, the eigenvector associated to the null eigenvalue does not induce any perturbation on any other eigenvalue/eigenvector, because it is constant;
3. the eigenvector perturbation is larger for quantities (either eigenvalues or eigenvectors) associated to eigenvalues very similar to each other (recall that formulas (0.23) and (0.24) hold true only for distinct eigenvalues).

0.6.1 A NEW MEASURE OF EDGE CENTRALITY

Based on the above derivations, we propose a new measure of edge centrality, which we call *perturbation centrality*. We assume a connected undirected graph. If we denote by K the number of clusters in the graph and by $\Delta\lambda_i(m)$ the perturbation of the i-th eigenvalue due to the deletion of edge m, we define the topology perturbation centrality of edge m as follows [49]:

$$\mathsf{p}_K(m) := \sum_{i=2}^{K} |\Delta\lambda_i(m)|. \tag{0.28}$$

The summation starts from $i = 2$ simply because, from (0.23), the perturbation induced by the deletion of any edge on the smallest eigenvalue is null. The above parameter $\mathsf{p}_K(m)$ assigns to each edge the perturbation that its deletion causes to the overall network connectivity, measured as the sum of the K smallest eigenvalues of the Laplacian matrix [40]. This parameter is particularly relevant in case of modular graphs, i.e. graphs evidencing the presence of clusters. In such a case, it is well known from spectral clustering theory [40] that the smallest eigenvalues of the Laplacian carry information about the number of clusters in a graph.

In Fig. 0.6 we report an example of modular graph, obtained by connecting two clusters through a few edges. The *perturbation centrality* is encoded in the color intensity of each edge. It is interesting to see that the edges with the darkest color are, as expected, the ones connecting the two clusters.
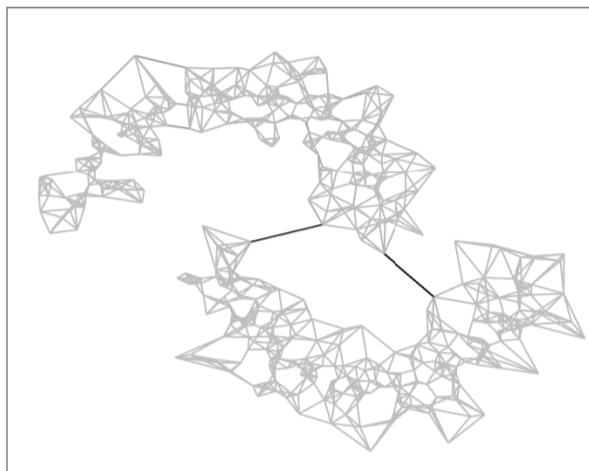
24   Chapter Title
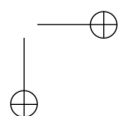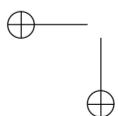
Figure 0.6   Example of perturbation centrality measure.

0.6.2 APPLICATION: ROBUST INFORMATION TRANSMISSION OVER WIRELESS NETWORKS

Now we apply our statistical analysis to optimize the resource (power) allocation over a wireless network in order to make the network robust against random link failures. We consider a wireless communication network with M links, where each link is subject to a random failure because of fading or blocking. Every edge is characterized by an outage probability $P_{out}(m), m = 1, \ldots, M$. We suppose the failure events over different links to be independent of each other. We consider first a single-input-single-output (SISO) Rayleigh flat fading channel for each link. In such a case, the channel coefficient h is a complex Gaussian random variable (r.v.) with zero mean and circularly symmetric. Hence, the r.v. $\alpha = |h|^2$ has an exponential distribution. Denoting with $F_n(x; \lambda)$ the cumulative distribution function (CDF) of a gamma random variable x of order n, with parameter λ, the CDF of α can then be written as $F_1(\alpha; \lambda)$. We also denote with $C = \log_2(1 + |h|^2 \rho)$ the link capacity (in bits/sec/Hz), where $\rho = \frac{P_T(m)}{\sigma_n^2 r_m^2}$ is the signal-to-noise ratio (SNR), $P_T(m)$ is the transmitted power over the m-th link, σ_n^2 is the noise variance, and r_m the distance covered by link m. Denoting by R the data rate, the outage

probability $P_{out}(m)$ is defined as:

$$P_{out}(m) = Pr\{C < R\} = Pr\{\log_2(1 + |h|^2\rho) < R\} \tag{0.29}$$

$$= Pr\{|h|^2 < \frac{2^R - 1}{\rho}\}$$

$$= \int_0^{\frac{2^R-1}{\rho}} \lambda e^{-\lambda\alpha} d\alpha = F_1\left(\frac{2^R - 1}{\rho}; \lambda\right) = 1 - e^{-\frac{\lambda}{\rho}(2^R-1)}.$$

