

(c) 2016 IEEE. Personal use of this material is permitted. Permission from IEEE must be obtained for all other users, including reprinting/ republishing this material for advertising or promotional purposes, creating new collective works for resale or redistribution to servers or lists, or reuse of any copyrighted components of this work in other works.

Foggy clouds and cloudy fogs: a real need for coordinated management of fog-to-cloud (F2C) computing systems

Xavi Masip-Bruin, Eva Marín-Tordera, Ghazal Tashakor
Universitat Politècnica de Catalunya
CRAAX-UPC, Spain
{xmasip, eva, tashakor}@ac.upc.edu

Admela Jukan
Technische Universität Braunschweig
Germany
a.jukan@tu-bs.de

Guang-Jie Ren
IBM, Almaden Research Center, USA
gren@us.ibm.com

Abstract – The recent advances in the cloud services technology are fueling a plethora of information technology innovation, including networking, storage and computing. Today, various flavors have evolved of Internet of Things (IoT), cloud computing and the so-called fog computing, – a concept referred to capabilities of edge-devices and user’s clients to compute, store and exchange data among each other and with the cloud. Though the evolution was not easily foreseeable to happen at such a rapid pace, each piece of it today facilitates and enables the deployment of what we commonly refer to as a *smart* scenario, including smart cities, smart transportation and smart homes. As most of the cloud, fog and network services today run simultaneously in each scenario, we observe that we are at the dawn of what maybe the next big step in the cloud computing and networking evolution, whereby services might be executed at the network edge, both in parallel and in a coordinated fashion, as well as supported by the unstoppable technology evolution. As edge devices become richer in functionality and smarter, – embedding capacities such as storage or processing, as well as embedding new functionalities, such as decision making, data collection and forwarding, sharing, etc, a real need is emerging for coordinated management of fog-to-cloud (F2C) computing systems. This paper introduces a layered fog-to-cloud (F2C) architecture, its benefits and strengths as well as the arising open and research challenges, making the case for the real need for their coordinated management. Our architecture, the illustrative use case presented and a comparative performance analysis, albeit conceptual, all clearly show the way forward towards a new IoT scenario with a set of existing and unforeseen services provided on a highly distributed and dynamic compute, storage and networking resources, bringing together heterogeneous and commodity edge devices, emerging fogs as well as conventional clouds.

Keywords- Cloud computing, fog computing, fog-to-cloud, Internet of Things (IoT)

I. INTRODUCTION: THE SCENARIO

The most recent developments in the information and communications technologies area have started to make a profound impact, through massive connectivity of humans and computers, as well as a massive proliferation of edge devices carried by humans (i.e., smart phones, and those associated with all the surroundings – the Internet of Things). These two major commodities not only have facilitated the true “anywhere, anyhow, anytime” users’ connectivity, but also the data collection, further enabling the deployment of new value-added services. Today, the scenarios of smart cities, smart transportation and smart homes are no more domain of research of distant future, but are becoming the new “normal”. Several references can be found in the literature that already showed the notable effect these concepts can bring to the business market [1]. For a rapid business and technological success to happen, however, two inherent features need to be addressed in the process of introduction of edge devices, and that is the management of their fundamental features, most notably including (i) their transitional presence and (ii) a fundamentally decentralized control and operation. The transitional presence of edge devices is typical due to end-user mobility, and it stresses technological and connectivity providers with a significant set of problems, such as security, reliability, Quality of Service (QoS), etc., many of them currently addressed under the umbrella of 5G. Decentralized operation, on the other hand, is due to the low processing and storage capacities embedded into edge devices. As such, it is introducing other known, but widely unsolved challenges in distributed systems about global control, coordination, etc. As cloud computing paradigm has been widely adopted precisely to alleviate the problems of decentralized computing and storage, the question arises of whether the further extension of cloud computing principles would be the best way forward in integrating IoT and user’s end-devices.

However, despite several commercial products available on the market made cloud computing an indispensable communication and computing paradigm for all users and Internet applications, a couple of major limitations remain unsolved in regard to edge-devices. First, the larger the distance between the cloud and the edge device, measured in the amount of connectivity services required to reach the cloud, the lower the performance, efficiency of resource usage, as well as security/reliability guarantees. Second, cloud computing and storage service notoriously cannot provide guarantees to low latency applications. On the other hand, advocates of the fog computing concepts claim the solutions to these limitations. In fog computing, since data to be processed is usually close to the user, common in IoT scenarios, there would be no need to process the data far away from the user [2]. In this way, fog computing has been conceptualized to facilitate the deployment of new services that cloud computing cannot provide, particularly including those demanding real time guarantees. A classical example of a fog computing service is an emergency medicine service with real-time low-latency constraints associated with implantable medical devices, emergency car communications, or mobile access to patient health records. This leads to a second major feature of fog computing, which can incorporate the user’s mobility as an inherent

feature. For example, vehicular fog-clouds can be built by grouping cars located in a parking lot; roadside clouds can be built by putting together buildings, traffic lights, etc. Even humans can “group” by creating neighborhoods clouds, ad-hoc communities, setting an old wish list, but yet-unrealized scenario of customized dynamic clouds that can be set, released and offered on-demand, as a service, according to certain policies and access agreements. Also from the business perspective, the fog computing concept fuelled a unique change in the way traditional cloud computing was managed, paving the way to the new actors in technology innovation, including municipalities, cities, transport providers, public institutions, etc. That said, fog computing alone is no panacea, and a few issues, if unsolved, can prevent its wide deployment, including:

- Volatility due to intermittent presence: the mobility linked to fog brings a high level of volatility in both the processed data and the infrastructure used. As mentioned, in fog computing the mobility is inherent. At the same time the infrastructure created is temporary for the very same reasons. For instance, gridding of capacities located in cars parked in a mall (infrastructure and even energy) can be made available as long as the cars stay parked there.
- Security and reliability: the fog infrastructure typically located at the edge and usually without strict control from a provider, is inherently more sensible than a controlled cloud. Hence, although some security concerns might be minimized because of the resources proximity, there is a new security factor coming from the concept.
- Business model: while the business model in cloud computing is very clear in identifying providers and clients, the scenario envisioned in fog is much more open, which is an opportunity for a new wave of shared economy, but also a challenge. Going back to the cars parked on the mall, a few fundamental questions are: Who is the fog provider? Who is setting the agreement to get these resources? How do resources grid to increase the offered capacities?
- Reliability: the dynamism associated to fog devices turns into a low reliable infrastructure, what undoubtedly impacts on the quality offered by the services to be executed.

To address the above challenges, we believe that a real need has emerged to address the different cloud/fog resources in a joint framework capable of managing the emerging Fog-to-Cloud (F2C) computing and network architecture. In our mind, user’s edge devices and IoT devices embedded into environment will continue to leverage the capacities provided by cloud, and gradually include functions innate to fog computing, in a joint process available to collect, exchange and analyze data, which effectively will result in a large set of new services offered to users. These services will be provided at either cloud or fog premises, or both, and that according to the required services needs and demands. In this way, traditional cloud providers will be empowered to extend their reach closer to the edge (foggy cloud), which at the same time edge devices will be able to offer more capacities than currently envisioned (cloudy fog). To address this evolution, we envision and propose a novel hierarchical architecture, referred to as Fog-to-Cloud (F2C) computing, consisting of a layered management structure that can bring together different heterogeneous cloud/fog layers into a hierarchical architecture, whereby all different cloud implementations and concepts can be dynamically setup and released, while offering user-driven customization and services parallelization. This paper introduces the F2C architecture as well as the need for a control strategy to coordinate the different resources building the overall layered architecture. We present and discuss a few use cases and show illustrative results on the benefits of coordinated control and management of a combined fog and cloud system.

