Exploring local service allocation in Community Networks

Davide Vega, Roc Meseguer
Computer Architecture Department
Universitat Politècnica de Catalunya - BarcelonaTech
C/ Jordi Girona, 1-3
08034 Barcelona, Spain
Email: {dvega, meseguer}@ac.upc.edu

Guillem Cabrera, Joan M. Marquès
Internet Interdisciplinary Institute
Universitat Oberta de Catalunya
C/ Roc Boronat, 117 7th floor
08018 Barcelona, Spain
Email: {gcabreraa, jmarquesp}@uoc.edu

***Abstract*—Community Cloud computing is a new trend on cloud computing that aims to build service infrastructures upon Wireless Community Networks taking advantage of underused community physical resources. Service allocation protocols are a key design challenge that all cloud systems must properly address to optimize resource utilization. They are specially important when cloud services requires a Quality of Service (QoS) and network stability or performance (delay, jitter, minimum bandwidth) cannot be guaranteed a-priory. This work presents a study that tries to understand how to address cloud service deployments in such scenario. In particular, we start proposing an allocation algorithm to find optimal solutions when there is a central authority that coordinates the process. These solutions optimize the communication cost in two ways: (1) minimizing the service overlay diameter and, (2) minimizing the coordination cost along the network. Based on the study of the algorithm and the experimental simulations, we study the variables that outcome optimal service allocations in detriment to other solutions. We verify these findings using data mining techniques. Researchers can take advantage of the simulation results and our observations to design more reliable distributed algorithms able to dynamically self-adapt to network changes.**

Keywords*—Service allocation, community cloud, wireless network

I. Introduction and motivation

Wireless Community Networks (WCN) showed up as a cutting-edge model of open communication infrastructure, as they offer low-cost but participatory connectivity to citizens. These novel infrastructures are constructed, operated, maintained and owned by the citizens themselves and bring closer information technologies to underdeveloped countries or isolated areas often out of the plans of traditional telecommunication operators. Many of them succeeded in the recent years across the world as non-profit actions, thanks to the progress of ad-hoc wireless technologies. Guifi.net [1], Athens Wireless Metropolitan Network [2], FunkFeuer [3], Seattle Wireless [4] and Consume [5] are the most fruitful exponents of community Internet access providers.

Through their openess, neutrality and the effort of their own members, Guifi.net [1] [1] has become the world's bigger independent wireless community network. Mainly deployed in Catalonia (Spain), it has more than 17,000 operative nodes and up to 30,000 km of links. It uses a growing pattern mostly based on planed radio-link deployment adapted to each region. As a result, the network topology contains many recognizable topology patterns also present in other community networks.

Beyond participants' initiatives to improve the Internet access or increase the network quality and security, most community members also devote efforts to build and deploy distributed and scalable applications. This useful applications (e.g. web or FTP servers, monitoring systems) are typically provided to other community members as an open-access service, often without any other form of recognition than promoting the community network usage. As a natural evolution of this model, some communities started to provide mechanisms that regulate and normalize how their members deploy services or contribute with their computational resources. In [6] authors adopted the term *Community Cloud Computing* to describe such contributory systems that provide community-based services upon Wireless Community Networks.

Several initiatives can be found in the literature of community-based clouds, like Cloud@Home [7], Nebulas [8], Contributory Communities [9] or Clommunity [10]. They all set the trend on the aggregation of user-donated computing resources in such a way they can be regarded as a platform to provide long-lived services, as happens in *traditional* cloud computing.

A common particularity of these scenarios is that cloud services coordination and performance computing are highly dependent on the diversity caused by the networking characteristics of their host being in a community network. Some examples of such diversity are: (a) network deployment based on geographic singularities rather than network QoS, (b) powerless and heterogeneous network – and computation – equipment or (c) wireless radio links with asymmetric quality of services.

This paper aims to explore the consequences of a local service allocation in a particular wireless community cloud, called Clommunity, such that minimizes the coordination cost of such services. In our proposal, service locality is defined as a service deployment which guarantees a solution overlay with minimum diameter. A typical service with this requirements is a redundant information backup. The coordination cost, which in Clommunity is associated to the use of nodes from different administrative zones, has been also minimized looking for placement solutions that traverses a minimum number of zones. In the previous example, is has also sense for the clients

if they want fast access to their backup content.

To evaluate the potential interest of our idea, we employed a topological snapshot from Guifi.net community network taken on April 2013. We assumed a central coordination entity that has knowledge about the WCN topology in real-time, which is not improvable due the Guifi.net monitoring agregation system. Then, we used an exploratory algorithm to find all the optimal and sub-optimal service overlay placement on the Guifi.net core network, namely the minimum sub-network diameter that traverses less zones in Guifi.net network backbone representation.

The results of this paper focus on the study of the selected deployment features, rather than providing a straightway implementable algorithm for a real-world scenario. The experimental results were also validated using wll-known machine learning techniques. Our main contribution is to provide results that can be easely generalized to build descentralized diameter and zone optimal algorithms for service deployments in Community Cloud Computing systems.