Since the CDF of α is invertible, it is useful to introduce its inverse. In particular, if $y = F_n(x; \lambda)$, we denote its inverse as $x = F_n^{-1}(y; \lambda)$. Expression (0.29) can then be inverted to derive the transmit power $P_T(m)$ as a function of the outage probability:

$$P_T(m) = -\frac{\lambda\sigma_n^2 r_m^2 (2^R - 1)}{\log(1 - P_{out}(m))} = \frac{\sigma_n^2 r_m^2 (2^R - 1)}{F_1^{-1}(P_{out}(m); \lambda)}. \tag{0.30}$$

The small perturbation statistical analysis derived above can be used to formulate a robust network optimization problem. We assess the network robustness, in terms of connectivity, as the ability of the network to give rise to small changes of connectivity, as a consequence of a small number of edge failures. The network connectivity is measured by the second smallest eigenvalue of the Laplacian, also known as the graph *algebraic connectivity*. This parameter is known to provide a bound for the graph conductance [52]. Our goal now is to evaluate the transmit powers $P_T(m)$, or equivalently, through (0.30), the outage probabilities, that minimize the average perturbation of the algebraic connectivity, subject to a cost function on the total transmit power $P_{T_{max}}$ of the overall network. In formulas, we wish to solve the following optimization problem:

$$\min_{\mathbf{P}_{out}} \quad \sum_{m \in \mathcal{E}} \mathbb{E}\{|\Delta\lambda_2(m)|\}$$

$$s.t. \quad \sum_{m \in \mathcal{E}} P_T(m) \le P_{T_{max}}$$

$$P_{out}(m) \in [0, 1], \, \forall \, m \in \mathcal{E}.$$

Using equation (0.25) and (0.30), we can rewrite the optimization problem explicitly in terms of the outage probabilities $P_{out}(m)$ as:

$$\min_{\mathbf{P}_{out}} \quad \sum_{m \in \mathcal{E}} P_{out}(m)[u_2(i_m) - u_2(f_m)]^2$$

$$s.t. \tag{$\mathcal{Q}$}$$

$$\sum_{m \in \mathcal{E}} \frac{r_m^2}{F_1^{-1}(P_{out}(m); \lambda)} \le C_{max}$$

$$P_{out}(m) \in [0, 1], \, \forall \, m \in \mathcal{E}$$

where $C_{max} := \frac{P_{T_{max}}}{\sigma_n^2 (2^R - 1)}$.

Problem $(\mathcal{Q})$ is non-convex because the constraint set is not convex. However, if we perform the change of variable $t_m := 1/F_1^{-1}(P_{out}(m); \lambda) = -\lambda / \log(1 - P_{out}(m)), m = 1, \ldots, M$, the first constraint becomes linear. The objective function becomes non-convex. However, if we limit the variability of the unknown variables to the set $t_m \geq \lambda/2, \forall m$, the objective function becomes convex, so that the original problem converts into the following convex problem:

$$
\begin{aligned}
\min_{\mathbf{t}} \quad & \sum_{m \in \mathcal{E}} F_1(\tfrac{1}{t_m}; \lambda)|\Delta\lambda_2(m)| = \sum_{m \in \mathcal{E}} (1 - e^{-\frac{\lambda}{t_m}})|\Delta\lambda_2(m)| \\
s.t. \quad & \sum_{m \in \mathcal{E}} r_m^2 t_m \leq C_{max} \qquad\qquad (\mathcal{Q}_1) \qquad (0.31) \\
& t_m \geq \tfrac{\lambda}{2}, \quad \forall m \in \mathcal{E}.
\end{aligned}
$$