The rest of the paper is organized as follows. Section II gives an overview of the proposed F2C architecture, including a few key references to the related work, along with an illustrative example of a coordinated control strategy. Section III goes in media res with one of the highly demanding scenarios of emergency medicine service, along with the numerical results showing the benefits of the F2C architecture proposed. Finally, Section IV outlines open issues and challenges and concludes the paper.

II. THE FOG-TO-CLOUD (F2C) ARCHITECTURE: A NEED FOR COORDINATED MANAGEMENT

A) F2C Architecture: from static to dynamic coordination

From a conceptual perspective, the F2C architecture is a new, but rather realistic scenario brought by the IoT services requirements, users demands and technology progress that, to the best of our knowledge, has not been addressed before in a coordinated fashion. Currently, there is no contribution describing and detailing a hierarchical or layered architecture needed to address the F2C architecture. That said it is a concept related to some recently proposed approaches. The recent literature, see survey in [3], in particular, focuses on the concept of Autonomous Vehicular Clouds (AVC) introduced as "*A group of largely autonomous vehicles whose corporate computing, sensing, communication and physical resources can be coordinated and dynamically allocated to authorized users*", posing for the first time the idea of car gridding. The same work surveys solutions for vehicles clustering (e.g., Vehi-Cloud), suggesting that vehicles can be turned into mobile nodes (managed with location based routing), setting a layered architecture consisting in vehicular cyber physical system network, vehicle-to-vehicle network and vehicle to infrastructure layers, all with an access to cloud computing facilities; as well as ITS-Cloud consisting in setting three non-extendable layers, cloud, communication and end-user, aimed at optimizing application performance in terms of security and reliability.

The layering concept was extended to hierarchical cloud computing in [4], and further applied in specific scenarios of a scheduling strategy, designed to load balance and reducing the execution time. Yet another and different hierarchical cloud architecture for vehicular clouds was proposed in [5], with concepts closest to the F2C concept, based on a three-layered architecture to organize the

cloud resources into layers, divided into central (conventional) cloud, roadside cloud and vehicular cloud. The work describes the particular benefits of a hierarchical cloud architecture to vehicular traffic management and optimized IT resources utilization. The key conceptual difference with F2C computing that we are proposing here is that our proposed hierarchy is not static, but is a more open and dynamic cloud layered framework, where services can dynamically use and choose both a specific cloud offer (i.e., a specific layer in hierarchy) as well as the suitable cloud or fog resources for the service execution (inside each layer).

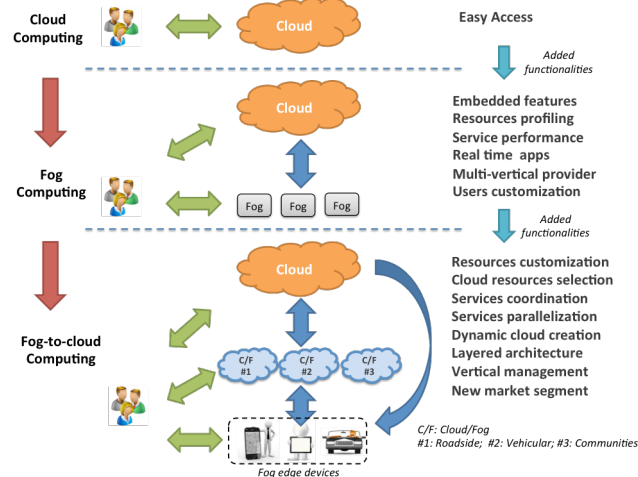


Fig. 1. Cloud evolution towards fog-to-cloud computing

Figure 1 introduces the Fog-to-Cloud (F2C) concept as an evolution of the current and expected progress in cloud computing. The F2C architecture is envisioned to leverage the progress made in both the cloud and fog computing areas, thus benefitting from but also inheriting strengths of both concepts, while addressing weaknesses in a coordinated and holistic fashion. The cloud computing paradigm (top row in Figure 1) started in the last decade to facilitate the decoupling of computing resources from the edge terminals. Although widely adopted, cloud computing has identified several weaknesses, such as issues under real-time conditions, high network load, low security, etc. This in turn has hindered the deployment of novel IoT services, leading to the advent of fog computing (center row in Figure 1) as a cloud-based paradigm with extended key features for real-time application support, low latency, load reduction and energy efficiency. However, in the future scenario, as shown at the bottom of Figure 1, we envision a mix of both concepts, where different clouds and fogs can be built through traditional clouds, intermediate clouds/fog systems and edge fog devices. In such a scenario, as shown in the Figure 1, the different fogs and clouds are defined as layers setting a hierarchical (layered) F2C architecture where the service running on the users' device can decide on-the-fly the best suited fog/cloud resource.

We envision a hierarchical architecture leveraging cloud and fog framework in a coordinated fashion, making it possible for new services to be created, but also improving the performance, such execution time, parallel execution, edge processing, fog security, low resources utilization and high energy efficiency. To this end, a comprehensive control and management strategy must be defined, addressing coordination aspects as well as the different relationships to be set among the different cloud/fog components.

B) The need for coordinated management

For the whole ecosystem of Fog-to-Cloud computing to work in a dynamic, collaborative and coordinated way, the control and management is the key challenge. For example, in the event of a car accident, the workflow of actions to be executed by all different involved actors at their own cloud premises, such as a taxis fleet manager, traffic lights controller, garbage trucks fleet manager, scholar buses, overall traffic control in a city, ambulance managers and fireman department, must be coordinated, orchestrated and efficiently managed to provide the more appropriate real-time response. Thus, what is with no doubt happening in this evolution is that the static, single cloud management model is evolving into a new model, the one that is much more dynamic, and heterogeneous, located at different, heterogeneous and usually on the move fog premises, offering distinct services that can be linked together, as well as capacities that can be jointly offered. Thus, the design of a coordinated management strategy becomes critical and needs to address the following associated challenges:

Cloud/fogs identification: An existing cloud capacity must be identified before being used. This means that an *address*, a *label* or a *name* must be linked to the said cloud capacity. This is especially important when considering dynamic clouds, whose time in the market is not prefixed, constant or even guaranteed.

Semantic adaptation: Performance and expected capacities are far different depending on the cloud/fog characteristics. Therefore, the adequate taxonomy should be defined to facilitate the mapping between the capacities required by a service and the capacities offered by a cloud layer, and thus optimize the allocation of resources. This taxonomy must be supported by a semantic adaptation

service, defining metrics and attributes offered by the participating cloud layers and including new parameters, such as static/dynamic (i.e., infrastructure perdurable in time or not), time-in-market (how long this infrastructure is available), time-to-leave (time to expected teardown), connectivity, backup or QoS, are undoubtedly critical in this scenario.

Coordinated layer orchestration: Overall orchestration is required to: i) generate an individual workflow of actions for each service; ii) map the individual action into the fog layer taxonomy best suited for the expected requirements, and; iii) coordinate the interactions among the different layers involved in the execution of the service. Orchestration of services can be ad-hoc or pre-defined, based on the current practices. For instance, in case of medical emergency it is likely that service workflows will be pre-defined, as defined in current emergency systems and based on health care policy and regulation.

Cloud resources discovery and allocation: Executing a service requiring different fog layers to interact each other will first demand the proper selection, and in some cases creation, of these fog layers. A management entity must be responsible for discovering the set of available (visible) fogs and then choosing those that can best meet the service's requirements. The fog visibility might be handled by a publish/subscribe strategy (e.g. a solution based on the marketplace concept introduced in [6]), or by a simple registration process, based on the cloud owner's profile (e.g. a solution extending the resource reservation scheme for virtual machine migration proposed in [5]).