The rest of the paper is structured as follows. Next section (II) describes the experimental scenario we used to perform our study. In Section III we present an exploratory service allocation algorithm. We present and discuss the obtained results in Section IV. We then analyze in detail these results using machine learning techniques in Section V. Related researches to our proposal and analysis are listed and described in Section VI. Finally, Sections VII and VIII close the paper with the lessons learned and conclusions of our work.

II. System model

In this section we present the models used to carry out our simulations. The Guifi.net community network was modeled using the previous work presented in [11] [12] based on data collection and experimentation over the real network. Furthermore, the computational model is based on the Clommunity project specifications.

A. Network structure

Current community networks consist of a set of locations interconnected through mostly wireless equipment that users — different stake-holders such as individuals, companies, administrations or universities — must install and maintain in addition to its links, typically on building rooftops. These hardware devices can be either low power computers that provide some service to network devices connected to them or network devices that create network connections to other nodes, namely links. These nodes and links are organized under a set of mutually exclusive and abstract structures called administrative zones, which represent the geographic areas where nodes are deployed.

After long-term observations of the community members behaviour, we can also infer some properties that could assist us to build our model:

1) **Network planning**. New nodes are added to the network to cover needs or improve particular network quality. Therefore, new equipment or links only become operational after an intensive test period. Thus, it is guaranteed any two nodes would eventually be able to communicate unless a great variation of external conditions (e.g. weather, interferences, etc).
2) **Long-term equipment contributions**. Even though users are responsible to fund and maintain their devices, the operational cost is close to zero. Hence, users usually show no interest in withdrawing them unless there is an external and improvable reason (e.g. a legislative change).

For our purpose, both properties enables the static, unweight and bidirectional topologies proposed in [11] [12] as a realistic network model where nodes can always be reached by others. In such models, network topology was divided into two representation graphs called *base-graph* and *core-graph*.

The base-graph of Catalunya Guifi.net has a node if and only if (a) its location is marked as "Working" status; and (b) at least one of its operational devices is connected to another node device. Hence, it represents the whole network (core infrastructure and clients). The core-graph consists on a subgraph from the first one on which the terminal nodes of the graph (i.e. leaf nodes) were removed. So, it does not contain client nodes. Table I overviews the main properties of the base and core graphs that we used in our study.

Table I. Properties summary of the used network graphs

	nodes/edges	degree max/mean/min	diameter	zones
Base-Graph	8907/9047	495/2.03/1	26	129
Core-Graph	419/559	27/2.67/1	24	85

Figure 1 depicts the resulting zones partition graph of the Guifi.net core-graph. The size of each vertex is proportional to the number of Guifi.net locations contained in that zone, while the thickness of the edges stands for the number of physical links between nodes on both connected zones. The base-graph zones partition graph is not printed because we focused the service deployment on the core-graph only.

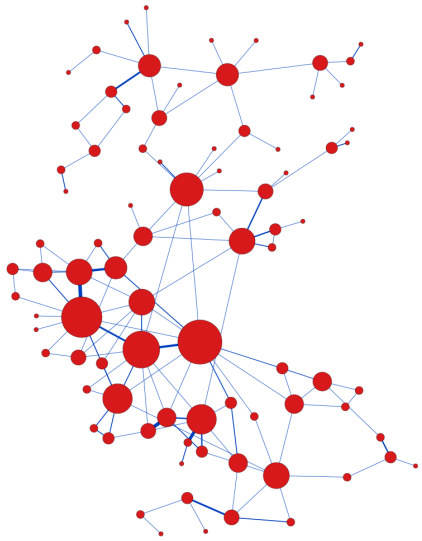

Figure 1. Guifi.net zones partition graph.

As the zones partition graph represents a geographic structure, it is not surprising that it contains some zones weakly connected – geographic innaccesible areas – and other with high degree of connectivity – surrounded by many other zones.

B. Clommunity architecture

Clommunity architecture is based on a hybrid peer-to-peer model with three hierarchical levels of responsability. On each level, their members are able to share information between themselves. However, their coordination is managed by some peer designated from the inmediate upper layer. Three types of peers can be easily identified:

1) **Resource nodes** are networking and computing equipment placed along the Wireless Community Network by users. Besides than contributing as any other "conventional" participant to the maintenance of the network quality and stability, they also share all or part of their physical resources to other community members in an infrastructure as a service (IaaS) fashion. Our model assumes all of these nodes have the same tye and amount of resource, in such a way that from the service point of view there is not allocation preference.
2) **Zone managers** are single nodes – only one within each zone – selected among all the *resource nodes* with the extra responsability to manage local zone services and coordinate inter zones agregated information. In practice, these managers are selected nodes with powerful computational capabilities and high availability.
   Our model does not explicitly identifies these managers and assumes the existance of at least one of them in each area. However, our model reckon with the coordination cost of placing services in computational resources belonging to different zones. Thence, our aim is to account and minimize the zone traversal of services.
3) **Controller** is a unique centraized entity in our system. It manages all the service allocation requests from the users and updates service structures by pulling the configuration information for the zone managers.