We can now generalize the previous formulation to the Multi-Input Multi-Output (MIMO) case, assuming multiple independent Rayleigh fading channels. One fundamental property of MIMO systems is the diversity gain, which makes them more robust against fading with respect to SISO systems [53]. In fact, different performance can be obtained depending on the number of antennas on the transmitting sides n_T and receiving sides n_R exploiting the diversity gain. In a MIMO system whit $n = n_T \times n_R$ statistically independent channels, denoting by h_{ij} the coefficient between the i-th transmit and the j-th receive antenna, the pdf of the random variable $\alpha := \sum_{i=1}^{n_T} \sum_{j=1}^{n_R} |h_{ij}|^2$ is the Gamma distribution:

$$
P_A(\alpha) = \frac{\lambda^n}{(n-1)!} \alpha^{n-1} e^{-\lambda\alpha} \qquad (0.32)
$$

and we denote by $F_n(\alpha; \lambda)$ its cumulative distribution function (CDF), with parameters n and λ. Proceeding similarly to the SISO case, the optimization problem can be formulated as

$$
\begin{aligned}
\min_{\mathbf{t}} \quad & \sum_{m \in \mathcal{E}} F_n(\tfrac{1}{t_m}; \lambda)|\Delta\lambda_2(m)| \\
s.t. \quad & \sum_{m \in \mathcal{E}} r_m^2 t_m \leq C_{max} \qquad\qquad (\mathcal{Q}_2) \qquad (0.33) \\
& t_m \geq \lambda/(n+1), \quad \forall m \in \mathcal{E}
\end{aligned}
$$

where the constraint on the variables t_m has been introduced to make the problem convex. Indeed, problem $\mathcal{Q}_1$ is a special case of problem $\mathcal{Q}_2$, when $n = 1$. An interesting result about the convexity of problem $\mathcal{Q}_2$ is that the bounding region increases with the number of independent channels.

As a numerical example, we considered a connected network composed by two clusters, with a total of $|\mathcal{E}| = 761$ edges and four bridge edges between the two clusters. For the sake of simplicity, we assumed the same distances r_m over all links. In Fig. 0.7, we compare the expected perturbations of the

algebraic connectivity, normalized to the nominal value λ_2, obtained using our optimization procedure or using the same power over all links, assuming the same overall power consumption. We report the result for both SISO and MIMO cases. From Fig. 0.7, we can observe a significant gain in terms of the total power necessary to achieve the same expected perturbation of the network algebraic connectivity. We can also see the advantage of using MIMO communications, at least in the case of statistically independent links.
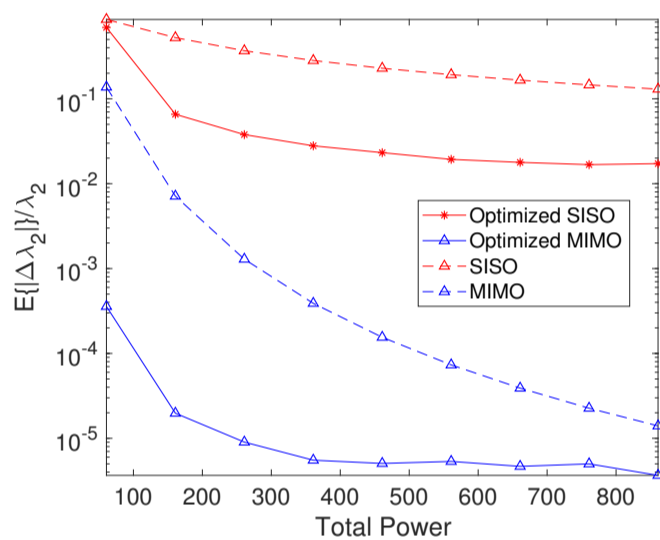

Figure 0.7   Expected perturbation of algebraic connectivity vs. total power.