Services execution scheduling: Service scheduling is required to decide how a service's individual functions are split into the different fog layers, and how each layer processes the services. Many considerations must be observed. For example, a traffic light system managing real time traffic must be robust against an ad-hoc changing of the defined rules, since its impact on real-life traffic conditions is high. A police or a medical emergency may shift that traffic light settings indeed, but a smart processing is required to minimize the impact of this particular decision on the overall city traffic. Preferences, priorities, "red codes", alarms and warnings, must be considered when scheduling the services execution.

Seamless-transparent management: The overall performance observed by the client (i.e., the service) must be completely transparent to the real resources utilized to deploy the service. In a dynamic scenario, where fogs and their characteristics are frequently set and torn down, a policy must be introduced to guarantee seamless performance for services (depending on the agreement signed). A management entity is then required to handle a correct sequence of actions defining the workflow of a particular service on each layer. A solution can be to set cloud controllers on each individual layer (extending the solution proposed for vehicular clouds in [5]), although this brings some issues associated with a process of abstraction.

QoS guarantees: In the IoT context, when artifacts like cloudlets need to be dynamically set, even on the move, the service quality is hard to impossible to guarantee. Therefore, strategies to achieve QoS (or near QoS) must be set to guarantee such quality, including, for example: i) resource availability guarantees by setting thresholds to define the setup and teardown for the different fog layers; ii) minimization of service response time by both selecting the "shortest" number of layers and generating the suitable workflow of actions to run a service; iii) improvement of reliability, by generating "protection" techniques to adequately handle tolerance to fog layers failures. Moreover, an important aspect to grant QoS, refers to the identification of the measures of performance bottlenecks. This issue is particularly important in the proposed F2C scenario due to the inherent dynamism and heterogeneity of the resources, be it data or physical infrastructure. In a nutshell, bottleneck may be caused by communication, I/O or storage constraints, compute power, etc., and the individual impact of each of these metrics and their combinations will depend on the particular context and service to be executed.

Security/data privacy: Security and privacy are well-known and currently unsolved challenges in the cloud arena, and remain as such also in F2C. Deploying fogs in fact exacerbates these issues, as already pointed out in [7] and also in [8] for vehicular clouds, mainly because of the inherent uncontrolled and out-of-surveillance operation mode for some edge devices. In the proposed F2C scenario, and just like in fog computing, security is envisioned as a key issue. Whether an F2C solution will commercially succeed, or not, depends on the adequate security and data privacy solution.

The value of any new framework and management architecture is best measured on the services it can support. In contrast to what has been addressed by fog computing concepts alone, where services are traditionally running sequentially at fog or cloud premises, see [7], we now show that F2C computing concept can actually optimize the performance. To this end, we now introduce in Figure 2 the use case of medical emergency to present some preliminary results aiming at illustrating how a specific service might benefit from parallel execution in the F2C architecture. Naturally, there are several more complex and complete scenarios where F2C might be widely deployed (e.g., real-time navigation, urban resilience, failure or quality control in the industrial sector), but the proposed medical emergency use case (as a well-known, highly demanding, simple and not far from the coming reality scenario) is just considered to set a proof-of-principle to position and highlight the F2C benefits.

Let us assume that a pedestrian walking in a city suffers a medical emergency. Let us also consider that the pedestrian embeds a device generating broadcast warnings, asking for medical assistance. The warnings may be collected by any of the cars in the immediate vicinity through the fog layers (cloudlets deployed in the city). It is imaginable that the embedded medical devices and, for instance, light signals can schedule the change of lights for the traffic to accommodate the emergency squad to faster come to the impacted human in medical need, together with the services running in parallel handling city control services and emergency medical

services discovery and location. The former will take over the actions to guarantee a fast and easy access to the person in medical emergency, such as tuning traffic lights, signaling streets, and computing short path for the ambulance. The latter will be responsible for localizing the nearest emergency service car, the nearest ambulance that can handle the condition discovered, or reporting the hospital from the ambulance about the patient status while the ambulance is in the way to the hospital, etc. This chain of actions (service workflows) can be designed to guarantee not only a fast response to the emergency, but also the response that best matches the particular type of emergency (personal injury, car accident, etc.). The different processes setting these service workflows are coherently and coordinately handled in parallel at premises located near to the event, including for instance, city management system, medical emergency management system, and nearby city infrastructure, such as roadside units (traffic lights), vehicular clouds (cars), etc.

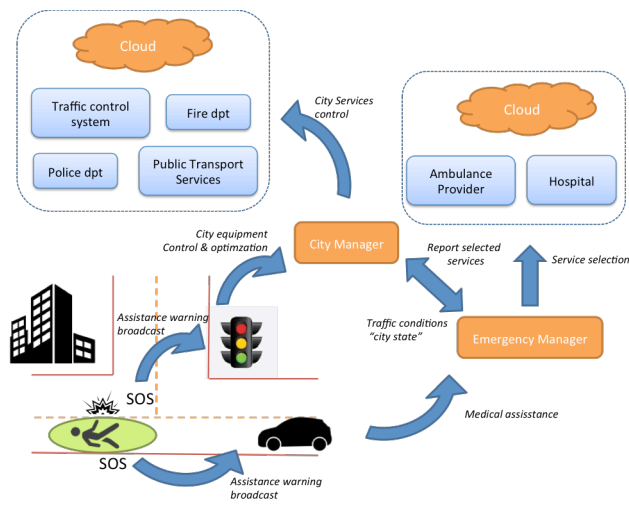


Fig. 2. Medical emergency scenario for F2C performance evaluation

III. F2C PERFORMANCE ANALYSIS OF A MEDICAL EMERGENCY SCENARIO

As it becomes obvious, the coordination among layers and the services composition, as well as their integration, they all require some degree of common control and management, as proposed in this paper. This section aims at illustrating how the fog-to-cloud architecture improves and optimizes both the execution time and the speed-up for a particular service, as compared to a traditional cloud scenario. To this end we propose to compare the performance of both a cloud and a fog-to-cloud scenario to manage all tasks linked to the medical emergency service described in Section III. The graphs in Figure 3 and Figure 4, show how the emergency service is parallelized for three different scenarios: i) cloud (i.e., the traditional cloud scenario excluding fog devices, shown to depict the F2C benefits); ii) fog-to-cloud (F2C), and; iii) optimized fog-to-cloud (OF2C). The parallelization has been obtained applying the Tareador tool [9], developed at the Barcelona Supercomputing Center (BSC). The tool is widely used in the High Performance Computing field to assist parallelization of sequential applications by decomposing sequential specification into tasks and dependencies facilitating parallel execution. The graphs depicted in Figure 3 and Figure 4 are included to easily show the differences in time execution (through the number of steps) for different parallelization options. The data included in the different blocks refers to the “task name” and “identifier”, as represented by the Tareador tool. Observing Figure 3 and Figure 4, we can easily conclude that: i) the degree of parallelization is at top 4, for any scenario, therefore no more than four different tasks may run in parallel, and; ii) different “layers” are shown. Finally, the parallelization degree is differently managed depending on the scenario; for a cloud scenario we consider potential parallelization at Data Center premises, i.e., number of processors (all tasks are executed within a DC), and for F2C and OF2C scenarios we consider different fogs as well as different levels of fogs.