C. Allocation model

With the intention of generalizing the allocation model for community services, we made few assumptions that give to our model the flexibility to adapt to many different types of real services and conditions. We considered a service a set of S generic processes or replicas (with different roles or not) that interact or exchange information through the community network. Each of these replicas will be deployed over a node in the network, creating a service overlay graph with the selected nodes. Each of these nodes will host one and only one process, no matter which service it belongs to.

It is important to remark the services aimed in this work should be at infrastructure level, as cloud services in current dedicated datacenters. Therefore, these services would be only deployed over the core resources of the network (the ones in the core-graph) and accesed by the base-graph clients. One step further in the descentralization strategy would be to include user-hosted services, similarly as done in volunteer computing or in the Contributory Communities proposal.

Finally, our goal is to obtain service allocations that present the minimum possible distance in number of hops between the two furthest selected resources. To achieve this, we will only consider service overlays with no partitions on it, which will asure the elected nodes are as close as possible in the network topology.

III. Service allocation algorithm

We devised an algorithm that explores many different allocations seeking for local minimal diameter services leveraging the administrative split of Guifi.net in different zones (Algorithm 1).

The algorithm constructs different service allocations within each zone and angles for the local optimal by comparing them using the ISBETTER() method. This method evaluates the service diameter and, in the case of equal diameter allocations, the mean outdegree (understood as the mean boundary of the nodes in the service overlay with the nodes outside of it). The service allocation with smallest diameter and largest mean outdegree is kept as the optimal.

After that, it starts an iterative process of joining adjacent zones and applying the same search process on these aggregations, scavenging for more optimal allocations by combining nodes in several network areas. Once all the zones and zone compositions are evaluated, the best found service allocation (evaluated as in the zone search process) is chosen as the actual service allocation in the network.

Algorithm 1 Find optimal diameter service overlay

Require: $N(V_n,E_n)$ ▷ Network graph
Require: $Z(V_z,E_z)$ ▷ Zones graph
Require: S ▷ Number of nodes in the service
Require: L ▷ Number of zones to aggregate

```
1: bestSer ← null
2: for all V_{z_i} ∀i ∈ Z do
3:     bestSer ←EVALZONE(
           N, Z, V_{z_i}, V_n ∈ V_{z_i}, S, L, bestSer)
4: end for
5: optSer ← null
6: for all service ∈ bestSer do
7:     if ISBETTER(service, bestSer) then
8:         optSer ← service
9:     end if
10: end for
11: return optSer
```

Algorithm 1 relies on the method EVALZONE() to evaluate the different service allocations in a zone and generate the adjacent node aggregations, as sketched in Algorithm 2. The algorithm iterates through the nodes in that zone and selects the closest $S-1$ nodes in number of hops (using method MINDAROUND(). This allocation is then compared with the best allocation found so far for that given zone or zone aggregation and if it reduces the diameter or increases the outdegree, it is kept as the new optimal allocation. From now on we will refer to the solution that overlay optimizes diamter and the number of zones travesed as the *optimal solution*. In the same way we will refer to evaluated solutions that only optimizes the solution diameter as *sub-optimal solutions*.

The auxiliar MINDAROUND() method runs a Breadth-First-Search algorithm (BFS) in the network graph taking as root the given node and selects the first $S-1$ closest resources to it. In the case of several nodes at the same distance, nodes are selected randomly. Thanks to this feature, our algorithm performs faster than a pure exhaustive search procedure, since size equivalent allocations are not evaluated.

Algorithm 2 Find optimal diameter node set in a zone

Require: $N(V_n,E_n)$ ▷ Network graph
Require: $Z(V_z,E_z)$ ▷ Zones graph
Require: $zone$ ▷ Selected zone in Z
Require: $nodes$ ▷ Nodes in $zone$
Require: S ▷ Number of nodes in the service
Require: L ▷ Number of zones to aggregate
Require: $bestSer$ ▷ Map of best services for each zone

1: **procedure** EVALZONE($N, Z, zone, nodes, S, L, bestSer$)
2: $L \leftarrow L-1$
3: $bestSer[zone] \leftarrow null$
4: **for all** $node \in nodes$ **do**
5: $service \leftarrow$MINDAROUND($N, nodes, node, S$)
6: **if** ISBETTER($service, bestSer[zone]$) **then**
7: $bestSer[zone] \leftarrow service$
8: **end if**
9: **end for**
10: **if** $L > 0$ **then**
11: **for all** $neighbor$ of $zone \in Z$ **do**
12: $newNodeSet \leftarrow nodes + V_n \in neighbor$
13: $bestSer \leftarrow$EVALZONE($N, Z, neighbor, newNodeSet, S, L, bestSer$)
14: **end for**
15: **end if**
16: **return** $bestSer$
17: **end procedure**

Notice the same set of nodes might be obtained from different root nodes, since allocations in near network areas would involve the exact same nodes. This phenomena is caused by the randomness introduced in the MINDAROUND() method to determine the closest nodes to a given resource. We avoid re-evaluating these allocations by a cache mechanism. We also use a similar mechanism to avoid evaluating several times the same zone aggregation. However, we omitted both features in Algorithm 2 for simplicity.