0.7 **CONCLUSIONS**

In this chapter we have described some of the aspects of the edge-cloud architecture, a framework proposed to bring cloud and communication resources as close as possible to mobile users to reduce latency and achieve a more efficient usage of the available energy. From the edge-cloud perspective, we have motivated a holistic view that aims at optimizing the allocation of communication, computation and caching resources jointly. Within this framework, graph-based representations play a key role. In this chapter, we considered just a few cases where these representations can provide a valid and innovative tool for an efficient deployment of the edge-cloud system. As it happens in most engineering problems, big potentials come with big challenges. One of

these is complexity. To take full advantage of graph representations, there is the need for devising efficient distributed computational tools to analyze graph-based signals. Furthermore, we believe that graph representations are only the beginning of the story, as they are built incorporating only pairwise relations. More sophisticated tools may be envisaged by enlarging the horizon to include multi-way relations, using for example simplicial complexes or hypergraphs, as suggested in [54], or multilayer network representations [55], [56]. Furthermore, in this work, we have basically restricted our attention to time-invariant graph representations and to linear models. Clearly, a significant improvement can be expected by enlarging the view to time-varying graphs and nonlinear models [57], [58].

0.8 ACKNOWLEDGMENTS

REFERENCES

1. 5G empowering vertical industries. 5G PPP White paper, 2016.
2. Andrews JG, Buzzi S, Choi W, Hanly SV, Lozano A, Soong AC, et al., What will 5G be? IEEE J Sel Areas Commun 2014; 32(6):1065–1082.
3. Mijumbi R, Serrat J, Gorricho JL, Bouten N, De Turck F, Boutaba R, Network function virtualization: State-of-the-art and research challenges. IEEE Commun Surveys & Tutorials 2016; 18(1):236–262.
4. Rost P, Mannweiler C, Michalopoulos DS, Sartori C, Sciancalepore V, Sastry N, et al., Network slicing to enable scalability and flexibility in 5G mobile networks. IEEE Commun Mag 2017; 55(5):72–79.
5. Vassilaras S, Gkatzikis L, Liakopoulos N, Stiakogiannakis IN, Qi M, Shi L, et al., The algorithmic aspects of network slicing. IEEE Commun Mag 2017; 55(8):112–119.
6. Heath RW, Gonzalez-Prelcic N, Rangan S, Roh W, Sayeed AM, An overview of signal processing techniques for millimeter wave MIMO systems. IEEE J Sel Topics Signal Process 2016; 10(3):436–453.
7. Xiao M, Mumtaz S, Huang Y, Dai L, Li Y, Matthaiou M, et al., Millimeter wave communications for future mobile networks. IEEE J Sel Areas Commun 2017; 35(9):1909–1935.
8. Sakaguchi K, Haustein T, Barbarossa S, Calvanese-Strinati E, Clemente A, Destino G, et al., Where, when, and how mmWave is used in 5G and beyond. IEICE Trans Electron 2017; E100-C(10):790–808.
9. Taleb T, Samdanis K, Mada B, Flinck H, Dutta S, Sabella D, On multi-access edge computing: A survey of the emerging 5G network edge architecture & orchestration. IEEE Commun Surveys & Tutorials 2017; 19(3):1657–1681.