The main objective is to compute the Execution Time (T) in terms of parallel processing units, ranging from 1, 2 and 4, standing for processors or fogs in cloud or cloud-to-fog scenarios respectively for the three envisioned scenarios.

a) Cloud Scenario: All service computation tasks in a cloud scenario (including data processing and connectivity) are handled at cloud premises). Looking at the graph in Figure 3, we see seven layers of execution actions, according to the Tareador tool. The first blue block in Figure 3 stands for the main process, giving room to the light green block at the first layer, responsible for cloud discovery, selection and connection. Four blocks are shown at the second level meaning for different tasks, red is “Get Profile”, yellow is “look for ambulances”, pink is “look for hospitals” and light blue is “call family”. Then, we see the two blocks at the third layer, brown and green standing for “select ambulance” and “select hospital” respectively. Next layer, fourth, includes three blocks,

light brow-green is “compute path” (to the hospital), purple is “inform hospital” and dark blue is “inform family with hospital”. The fifth layer shows two blocks, the green is “configure traffic lights” and the grey one is “warn heavy transport”. Three different blocks are generated at the sixth layer (forming a loop during the ambulance ride), the light blue is “loop during the path”, the light green is “communication with data center” and the blue is “computing and signaling”. Finally, the last layer, collects one single block, in yellow standing for the task “Inform hospital when ambulance is at 500m”. Different trials are carried out on the Tareador tool to run the service considering 1, 2 and 4 processors. To this end, we assume that: i) the ambulance is continuously connected to the cloud to inform about its position and hence enabling the cloud to process obtained data; ii) the time to discover and connect to the cloud is set to 1000; iii) the time required by the ambulance to communicate to the cloud is set to 100 (for each step of the loop); iv) ambulance path, traffic lights settings, warning, etc, are all handled at cloud premises, and; v) we do not consider the overhead of communication between processors for simulations including more than 1 processor.

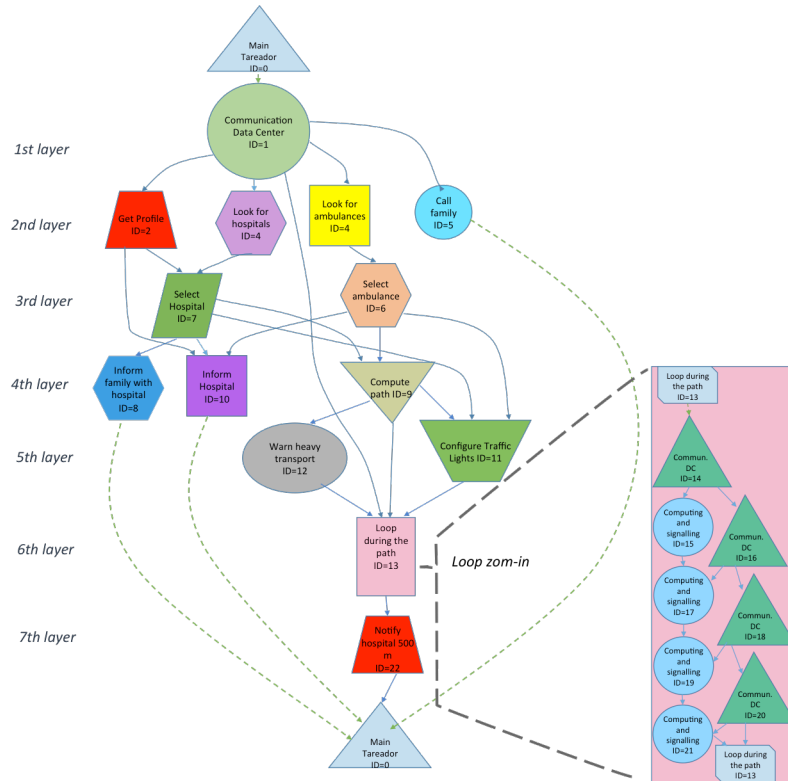


Fig. 3: Graph for cloud

Figure 5 shows the different execution times observed for the different options, using the Paraver tool [10]. The Paraver tool is a data browser, also developed at the Barcelona Supercomputing Center (BSC), to respond to the need to have a qualitative global perception of the application behavior by visual inspection. The tool simulates the service execution time using different number of processors. Figure 5 shows one processor performance per row, clearly drawing the time required for executing all processors’ tasks (in different colors corresponding to those used in the graphs shown in Figure 3 and Figure 4). We see that a notorious reduction is obtained when 4 processors are used, but as expected, using more than that will not improve the time, as shown by the different light blue colored slots representing inactivity gaps.

b) Fog-to-cloud scenario (F2C): This scenario is represented by the graph in Figure 4.a, considering the following assumptions: i) we assume as many fog layers as service layers (worst case); ii) the initial time of communication with the first fog is 10% of the time needed for the communication with a far cloud, and exactly the same that the communication time among the different fog layers, and; iii) the ambulance embeds fog premises, so avoiding the need to communicate to the cloud. The main differences with the cloud scenario, as shown in the graph, can be described as: i) initial discovery and communication to the closest fog (first layer), and; ii) “communication between fogs layers” task (grey-blue hexagon blocks, layers 3, 5, and 7) to facilitate task processing. We set different experiments considering distinct processing units, i.e., we consider 1, 2 and 4 fog devices per level. Due to space constraints we do not show the obtained plots but we bring together all results in Table 1. It can be seen that 8,6% time reduction (speed-up) can be obtained when considering 4 processing units.