Although we are aware this algorithm is not a fast heuristic to quickly find optimal allocations, its simplicity and the wide range of evaluated allocations allowed us to perform several studies over the network and so understand the traits of Guifi.net. Notice that given a first node as stating searh point, the algorithm only needs local information to find the sub-optimals overlays around her – using a well-known BFS search tree. Hence, if the Controller was able to properly select the initial zone manager – as the less computationally loaded zone – our algorithm would be easily implementable in a distributed manner.

IV. EXPERIMENTAL RESULTS

We used the search and selection methodology early presented in Section III to simulate the allocation of several services in Guifi.net. The challenge was to determine key features of the network and its nodes that could help us to design new heuristic framework for local service allocation in community networks.

Our first interest was to ascertain the minimal diameter that could be obtained when allocating services of different sizes and how zones were aggregated to achieve that. Figure 2 shows for each different service of n nodes (3, 5, 7, 10, 15 and 20) both (a) the diameter of the service overlay (as minimum, maximum and average values) and (b) the average number of zones traversed by each bucket of optimal solutions.

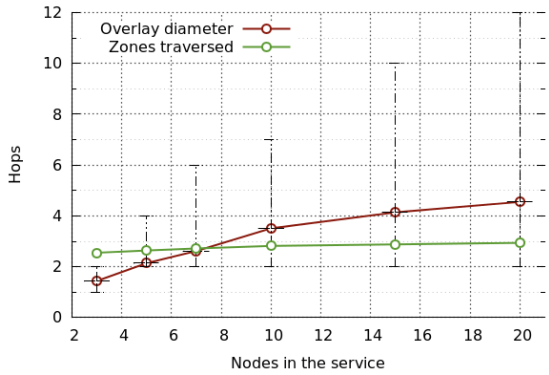

Figure 2. Obtained service diameter (max, min and average) and number of aggregated zones (average) for different optimal service overlay orders.

Notice that it is always possible to find an optimal service allocation with an overlay diameter of 2 hops within Guifi.net. Aditionally, the average solutions diameter increases as the number of nodes that composes the services does. It is also interesting to discover that the average number of aggregated zones considered to find the optimal allocation does not depend on the service size. Both behaviors indicate the existence of nodes with very high degree values, either within the same zone or between several zones in the network topology.

Next sections are focused on finding the patterns that identify these solutions and which nodes' properties lead to optimal service allocations in terms of overlay diameter and zone traversal.

A. *Effects of multiple zones traversal*

The Guifi.net split in administrative areas appeared to be a property to explode when allocating local services, in which seems reasonable to place the service replicas close to its final users. Therefore, we were interested on finding out how many nodes in the network were at one hop of any service replica when the final allocation was done across different network zones.

We plot in Figure 3 the cumulative probability function (CDF) of the number of nodes in the network base-graph that are at a single hop to any of the nodes in the optimal service overlay. Different zone agregation sizes are considered independently.

As one can logically expect, Figure 3 shows that the more zones are taken into account for resource selection, the more nodes are at one hop of any of the allocated replicas.

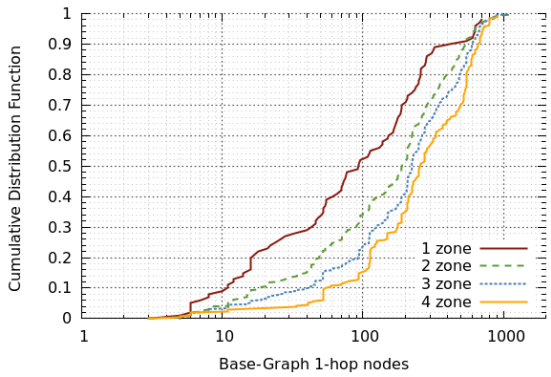

Figure 3. CDF of base-graph nodes at 1 hop to optimal solutions components, for 1, 2, 3 and 4 zones solutions.

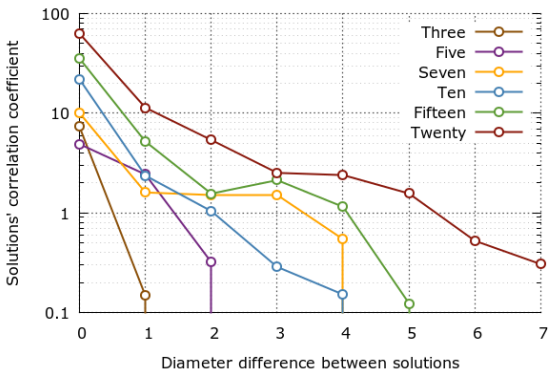

Figure 4. Solutions' correlation coefficient on diameter-adjacent service allocations

This result is very reasonable because the selected nodes are spread along geographic larger network areas and so, closer to many other nodes in the whole network. For instance, around 50% of the service allocations obtained considering only one administrative zone had about 100 nodes or less at one hop. However, 17% of the solutions reach the same limit of nodes at one hop if the considered resources are from four different zones. We did not expand the network aggregation to larger values due to the locality nature of the intended services: the more nodes regarded as the node election area, the further the service could end up placed, breaking the intended locality.