10. Hu YC, Patel M, Sabella D, Sprecher N, Young V, Mobile edge computing: A key technology towards 5G. ETSI White Paper 2015; (11).
11. Wang S, Zhang X, Zhang Y, Wang L, Yang J, Wang W, A survey on mobile edge networks: Convergence of computing, caching and communications. IEEE Access 2017; 5:6757–6779.
12. Bonomi F, Milito R, Natarajan P, Zhu J, Fog computing: A platform for Internet of things and analytics. In: Big Data and Internet of Things: A Roadmap for Smart Environments, Springer, 2014; pp. 169–186.
13. Zhang N, Liu YF, Farmanbar H, Chang TH, Hong M, Luo ZQ, Network slicing for service-oriented networks under resource constraints. IEEE J Sel Areas Commun 2017; PP(99):1–1.
14. Barbarossa S, Sardellitti S, Di Lorenzo P, Communicating while computing: Distributed mobile cloud computing over 5G heterogeneous networks. IEEE Signal Processing Mag 2014; 31(6):45–55.
15. Wang C, Liang C, Yu FR, Chen Q, Tang L, Computation offloading and resource allocation in wireless cellular networks with mobile edge computing. IEEE Trans Wireless Commun 2017; 16(8):4924–4938.
16. Barbarossa S, Sardellitti S, Di Lorenzo P, Joint allocation of computation and communication resources in multiuser mobile cloud computing. In: IEEE Workshop SPAWC 2013, pp. 26–30.
17. Sardellitti S, Scutari G, Barbarossa S, Joint optimization of radio and computational resources for multicell mobile-edge computing. IEEE Trans Signal Inform Process over Networks 2015; 1(2):89–103.
18. Pathak A, Hu YC, Zhang M, Where is the energy spent inside my app?: fine grained energy accounting on smartphones with Eprof. In: Proceedings of the 7th ACM European Conf Computer Systems, ACM, 2012, pp. 29–42.
19. Barbarossa S, Ceci E, Merluzzi M, Overbooking radio and computation resources in mmW-mobile edge computing to reduce vulnerability to channel intermittency. In: 2017 European Conf on Networks and Commun (EuCNC), 2017, pp. 1–5.
20. 5G-MiEdge Millimeter-wave Edge Cloud as an Enabler for 5G Ecosystem. Europe/Japan project co-funded by the European Commission's Horizon 2020 and Japanese Ministry of Internal Affairs and Communications, website: http://5g-miedge.eu.
21. Andrews JG, Bai T, Kulkarni MN, Alkhateeb A, Gupta AK, Heath RW, Modeling and analyzing millimeter wave cellular systems. IEEE Trans Commun 2017; 65(1):403–430.
22. Barbarossa S, Ceci E, Merluzzi M, Calvanese-Strinati E, Enabling effective mobile edge computing using millimeterwave links. In: 2017 IEEE Int Conf on Commun Workshops (ICC Workshops), 2017, pp. 367–372.
23. Mobile Edge Computing (MEC); End to End Mobility Aspects. ETSI GR MEC 018 V1.1.1, 2017.
24. Li W, Kanso A, Comparing containers versus virtual machines for achieving high availability. In: 2015 IEEE Int Conf on Cloud Engineering (IC2E), 2015, pp. 353–358.
25. Sardellitti S, Barbarossa S, Scutari G, Distributed mobile cloud computing: Joint optimization of radio and computational resources. In: 2014 IEEE Globecom Workshops (GC Wkshps), 2014, pp. 1505–1510.
26. Sardellitti S, Barbarossa S, Merluzzi M, Optimal association of mobile users to multi-access edge computing resources. Submitted to IEEE Trans. Signal Inform Process over

30   Chapter Title

Networks, 2017.

27. Scutari G, Facchinei F, Lampariello L, Parallel and distributed methods for constrained nonconvex optimization - Part I: Theory. IEEE Trans Signal Process 2017; 65(8):1929–1944.

28. Baştuğ E, Bennis M, Zeydan E, Kader MA, Karatepe IA, Er AS, et al., Big data meets telcos: A proactive caching perspective. J Commun Networks 2015; 17(6):549–557.

29. Zeydan E, Bastug E, Bennis M, Kader MA, Karatepe IA, Er AS, et al., Big data caching for networking: Moving from cloud to edge. IEEE Commun Mag 2016; 54(9):36–42.

30. Sadeghi A, Sheikholeslami F, Giannakis GB, Optimal and scalable caching for 5G using reinforcement learning of space-time popularities. Preprint arXiv:1708.06698, 2017.

31. Jacobson V, Smetters DK, Thornton JD, Plass MF, Briggs NH, Braynard RL, Networking named content. In: Proc of the 5th Int Conf on Emerging Networking Experiments and Technologies, ACM, 2009, pp. 1–12.

32. Llorca J, Tulino AM, Guan K, Esteban J, Varvello M, Choi N, et al., Dynamic in-network caching for energy efficient content delivery. In: INFOCOM, 2013 Proceedings IEEE, 2013, pp. 245–249.

33. Krishnan P, Raz D, Shavitt Y, The cache location problem. IEEE/ACM Transactions on Networking (TON) 2000; 8(5):568–582.

34. Bazerque JA, Mateos G, Giannakis GB, Group-lasso on splines for spectrum cartography. IEEE Trans Signal Process 2011; 59(10):4648–4663.