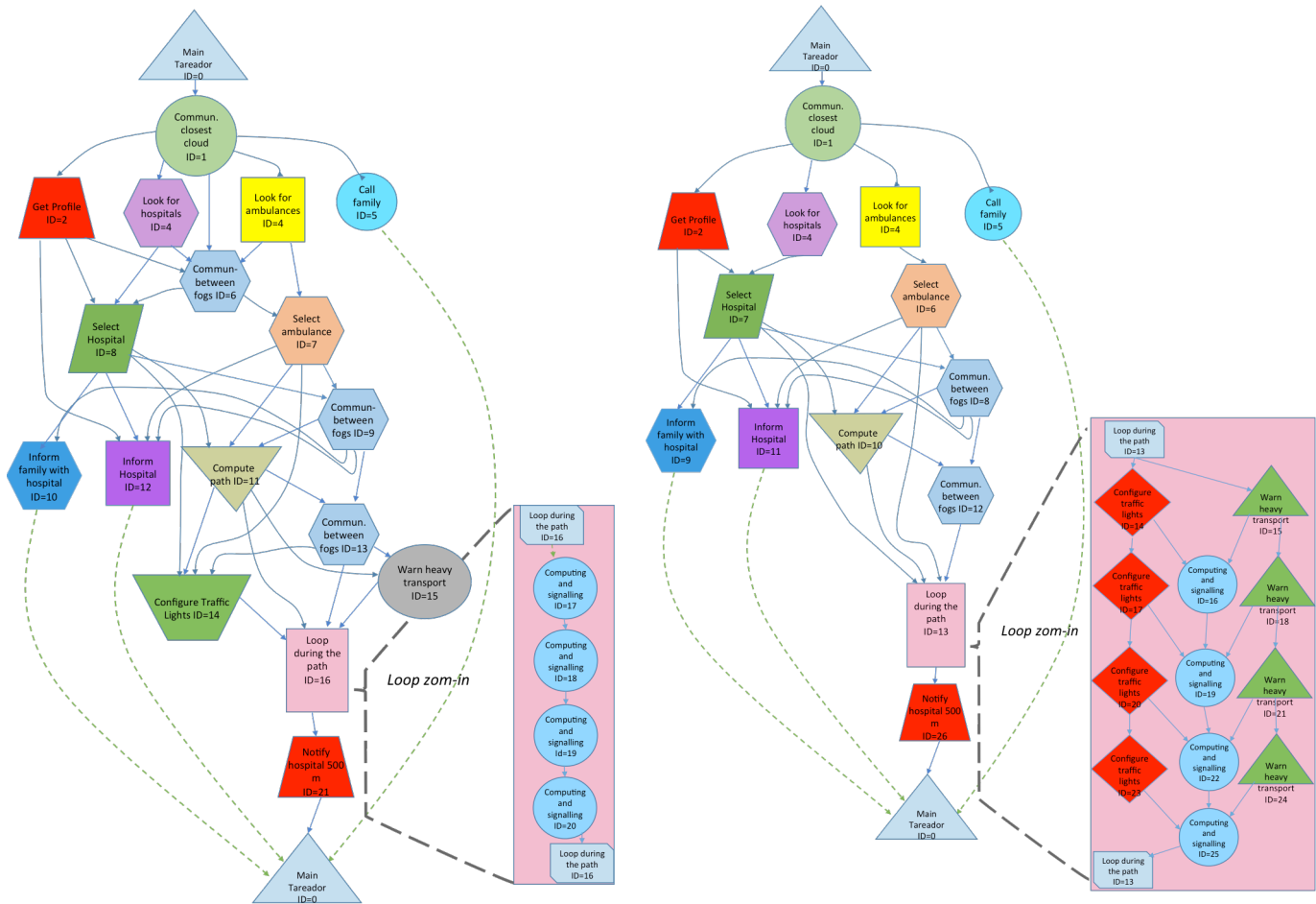


Fig. 4: Graph for a) fog-to-cloud (F2C); b) optimized fog-to-cloud (OF2C)

c) Optimized Fog-to-cloud scenario (OF2C): In the previous F2C scenario 4 steps of communication with or between fogs were considered. These are, the first one for discovering the closest fog (light green in the graph, first layer), and 3 more enabling communication between different cloud layers (grey-blue hexagon blocks, layers 3, 5 and 7). The OF2C scenario, shown in the graph in Figure 4.b, optimizes the F2C scenario by assuming two basic improvements. First, we consider that the number of required fog levels must not meet the number of task levels in the graph. This is rooted on the fact that fog resources have capacities enough to provide the processing required by the next task layer. This assessment is mapped into the graph by removing the “communication between fog layers” block (grey-blue hexagon block in F2C) in layer 3 in Figure 4.a, what is also removing the different created dependencies turning into a clear time reduction. In practical aspects this means that, in Figure 4.b the layer tasks responsible for “look for ambulances”, “look for hospitals”, “call family” and “get profile”, and the next layer responsible for “select hospital” and “select ambulance”, will run at the same fog premises, thus also removing the need for adding the “communication between fog layers” block. Second, we consider the two tasks linked to the ambulance ride are running within the loop. Indeed, those tasks at layer 8 in Figure 4.a, “warn heavy transport” and “configure traffic lights” are handled while the ambulance is on the way. As we did for the previous scenario we run several simulations on the Tareador tool, considering 1, 2 and 4 fogs per layer, to compute the execution time.

Table 1. Execution time and speed-up

Scenario	Time (ns)			Speed-up		
	1 process. unit	2 process. units	4 process. units (average)	Cloud-F2C	F2C-OF2C	Cloud-OF2C
Cloud	112,871,001	71,853,001	66,875,500	8,6%	21,51%	32,05%
F2C	91,500,001	62,450,001	61,537,000			
OF2C	77,926,001	57,762,001	50,644,501			

Results obtained for all scenarios are also presented in Table 1, showing execution times as well as the speed-up. A notorious 32,08% is obtained for OF2C vs the cloud scenario, hence assessing the impact of the proposed layered architecture. It should also be noticed that the time values shown when considering 4 processing units are average values computed after running 100 trials, in order to accommodate the randomness included by the “compute_path” function that randomly selects a path from a set of 20 candidates (indeed we can see different path lengths reflected in the total loop length in the graphs in Figure 2).

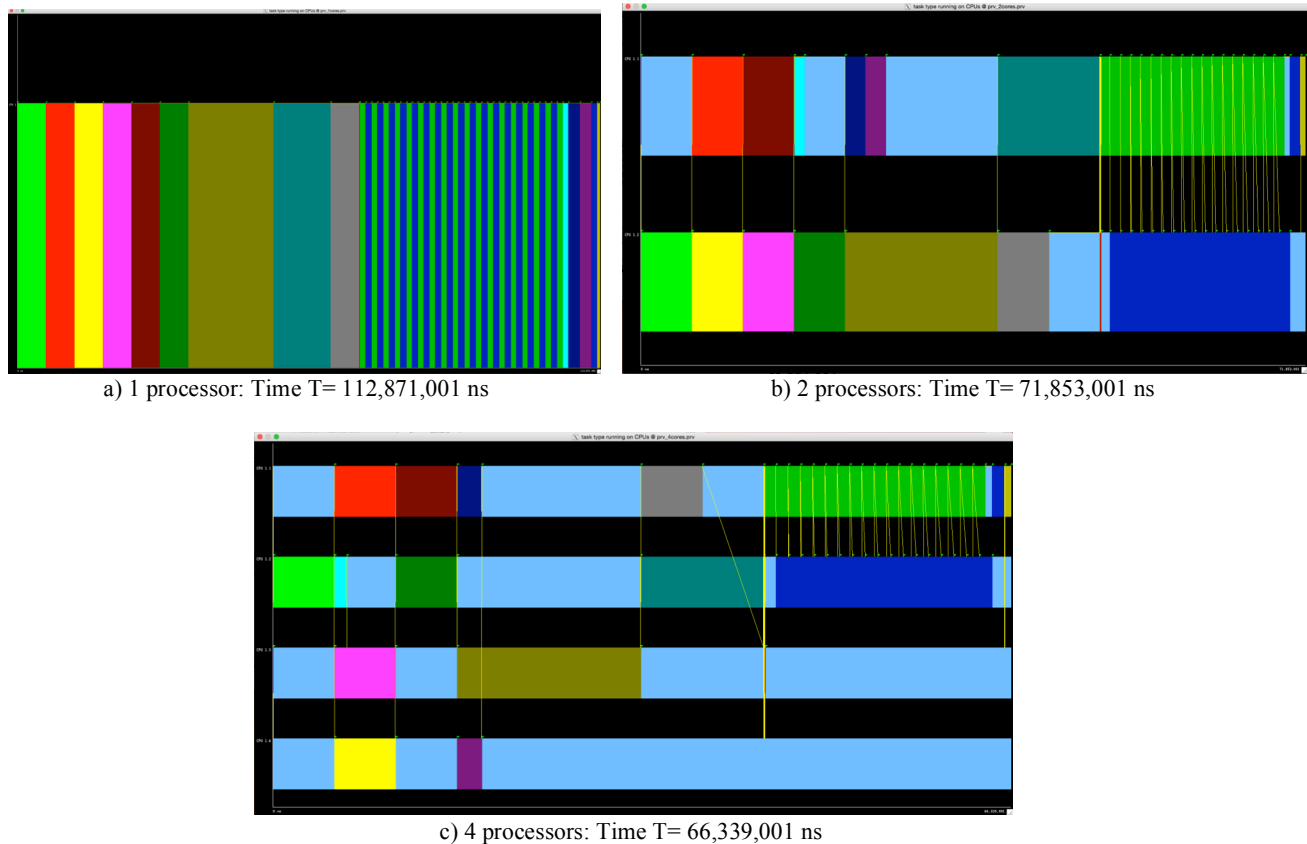


Fig. 5. Simulation results for the cloud scenario for 1, 2 and 4 processors

IV. CONCLUSION

As edge devices become richer in functionality and smarter, – embedding capacities such as storage or processing, as well as embedding new functionalities, such as decision making, data collection and forwarding, sharing, etc, a real need is emerging for coordinated management of fog-to-cloud (F2C) computing systems. This paper introduces a layered fog-to-cloud (F2C) architecture, its benefits and strengths as well as the arising open and research challenges, focusing on the need for the coordinated management. Our architecture, the illustrative use case presented and a comparative performance analysis, albeit conceptual, all clearly show the way forward towards a new IoT scenario with a set of existing and unforeseen services provided on a highly distributed and dynamic compute, storage and networking resources, bringing together heterogeneous and commodity edge devices, emerging fogs as well as conventional clouds. The different characteristics envisioned in F2C computing enable and also promote the deployment of different smart value-added services which is critical to the extension of the ITS services portfolio, based on aspects of dynamic customized clouds, performance through parallelization, and hierarchical layering for composite service execution.