Other experiments were performed switching the diameter of the solutions and showed there is not a clear correlation between the number of nodes reachable at one hop and the sub-optimal solutions that minimizes the overlay diameter.

B. *Impact on dynamic deployment*

One of the intentions of this research is to discover patterns that could lead us to a robust service allocation mechanism for long-lived services in communitary environments. In such systems, it is likely that more than a few services should be handled, leading to a scarcity of resources to provide optimal allocations. Therefore, we inspected the similarity of the different possible service overlays under diverse circumstances, regarding the service size and the number of aggregated zones.

We defined the *correlation coefficient* for a set of solutions as the average percentage of common nodes after comparing each pair of solutions on the set. Figure 4 shows this correlation coefficient between optimal and sub-optimal solutions for six different node deployments. The horizontal axes stands for the diameter difference between the compared node selections. For example the diameter difference 1 corresponds to the comparison of all solution with diameter d (from 1 to 8) with their adjacent solutions of diameter d - *1* and d + *1*. The zero position stands for all the solutions with the same diameter.

The results show that the larger the service, the more likely the selected resources would be the same if we want to maintain the smallest service diameter. This is due to the fact the same nodes would be involved in all the allocations of minimal diameter because they are the ones with higher degree values. Therefore, very few services of large size (>10 nodes) could be supported if only the absolute optimal allocation is required. On the other hand, dealing with small services (<10 nodes) should be easier in this kind of networks. Notice that both observations are independent of the diameter difference between solutions.

Moreover, Figure 4 also shows that the correlation coefficient between two services decreases as their diameter difference increases. This fact implies that in a dynamic cloud, in which services are allocated and deallocated continuously, if we want to maximize the number of supported services, non-optimal overlays should be allocated to some cloud consumers in certain circumstances. Accordingly, we must introduce a tradeoff between services allocation optimization and the number of effectively allocated services.

Figure 5 shows the overlay correlation coefficient when aggregating all optimal and sub-optimal solutions found after evaluating 20-node service allocations. In each case, we ploted the diamter difference between solutions and the number of different zones traversed.

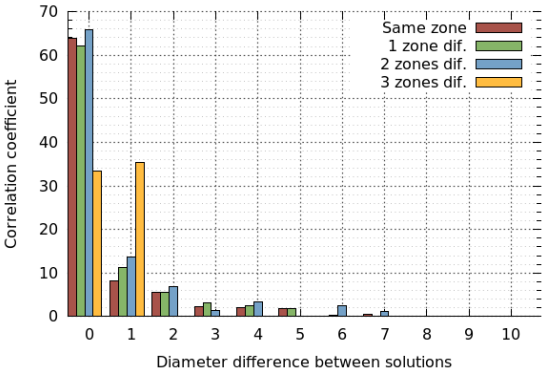

Figure 5. Solutions' correlation coefficient on diameter-adjacent and zones-adjacent for 20-node service allocations

Figure 5 indicates the same logical phenomena for network locality observed earlier. As long as the size of the adjacency diameter increases, the correlation coefficient between solutions decreases. Moreover, as a general rule, the agregation of solutions in zones shows that there is no significant impact. The artifact shown by the 2-diameter / 3-zones different solutions could suggest that there is a big exception to the general rule observed. However, it is caused by the concentration of

the space of solutions in two unique sets (1 and 2 adjacent solutions difference).

C. Solutions' dependency on social graph parameters

As we explained earlier, our algorithm searches in a large space of solutions, looking for those that minimize the overlay diameter. Identifying the properties of the chosen nodes that makes more likely to have a smaller diameter overlays would help us to design future efficient algorithms that reduces the comptutation time using greedy randomized heuristics.

As a first step in this direction, we decided to check the importance of nodes degree centrality metric. *Degree Centrality* is defined as the fraction of the nodes that a particular node is directly connected to. The nodes with higher degree centrality have more connections to the rest of the network and intuitevilly, are better candidates to build smaller degree sub-graphs.

Figure 6 exposes in a scatterplot the relationship between the average degree centrality of the resources in a service overlay and its diameter for service allocations. To ease the understanding of the figure, we plotted only services of 10, 15 and 20 components.

