Bandwidth-aware Service Placement in Community Network Micro-Clouds

Mennan Selimi
UPC
Barcelona, Spain
mselimi@ac.upc.edu

Llorenç Cerdà-Alabern
UPC
Barcelona, Spain
llorenc@ac.upc.edu

Liang Wang
Cambridge University
Cambridge, UK
liang.wang@cl.cam.ac.uk

Arjuna Sathiaseelan
Cambridge University
Cambridge, UK
arjuna.sathiaseelan@cl.cam.ac.uk

Luís Veiga
INESC-ID / IST Lisboa
Lisbon, Portugal
luis.veiga@inesc-id.pt

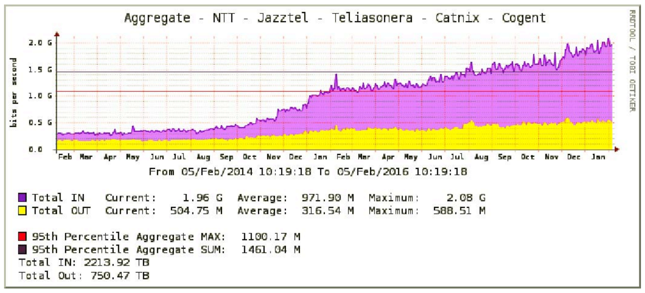
Felix Freitag
UPC
Barcelona, Spain
felix@ac.upc.edu

***Abstract*—Seamless computing and service sharing in community networks (CNs) have gained momentum due to the emerging technology of community network micro-clouds (CNMCs). However, deploying and running services in CNMCs confront enormous challenges to cope with, such as the dynamic nature of micro-clouds, limited capacity of nodes and links, asymmetric quality of wireless links, geographic singularity based deployment model rather than network QoS based, etc. CNMCs have been increasingly used by network-intensive services which exchange significant amounts of data between nodes, therefore their performance heavily relies on the available bandwidth resource in a network. This paper proposes a novel bandwidth-aware service placement algorithm which aims to replace the current random placement adopted by Guifi.net. Our experimental results show that the proposed BASP algorithm consistently outperforms the random placement in Guifi.net by 35% regarding its bandwidth gain. More promisingly, as the number of services increases, the gain tends to increase accordingly.**

***Index Terms*—service placement; community network;**

I. INTRODUCTION

Community networks (CNs) or Do-It-Yourself networks are built in a bottom-up and fully decentralized fashion, and are usually maintained by their own users. Early in the 2000s, CNs already gained momentum in response to the growing demands for network connectivity in rural and urban communities. One successful effort of such a network is Guifi.net, located in the Catalonia region of Spain. Guifi.net is defined as an open, free and neutral CN built by its members: citizens and organizations pool their resources and coordinate efforts to build and operate a local network infrastructure. Guifi.net was launched in 2004 and till today it has grown into a network of more than 30,000 operational nodes, which makes it the largest CN worldwide[1]. Figure 1 shows the evolution of total inbound and outbound Guifi.net traffic to the Internet for the last two years. Pink colour represents incoming traffic from Internet and yellow represents outgoing traffic. For two years, the traffic has tripled and peaks are as a result of a new users and bandwidth-hungry services in the network.



```
■ Total IN    Current:    1.96 G   Average:  971.90 M   Maximum:    2.08 G
■ Total OUT   Current:  504.75 M   Average:  316.54 M   Maximum:  588.51 M

■ 95th Percentile Aggregate MAX:  1100.17 M
■ 95th Percentile Aggregate SUM:  1461.04 M
Total IN: 2213.92 TB
Total Out: 750.47 TB
```

Figure 1. Guifi.net inbound and outbound traffic

Similar to other CNs, Guifi.net aims to create a highly localized digital ecosystem. However, the predominant usage we have observed, is to access cloud-based Internet services external to a CN. For instance, more than 50% of user-oriented services consumed in Guifi.net go through gateway proxies which provide Internet connectivity hence impose a heavy burden on the limit backbone links [1]. For a very long time in the past, user-oriented services had not been developed locally because of the lack of streamlined mechanisms to exploit all the available resources within a CN as well as other technological barriers [2]. With the adoption of *community network micro-clouds*[2], i.e. the platform that enables cloud-based services in CNs, local user-oriented services gained a huge momentum. CN users started creating their own homegrown services and using alternative open source software for many of today's Internet cloud services, e.g., data storage services, interactive applications such as Voice-over-IP (VoIP), video streaming, P2P-TV, and etc [3]. In fact, a significant number of services were already locally deployed and run within Guifi.net including GuifiTV, Graph servers, mail and game servers etc. All these services are provided by individuals, social groups, small non-profit or commercial service providers.