From the technological perspective, the layered model defined in F2C computing demands a novel coordinated management strategy that will be driving new research challenges as identified in this paper. From the business perspective, the need for managing the F2C architecture is opening a new opportunity in the cloud industry and market segment. Challenges and opportunities reside in the novel set of potential applications as well as management of the F2C infrastructure. The novel set of applications and services are already emerging in the domains of e-health, citizen assistance, disaster management, urban planning, etc. Despite a growing number of application ideas, technical challenges need to be addressed in the areas of real time navigation, video surveillance, cooperative upload/download, real-time traffic control, optimized events control, bus routes profiled to clients, cache services (lower clouds

caching upper clouds layers), etc. The presented results showed the benefits of parallel services execution into the different F2C layers, illustrating one of a myriad of new research choices and system engineering innovations in future cloud computing.

ACKNOWLEDGMENTS

This work was supported, for UPC authors by the Spanish Ministry of Economy under contract TEC2012-34682 (partially funded by FEDER) and by the Catalan Government under contract 2014SGR371.

REFERENCES

- [1] L. Duan, L.D. Xu, "Business Intelligence for Enterprise Systems: A Survey", IEEE Transactions on Industrial Informatics, vol. 8, n^o. 3, August 2012
- [2] F. Bonomi, R. Milito, P. Natarajan, J. Zhu, "Fog Computing: A Platform for Internet of Things and Analytics", N. Bessis and C. Dobre (eds.), Big Data and Internet of Things: A Roadmap for Smart Environments, Studies in Computational Intelligence 546, DOI: 10.1007/978-3-319-05029-4_7, Springer International Publishing Switzerland 2014
- [3] D.Petcu, "Consuming Resources and Services from Multiple Clouds", Journal of Grid Computing, vol 12, issue 2, pp. 321-345, June 2014.
- [4] S.Wang, K.Yan, S.Wang, C.Chen, "A three-Phases Scheduling in a Hierarchical Cloud Computing Network", 2011 Third IEEE Int. Conference on Communications and Mobile Computing
- [5] R.Yu, Y.Zhang, S.Gjessing, W.Xia, K.Yang, "Toward Cloud-based vehicular networks with efficient resource management", IEEE Network, Vol. 13, pp. 48-55, Sep 2013
- [6] A.Bestavros, O.Krieger, "Toward an Open Cloud Marketplace", IEEE Internet Computing, Vol. 18 (1), pp.72-77, 2014
- [7] Ivan Stojmenovic, Sheng Wen, "The Fog Computing Paradigm: Scenarios and Security Issues", Proceedings of the 2014 Federated Conference on Computer Science and Information Systems pp. 1-8, 2014
- [8] G.Yan, D.Wen, S.Olariu, M.C.Weigle, "Security Challenges in Vehicular Cloud Computing", IEEE Trans. Intell. Transp. Syst., col 14, n^o. 1, pp. 284-294, March 2013
- [9] V.Subotic, A.Campos, A.Velasco, E.Ayguadé, J.Labarta, M.Valero, "Tareador: The Unbearable Lightness of Exploring Parallelism", 8th International Workshop on parallel Tools for High Performance Computing, Stuttgart, Germany, October 2014
- [10] Paraver tool at <http://www.bsc.es/computer-sciences/performance-tools/paraver/general-overview>, Version 4.5.8 (July 2015)

BIOGRAPHIES

Xavi Masip-Bruin (xmasip@ac.upc.edu) is an associate professor at Universitat Politècnica Catalunya, UPC. He received his B.Sc., M.Sc and Ph.D. degrees in Telecommunications Engineering from UPC in 1989, 1997 and 2003 respectively. In 2007, he founded the CRAAX lab at UPC. He has served as Chair and TPC Chair for several conferences as well as editor and guest editor for some journals. His current research interests focuses on network management, cloud & fog computing, ITS and IoT.

Eva Marin-Tordera (eva@ac.upc.edu) received M.Sc. degree in Physics from Universitat Barcelona, UB, in 1993, and Ph.D. degree in Computer Science from UPC in 2007. She is with the CRAAX lab from 2007. Her research focuses on applying predictive strategies for resources management in fog computing, also including smart vehicles, novel QoE/QoS routing paradigms and network management.

Ghazal Tashakor (tashakor@ac.upc.edu) is pursuing her Ph.D. degree at Universitat Politècnica de Catalunya, UPC. She received her B.Sc. in Computer Engineering from Islamic Azad University and M.Sc in Computer Science from Mid Sweden University in 2009 and 2013, respectively. Her research interests are in cloud and fog computing, as well as new in smart approaches for resource discovery and selection.

Admela Jukan (ajukan@tu-bs.de) received the Dipl.-Ing. degree from the Fakultet Elektrotehnik i Racunarstva, FER, the M.Sc. degree in information technologies from the Politecnico di Milano, and the Dr.Tech. degree in electrical and computer engineering from the Technische Universität Wien. She is Chair Professor at the Technische Universität Carolo-Wilhelmina zu Braunschweig, TUBS. She serves as an Associate Technical Editor of the IEEE Communications Magazine, Senior Editor of the IEEE Journal of Selected Areas in Communications and Co- Editor-in-Chief of the Journal on Optical Switching and Networking (Elsevier). She is an elected Distinguished Lecturer of the IEEE Communications Society, 2015-2017

Guang-Jie Ren (gren@us.ibm.com) joined IBM Almaden Lab in August 2009 and is currently leading Mobile Research Group. His work is currently focused on creating enterprise apps that address industry pain points with intuitive design, mobile technologies, advanced analytics and user empowerment. His prior work includes business analytics, enterprise architecture and business modeling. Trained as an industrial engineer with bachelor and master degrees from Tsinghua University in Beijing, Guang-Jie received his PhD in Engineering and Management from Cambridge University.