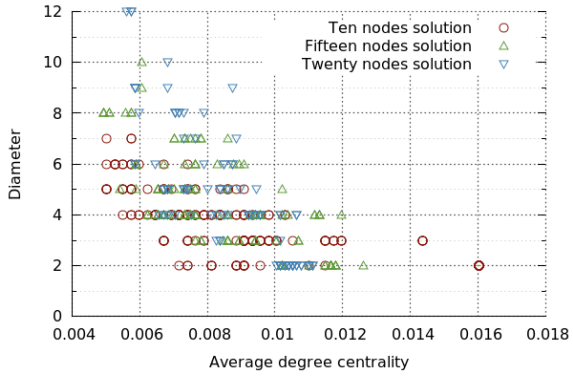

Figure 6. Service overlay diameter in function of the average degree centrality of the selected resources for 10, 15 and 20 nodes services

The analysis of Figure 6 shows that most of the solutions are concentrated on a small set of average centrality values (from 0.043 to 0.016). As a general rule, overlay networks that have higher diameter are composed by nodes with lower degree centrality. These results follow our intuition that the number of neighbours that a node is connected to defines their chances to easily reach the number of nodes needed to build the service.

Recall that on Section IV-A we discussed the independency between solution diameters and the number of nodes directly connected to the service overlay in the base-graph. That means the increment of the average degree centrality of optimal solutions can only be explained by the increase of the intra-solution connectivity in the overlay graph. As a result, smaller diameter solutions are also the most intra-connected ones. Hence, these solutions are more resilent to network failures and graph partitions.

A similar analysis performed using other social graph metrics (e.g. betweness centrality or nodes' degree) did not show a clear correlation. However, next section presents a comprehensive analysis on such solution components.

V. Analysis of Solutions Properties and Solutions Components

Feature selection is an important step to identify the components of a system that makes some results better or more desiderable than others. These techniques are specially useful when datasets have many variables and possibly correlated data, but it is necessary to first discard non-relevant variables [13]. This process, known as feature subset selection, uses machine learning techniques to reduce the space of possible solutions [14].

Our main objective is to reduce the properties of the nodes – as individuals, but also their collective ones – that are members of optimal service allocations. By clearly identifying them, we would able to improve our search algorithm in terms of elapsed time, communication cost and information needed.

We created six different datasets, one for each of the service sizes we simulated. The datasets are composed by all solutions evaluated by our Algorithm 1 (see method IsBetter() on line 7), no matter if they are optimal, sub-optimal or another minimum diameter solution not evaluated. Additionally to the diameter of the solutions and the number of zones travessed, we also evaluated the following features for each solution:

1) **isOptimal**. This feature is a boolean that indicates whether a given solution has been finally selected by our algorithm or it is an intermediate solution evaluated and discarted. It helps the algorithms to avoid false positives.
2) **Core-graph neighbors**. This feature counts the number of nodes at 1, 2 and 3 hops to the selected overlay in the core-graph to detect local communtiy properties that make some solutions better than others.
3) **Base-graph neighbors**. This feature counts the number of nodes at 1, 2 and 3 hops to the selected overlay in the core-graph. The idea is to filter dependencies between core-graph and base-graph communities.
4) **Nodes' degree**. This feature lists the degree of each node in the solution. It is intended to measure the fact that a more connected node is more likely to build a low diameter tree with the required number of nodes around her.
5) **Nodes' degree centrality**. This feature lists the sorted degree centrality of each node in the solution. Our simulation results have shown that the solution diameter depends on this parameter.
6) **Nodes' closeness centrality**. This feature lists the sorted closeness centrality of each node in the solution. It is defined as the inverse of the average distance to all other nodes and is a good measure on how efficient is a particular node to propagate information in the network.
7) **Nodes' betweness centrality**. This feature lists the sorted betweness centrality of each node in the solution. For a node n is defined as the sum of the fraction of all-pairs shortest paths that pass through n. Previous results demonstrated that solutions have a dependency of the *degree centrality* but not in betweness centrality. We are specilly interested on discovering the existence of any kind of equivalence.

A. Feature selection algorithms

We took advantage of the Weka machine learning framework [15] to execute two very well-known feature selection algorithms: (1) Correlation-based Features Subset Selection (CFS), and (2) ReliefF. The objective of both algorithms is to detect relevant features that we can after analyze to detect and use patterns to build better allocation algorithms. Next, we present a brief description of both algorithms:

Correlation-based Features Subset Selection (CFS): evaluates the worth of a subset of attributes by considering the individual predictive ability of each feature along with the degree of redundancy between them. The algorithm selects subsets of features that are highly correlated with the class, but having low correlation between them.

ReliefF: evaluates the worth of an attribute by repeatedly sampling an instance and considering the value of the given attribute for the nearest instance of the same and different class. The independent searching of this algorithm also provides a "relevance" weight to each feature, but it does not consider redundant or correlated results.

B. Selecting and analyzing features

We run the selection algorithms multiple times with different combination of objectives, meaning triying to "guess" the solutions with minimum diameter or the solutions marked as better. As we expected, both algorithms returned the *diameter* as an important feature to mark as best a particular solution, but not vice versa. This result is consistent with our overlay evaluation, as the minimum diameter is a necessary condition but not sufficient.

Another important difference between both algorithms is that ReliefF suggested particular nodes identifiers as a key feature. It was useful to check the correlation coefficient results presented in Section V, but it was useless to detect generic features. Hence, to reduce the ReliefF resulting feature set while keeping the optimal salient characteristics of the data, we removed the node identifiers as a trait. Then, both algorithms provided comparable information.