Because Guifi.net nodes are geographically distributed, given this set of local services, we need to decide where these services should be placed in a network. Obviously, without

[1]http://guifi.net/

[2]http://cloudy.community/

DOI 10.1109/LCN.2016.47

taking into account the underlying network resources, a service may suffer from poor performance, e.g, by sending large amounts of data across slow wireless links while faster and more reliable links remain underutilized. Therefore, the key challenge in CN micro-clouds is to determine the location of deployment [4], i.e. servers at certain geographic points in the network, with the different services multiplexed on a shared infrastructure. Although conceptually straightforward, it is challenging to calculate an optimal decision due to the dynamic nature of CNs and usage patterns. In this work we aim to address the following question: *"Given a community network cloud infrastructure, what is an effective and low-complexity service placement solution that maximises end-to-end performance (e.g., bandwidth)?"* Our preliminary results show that the proposed algorithm consistently outperforms the current random placement adopted in Guifi.net by 35% regarding its bandwidth gain. More promisingly, as the number of services increases, the gain tends to increase accordingly.

The rest of the paper is organized as follows. Section II defines our system model and presents the bandwidth-aware placement algorithm. In Section III we discuss the evaluation results. Section IV describes related work and section V concludes and discusses future research directions.

II. Bandwidth-Aware Placement

The deployment and sharing of services in CNs is made available through *community network micro-clouds* (CNMCs). The idea of CNMC is to place the cloud closer to community end-users [5], so users can have fast and reliable access to the local services. To reach its full potential, a CNMC needs to be carefully deployed in order to utilize the available bandwidth resources.

Currently, the service deployment (much as network deployment) at Guifi.net is not centrally planned but initiated individually by CN members. Public, user and community-oriented services are placed randomly on supernodes, and user's premises respectively. Hence there are imbalances on the service placement, fluctuations and inefficiencies in service performance.

Currently used *organic (random) placement scheme* in Guifi.net is not sufficient to capture the dynamics of the network and therefore it fails to deliver the satisfying QoS. The strong assumption under random service placement, i.e., uniform distribution of resources, does not hold in such environments. Furthermore, in Guifi.net the wireless links are with asymmetric quality for services (30% of the links have a deviation higher than 30%) and we observed a highly skewed traffic pattern and bandwidth distribution [6]. Based on this, our goal is to design a bandwidth-aware service placement algorithm that will improve the service quality and network performance by optimizing the usage of scarce resources in CNs such as bandwidth.

A. Formulation and Notations

We call the CN the *underlay* to distinguish it from the *overlay* network which is built by the services. The underlay network is supposed to be connected and we assume each node knows whether other nodes can be reached (i.e., next hop is known). We can model the underlay graph as: $G \leftarrow (OR, L)$ where OR is the set of outdoor routers present in the CNs and L is the set of wireless links that connects them.

Let f_{ij} be the bandwidth of the path to go from node i to node j. We want a partition of k clusters: $S \leftarrow S_1, S_2, S_3, ..., S_k$ of the set of nodes in the mesh network. The cluster head i of cluster S_i is the location of the node where the service will be deployed.The partition maximizing the bandwidth from the cluster head to the other nodes in the cluster is given by:

$$\arg\max_S \sum_{i=1}^{k} \sum_{j \in Si} f_{ij} \tag{1}$$

B. Proposed Algorithm: BASP

We designed a bandwidth-aware algorithm that allocates services taking into account the bandwidth of the network. We monitored a production CN over five months such as QMP (i.e., subset of Guifi network in the city of Barcelona). We took a network snapshot (capture) from QMP network regarding the bandwidth of the links[3]. Our bandwidth-aware service placement algorithm BASP (see Algorithm 1) runs in three phases.

(i) Initially, we use the naive k-means partitioning algorithm in order to group nodes based on their geo-location. The idea is to get back clusters of locations that are close to each other. The k-means algorithm forms clusters of nodes based on the Euclidean distances between them, where the distance metrics in our case are the geographical coordinates of the nodes. In traditional k-means algorithm, first, k out of n nodes are randomly selected as the cluster heads (centroids). Each of the remaining nodes decides its cluster head nearest to it according to the Euclidean distance. After each of the nodes in the network is assigned to one of k clusters, the centroid of each cluster is re-calculated. Grouping nodes based on geo-location is in line with how Guifi.net is organized. The nodes in Guifi.net are organized into a tree hierarchy of *zones* [7]. A zone can represent nodes from a neighborhood or a city. We use k-means with geo-coordinates as an initial heuristic for our algorithm.