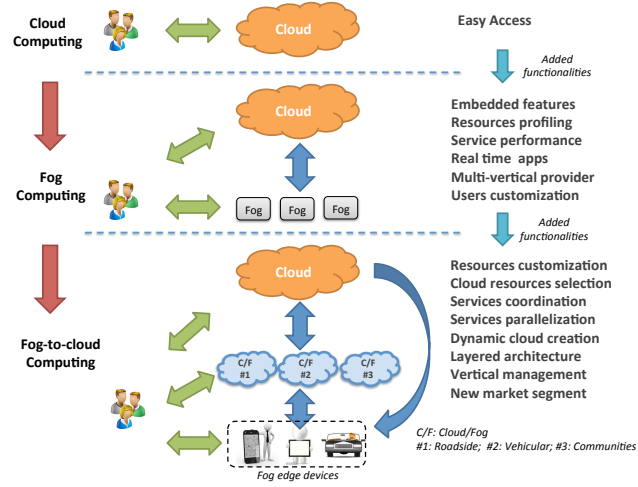


Figure 1. Cloud evolution towards fog-to-cloud computing

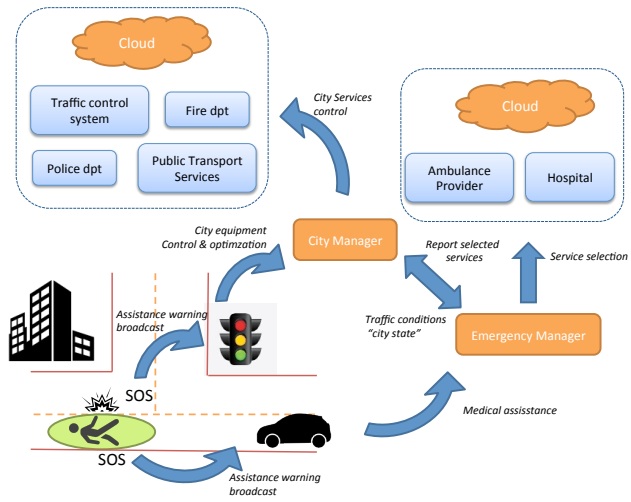


Figure 2. Medical emergency scenario for F2C performance evaluation

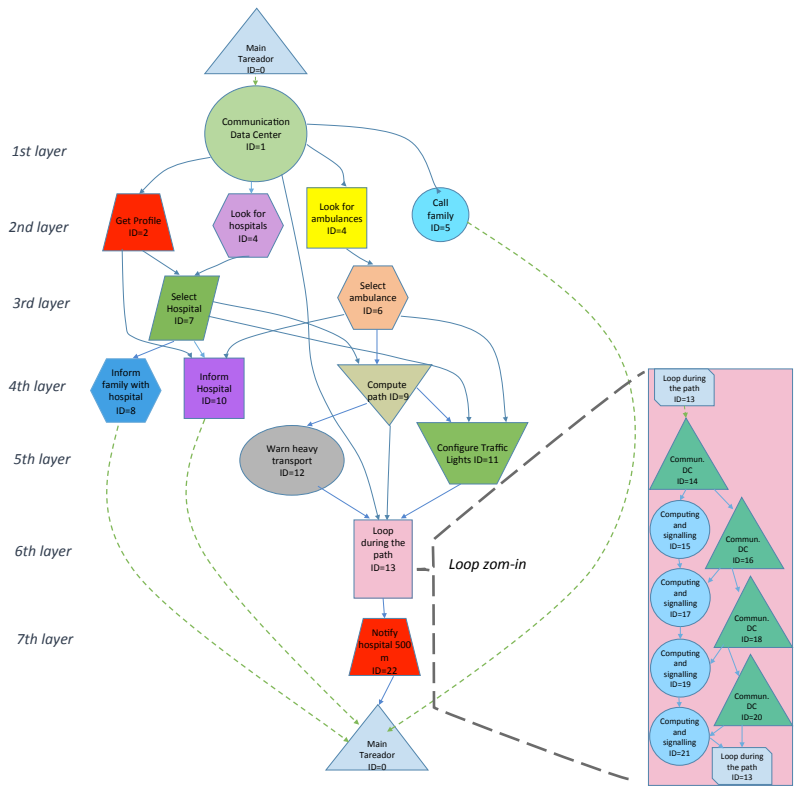


Figure 3. Graph for cloud

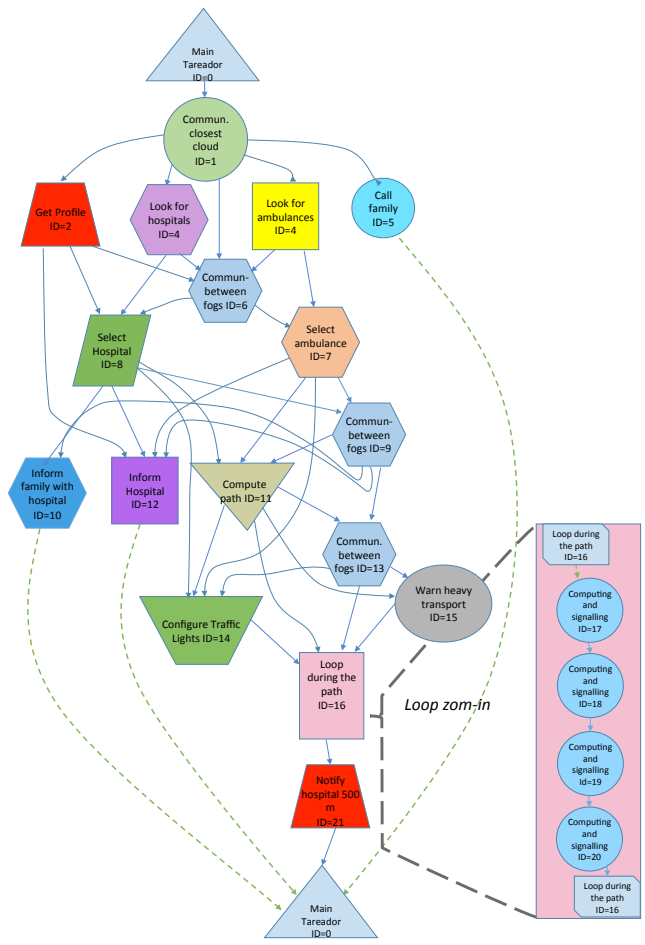


Figure 4-a. Graph for a) fog-to-cloud (F2C)