35. Yilmaz HB, Tugcu T, Alagoz F, Bayhan S, Radio environment map as enabler for practical cognitive radio networks. IEEE Commun Mag 2013; 51(12):162–169.

36. Romero D, Kim SJ, Giannakis GB, Lpez-Valcarce R, Learning power spectrum maps from quantized power measurements. IEEE Trans Signal Process 2017; 65(10):2547–2560.

37. Mardani M, Giannakis GB, Robust network traffic estimation via sparsity and low rank. In: 2013 IEEE Int Conf Acoustics, Speech, Signal Process (ICASSP), 2013, pp. 4529–4533.

38. Xu J, Deng D, Demiryurek U, Shahabi C, van der Schaar M, Mining the situation: Spatiotemporal traffic prediction with big data. IEEE J Sel Topics Signal Process 2015; 9(4):702–715.

39. Shuman DI, Narang SK, Frossard P, Ortega A, Vandergheynst P, The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains. IEEE Signal Proc Mag 2013; 30(3):83–98.

40. Von Luxburg U, A tutorial on spectral clustering. Stat Comput 2007; 17(4):395–416.

41. Tsitsvero M, Barbarossa S, Di Lorenzo P, Signals on graphs: Uncertainty principle and sampling. IEEE Trans Signal Process 2016; 64(18):4845–4860.

42. URL: https://www.remcom.com/.

43. Gu Y, Saad W, Bennis M, Debbah M, Han Z, Matching theory for future wireless networks: fundamentals and applications. IEEE Commun Mag 2015; 53(5):52–59.

44. Li T, Magurawalage CS, Wang K, Xu K, Yang K, Wang H, On efficient offloading control in cloud radio access network with mobile edge computing. In: 2017 IEEE 37th Int Conf Distributed Computing Systems (ICDCS), 2017, pp. 2258–2263.

45. D G, Shapley LS, College admissions and the stability of marriage. The Amer Math Monthly 1962; 69(1):915.

46. Saad W, Han Z, Zheng R, Debbah M, Poor HV, A college admissions game for uplink user association in wireless small cell networks. In: IEEE INFOCOM 2014 - IEEE Conf

Computer Commun, 2014, pp. 1096–1104.

47. Vu TK, Bennis M, Samarakoon S, Debbah M, Latva-aho M, Joint load balancing and interference mitigation in 5G heterogeneous networks. IEEE Trans Wireless Commun 2017; 16(9):6032–6046.

48. Samarakoon S, Bennis M, Saad W, Latva-Aho M, Dynamic clustering and sleep mode strategies for small cell networks. In: 2014 11th Int Symposium on Wireless Commun Systems (ISWCS), 2014, pp. 934–938.

49. Ceci E, Barbarossa S, Small Perturbation Analysis of Network Topologies. Submitted to ICASSP 2018.

50. Wilkinson JH, The Algebraic Eigenvalue Problem. New York, NY, USA: Oxford University Press, Inc., 1988.

51. Stewart G, Introduction to matrix computations. Computer Science and Applied Mathematics, Academic Press, 1973.

52. Newman M, Networks: An Introduction. 2010.

53. Barbarossa S, Multiantenna Wireless Communication Systems. Mobile Communications Series, Artech House, 2003.

54. Barbarossa S, Tsitsvero M, An introduction to hypergraph signal processing. In: 2016 IEEE Int Conf Acoustics, Speech, Signal Process (ICASSP), 2016, pp. 6425–6429.

55. Kivelä M, Arenas A, Barthelemy M, Gleeson JP, Moreno Y, Porter MA, Multilayer networks. Journal of complex networks 2014; 2(3):203–271.

56. Boccaletti S, Bianconi G, Criado R, Del Genio CI, Gómez-Gardenes J, Romance M, et al., The structure and dynamics of multilayer networks. Physics Reports 2014; 544(1):1–122.

57. Shen Y, Baingana B, Giannakis GB, Kernel-based structural equation models for topology identification of directed networks. IEEE Transactions on Signal Processing 2017; 65(10):2503–2516.

58. Romero D, Ioannidis VN, Giannakis GB, Kernel-based reconstruction of space-time functions on dynamic graphs. IEEE J of Sel Topics in Sig Proc 2017; 11(6):856–869.