In order to minimize the diameter of the selected solutions, CFS proposes to use the minimum nodes' *degree centrality*, some random *degree centrality* values and the maximum nodes' *closeness centrality* as the key features. Nevertheless, to evaluate the best solution, it becomes more important use a combination of *degree*, minimum nodes' *closeness centrality* and nodes' the medium *betweeness centrality* features. ReliefF algorithm in contrast, selects a complicated and not connclusive combination of the previous values.

Using selected features and the experimental sets, we run the PART algorithm [16] to build a decision tree to classify all posible solutions and try to find optimal ones. The algorithm builds a tree with 41 different rules and is able to classify correctly 94.5767% of the solutions.

However, using only the first rule rovided by PART we are able to classify more than 50% of the solutions with a single error comparing only three items: (1) minimum closeness centrality, (2) maximum betweness centrality and (3) maximum degree. This restrictive combination of conditions is not applicable to select the first node of the optimal solutions to construct the overlay around it, but it will help us to filter similar candidates in the EVALZONE() procedure (see Algorithm 2).

VI. Related work

Service placement is a key function of cloud management systems. Typically, it is responsible of monitoring all the physical and virtual resources on a system and balance their load through the allocation, migration and replication of tasks. The very final goal is to guarantee the service level agreement with the user, as well as to optimize the cloud consumption cost. Although the problem has been studied in centralized scenarios for a long time, it is still a studied issue in distributed architectures where network QoS has an important impact on the computational performance.

Among others, in [17] authors proposed an optimal allocation solution for ambient intelligence environments using tasks replication to avoid network performance degradation. Replication introduces an extra computational cost and turns the network to transitory states where it is not operable, which defeats this kind of solutions on large-scale networks.

A recent work by Rius et.al. [18] proposes a two-level architecture to solve the scheduling placement problem in large-scale computing systems. The main idea behind, similar to those applied in Clommunity, is to divide the resources in smaller areas controlled by low-level schedulers (LS). LS monitor their resources and apply policies at fine-grained level. All LS are controlled by a higher-level scheduler (HS) that coordinates them to distribute the tasks in a balanced and efficient manner. However, even it is a very promising work, it still lacks of QoS guarantees.

As far as we known, there are only two researches in the literature that provides service placement in distributed clouds with network-aware capabilities. While in the first one [19] – tested by [20]– authors propose a selection algorithm to allocate resources for service-oriented applications, the second proposal [21] focuses on resource allocation in distributed datacenters.

Notice that all these proposals emphasize the network latency or the resources distance as an utility measurement of the service deployment quality. Intuitively, it makes sense for local services, because the closer the resources are – in terms of network distance – the fewer communication overhead our cloud service will suffer. We use the same metric in our evaluation, but we wanted to introduce the coordination assesment, understood as the extra overhead caused by communication between different groups of resources.

VII. Lessons learned

After the detailed analysis of our results and the machine learning study, we summarize the lessons learned about optimal solutions for service allocation on static community cloud scenarios and their characteristics.

Coordination cost. In our experimental model, the coordination cost has been revealed almost fixed between 2 or 3 zones traversal on optimal solutions. However, the number of zones over which a given service is deployed have a huge impact on

the number of clients that will have fast access to the service. Hence, new algorithms must take into account there is a trade-off between the coordination cost and their service availability and reachability in the network.

Solutions components and properties. A clear pattern has been also observed in the node features that conform the optimal allocations. They reveal the minimum degree centrality can be used to select the first node that compose the service. However, selecting the next nodes in a particular range of closeness and betweness centrality is specially useful to obtain more optimal overlays.

VIII. Conclusions and Future Work

Community networks provide a perfect scenario to deploy cloud platforms to share services and resources in a contributory manner to their members. Recent works were focused on designing techniques and scalable architectures to provide proper service allocation, but to the best of our knowledge, none of them studied the task coordination issues.

In this work we introduced an allocation algorithm that, far from being optimal in computation times, provides optimal overlay allocations without need to verify the whole solution space. We used this first implementation as a baseline to identify node traits in the optimal service allocations. Additionally, we discovered and discussed some general rules that will ease the decentralization of the algorithm in a dynamic environment.

In future research, we plan to develop and implement a decentralized version of the presented algorithm, taking advantage from the lessons learned about nodes' selection. Furthermore, we want to investigate cloud scenarios based on users contributed resources. Introducing such hetereogeneity and network dinamism will require to consider network partitions and nodes temporal availability. Moreover, other kind of services will need other overlay requirements, like geographic distribution or graph coverage. In other words, we plan to generate a fast response greedy allocation algorithm to provide network-awareness to multiple community cloud services deployment. Finally, we plan to test the whole system on the Clommunity platform.

IX. Acknowledgments

This work was partially supported by the Spanish MEC DELFIN TIN2010-20140-C03-01; the Community Networks Testbed for the Future Internet (CONFINE), Large-scale Integrating Project: FP7-288535; A Community networking Cloud in a box (Clommunity), Small or medium-scale Focused Research Project: FP7-317879; the Spanish Ministry of Science and Innovation (grant TRA2010-21644-C03); the CYTED-HAROSA Network (511RT0419); and the IN3-UOC doctoral program.