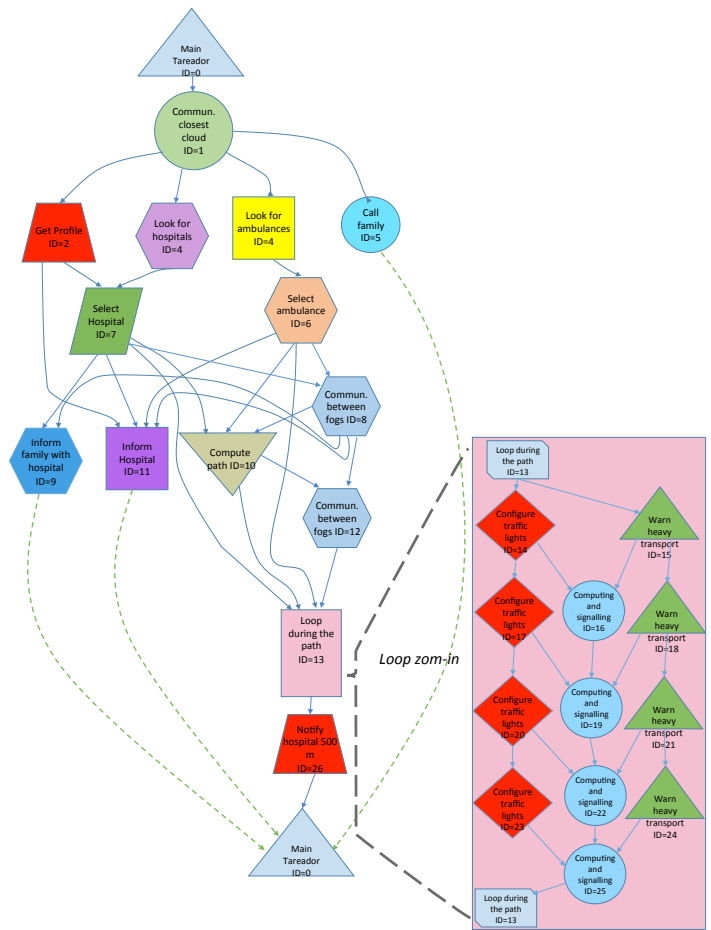
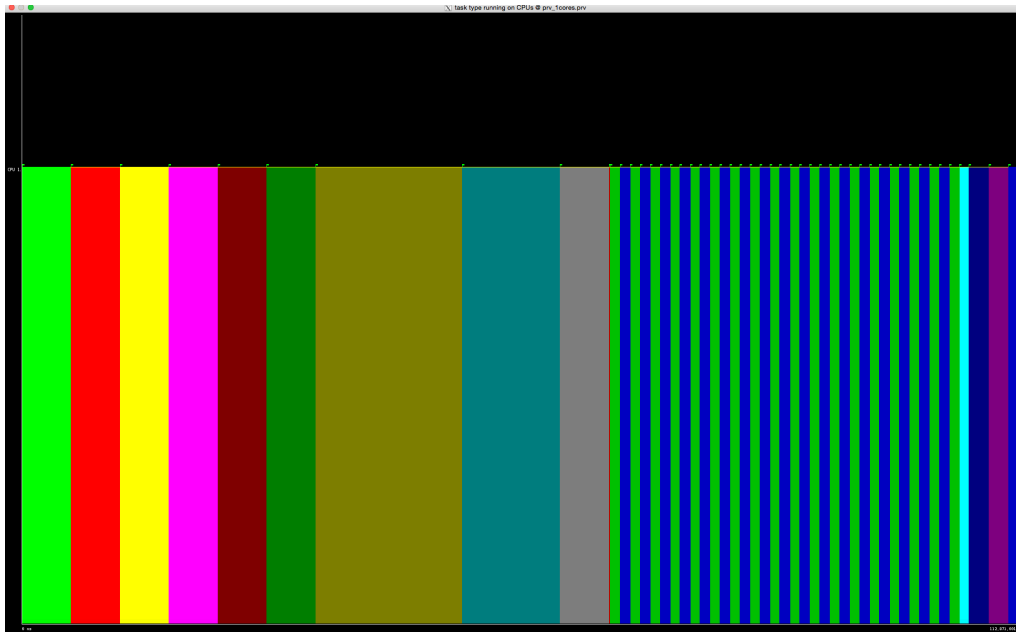
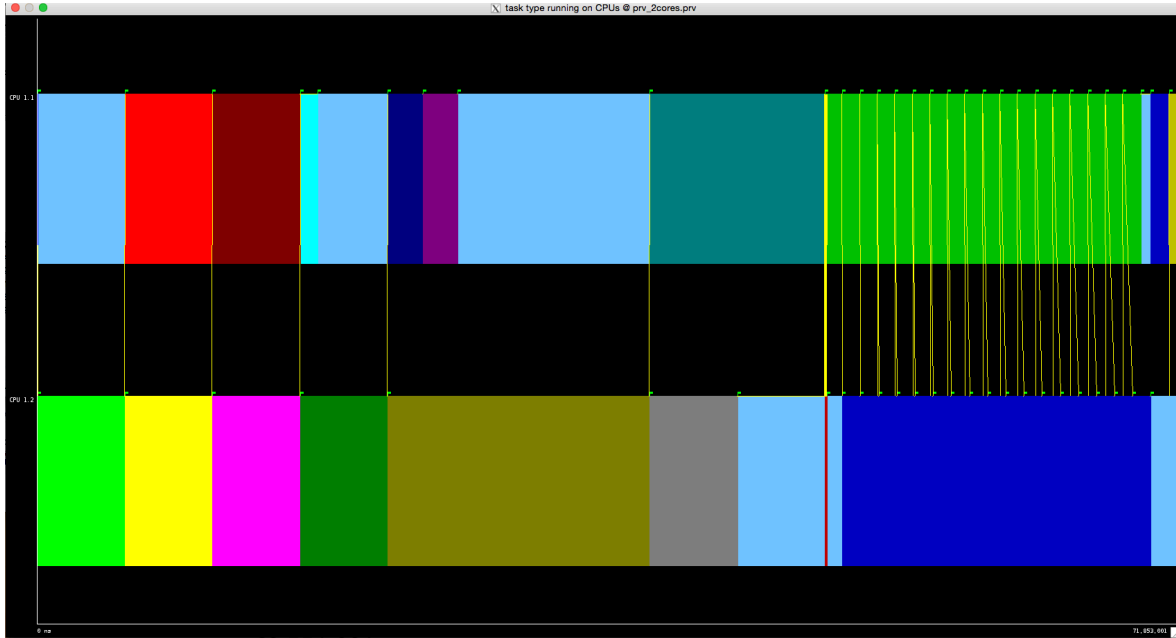


Figure 4-b. Graph for b) optimized fog-to-cloud (OF2C)



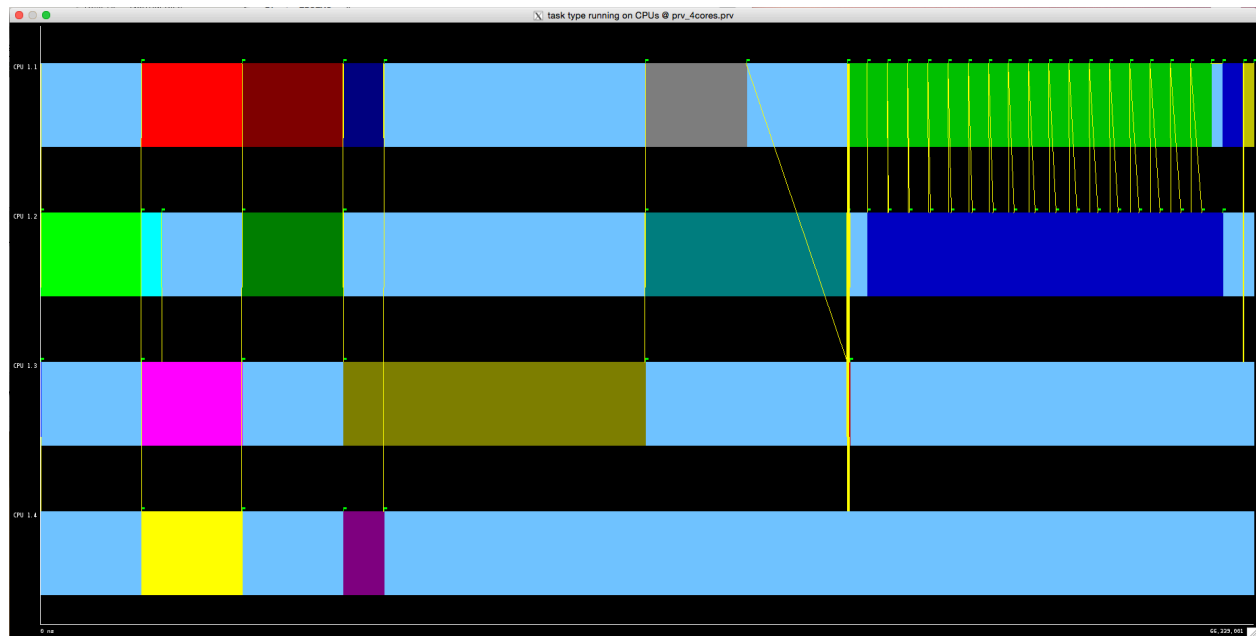
a) 1 processor: Time $T=112,871,00$

Figure 5-a. Simulation results for the cloud scenario for 1, 2 and 4 processors



b) 2 processors: Time T= 71,853,001 ns

Figure 5-b. Simulation results for the cloud scenario for 1, 2 and 4 processors



c) 4 processors: Time $T=66,339,001$ ns
Figure 5-c. Simulation results for the cloud scenario for 1, 2 and 4 processors