References

[1] "Open, Free and Neutral Network Internet for everybody," http://guifi.net/en.

[2] "Athens Wireless Metropolitan Network," http://www.awmn.net.

[3] "FunkFeuer free net," http://www.funkfeuer.at.

[4] "Seattle Wireless," http://www.seattlewireless.net.

[5] "Consume," http://www.consume.net.

[6] A. Marinos and G. Briscoe, "Community cloud computing," in *Proceedings of the 1st International Conference on Cloud Computing*, ser. CloudCom '09. Berlin, Heidelberg: Springer-Verlag, 2009, pp. 472–484. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-10665-1_43

[7] V. D. Cunsolo, S. Distefano, A. Puliafito, and M. Scarpa, "Cloud@home: bridging the gap between volunteer and cloud computing," in *Proceedings of the 5th international conference on Emerging intelligent computing technology and applications*, ser. ICIC'09. Berlin, Heidelberg: Springer-Verlag, 2009, pp. 423–432. [Online]. Available: http://portal.acm.org/citation.cfm?id=1802674.1802732

[8] A. Chandra and J. Weissman, "Nebulas: using distributed voluntary resources to build clouds," in *Proceedings of the 2009 conference on Hot topics in cloud computing*, ser. HotCloud'09. Berkeley, CA, USA: USENIX Association, 2009, pp. 2–2. [Online]. Available: http://portal.acm.org/citation.cfm?id=1855533.1855535

[9] D. Lázaro, "A middleware for service deployment in contributory computing systems," Ph.D. dissertation, Universitat Oberta de Catalunya, July 2011, http://dpcs.uoc.edu/lazaro/lazaro-thesis.pdf.

[10] Clommunity, "A Community networking Cloud in a box," http://clommunity-project.eu/, FP7 European Project 317879.

[11] D. Vega, L. Cerda-Alabern, L. Navarro, and R. Meseguer, "Topology patterns of a community network: Guifi.net," in *Wireless and Mobile Computing, Networking and Communications (WiMob), 2012 IEEE 8th International Conference on*, 2012, pp. 612–619.

[12] L. Cerda-Alabern, "On the topology characterization of guifi.net," in *Wireless and Mobile Computing, Networking and Communications (WiMob), 2012 IEEE 8th International Conference on*, 2012, pp. 389–396.

[13] M. A. Hasan, V. Chaoji, S. Salem, and M. Zaki, "Link prediction using supervised learning," in *In Proc. of SDM 06 workshop on Link Analysis, Counterterrorism and Security*, 2006.

[14] I. Guyon and A. Elisseeff, "An introduction to variable and feature selection," *J. Mach. Learn. Res.*, vol. 3, pp. 1157–1182, Mar. 2003. [Online]. Available: http://dl.acm.org/citation.cfm?id=944919.944968

[15] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten, "The weka data mining software: an update," *SIGKDD Explor. Newsl.*, vol. 11, no. 1, pp. 10–18, Nov. 2009. [Online]. Available: http://doi.acm.org/10.1145/1656274.1656278

[16] E. Frank and I. H. Witten, "Generating accurate rule sets without global optimization," in *Proceedings of the Fifteenth International Conference on Machine Learning*, ser. ICML '98. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1998, pp. 144–151. [Online]. Available: http://dl.acm.org/citation.cfm?id=645527.657305

[17] K. Herrmann, "Self-organized service placement in ambient intelligence environments," *ACM Trans. Auton. Adapt. Syst.*, vol. 5, no. 2, pp. 6:1–6:39, May 2010. [Online]. Available: http://doi.acm.org/10.1145/1740600.1740602

[18] J. Rius, F. Cores, and F. Solsona, "Cooperative scheduling mechanism for large-scale peer-to-peer computing systems," *Journal of Network and Computer Applications*, no. 0, pp. –, 2013. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1084804513000118

[19] A. Klein, F. Ishikawa, and S. Honiden, "Towards network-aware service composition in the cloud," in *Proceedings of the 21st international conference on World Wide Web*, ser. WWW '12. New York, NY, USA: ACM, 2012, pp. 959–968. [Online]. Available: http://doi.acm.org/10.1145/2187836.2187965

[20] M. Steiner, B. G. Gaglianello, V. Gurbani, V. Hilt, W. Roome, M. Scharf, and T. Voith, "Network-aware service placement in a distributed cloud environment," in *Proceedings of the ACM SIGCOMM 2012 conference on Applications, technologies, architectures, and protocols for computer communication*, ser. SIGCOMM '12. New York, NY, USA: ACM, 2012, pp. 73–74. [Online]. Available: http://doi.acm.org/10.1145/2342356.2342366

[21] M. Alicherry and T. V. Lakshman, "Network aware resource allocation in distributed clouds," in *INFOCOM, 2012 Proceedings IEEE*, 2012, pp. 963–971.