(ii) The second phase of the algorithm is based on the concept of finding the cluster heads maximizing the bandwidth between them and their member nodes in the clusters S_k formed in the first phase. The bandwidth between two nodes is estimated as the bandwidth of the link having the minimum bandwidth in the shortest path. The cluster heads computed are the candidate nodes for the service placement. This is plotted as Naive K-Means in the Figure 2.

(iii) The third and last phase of the algorithm includes reassigning the nodes to the selected cluster heads having the maximum bandwidth, since the geo-location of nodes in the clusters formed in the phase one is not always correlated with their bandwidth. This way the clusters are formed based on nodes bandwidth. This is plotted as BASP in the Figure 2.

Regarding computational complexity, the naive brute force method can be estimated by calculating the *Stirling number of the second kind* [8] which counts the number of ways to partition a set of n elements into k nonempty subsets, i.e.,

[3]http://tomir.ac.upc.edu/qmpsu/index.php?cap=56d07684

Algorithm 1 Bandwidth-aware Service Placement (BASP)

```
Require: $G(V_n, E_n)$                          ▷ Network graph
    $S \leftarrow S_1, S_2, S_3, ..., S_k$       ▷ $k$ partition of clusters
    $bw_i$                                       ▷ bandwidth of node $i$

 1: procedure PERFORMKMEANS($G, k$)
 2:     return $S$
 3: end procedure
 4: procedure FINDCLUSTERHEADS($S$)
 5:     $clusterHeads \leftarrow list()$
 6:     for all $k \in S$ do
 7:         for all $i \in S_k$ do
 8:             $bw_i \leftarrow 0$
 9:             for all $j \in setdiff(S, i)$ do
10:                 $bw_i \leftarrow bw + estimate.route.bandw(G, i, j)$
11:             end for
12:             $clusterHeads \leftarrow \max bw_i$
13:         end for
14:     end for
15:     return $clusterHeads$
16: end procedure
17: procedure RECOMPUTECLUSTERS($clusterHeads, G$)
18:     $S\prime \leftarrow list()$
19:     for all $i \in clusterHeads$ do
20:         $cluster_i \leftarrow list()$
21:         for all $j \in setdiff(G, i)$ do
22:             $bw_j \leftarrow estimate.route.bandw(G, j, i)$
23:             if $bw_j$ is best from other nodes $i$ then
24:                 $cluster_i \leftarrow j$
25:             end if
26:             $S\prime \leftarrow cluster_i$
27:         end for
28:     end for
29:     return $S\prime$
30: end procedure
```

$\frac{1}{k!}\sum_{j=0}^{k}(-1)^{j-k}\binom{n}{k}j^n \Rightarrow \mathcal{O}(n^k k^n)$. However, for BASP, finding the optimal solution to the k-means clustering problem if k and d (the dimension) are fixed (e.g., in our case $n = 54$, and $d = 2$), the problem can be exactly solved in time $\mathcal{O}(n^{dk+1}\log n)$, where n is the number of entities to be clustered. The complexity for computing the cluster heads in phase two is $\mathcal{O}(n^2)$, and $\mathcal{O}(n)$ for the reassigning the clusters in phase three. Therefore, the overall complexity of BASP is $\mathcal{O}(n^{2k+1}\log n)$, which is significantly smaller than the brute force method.

III. Algorithmic Behaviour & Performance

Solving the problem stated in Equation 1 in brute force for any number of N and k is NP-hard. For this reason we came up with our heuristic. Initially we used k-means algorithm for a first selection of the clusters. Then, we limit the choice of the cluster heads to be inside the sets of clusters obtained using k-means. Inside these clusters we computed the cluster heads having the maximum bandwidth to the other nodes. This is our baseline and is plotted in the graphs as *Naive K-Means* algorithm. Although we have the results, we do not consider the pure random placement actually used in Guifi.net since it is a very naive and not realistic approach for comparison (i.e., the results are much worse and we are not showing in the graphs).

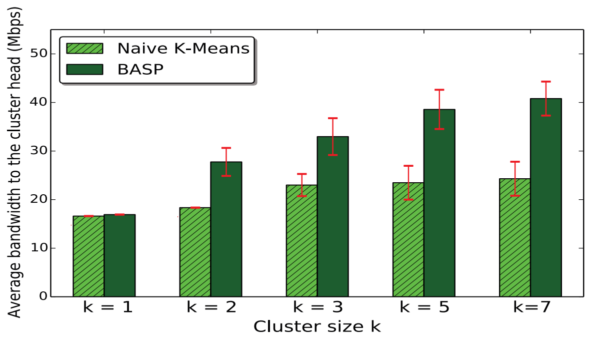


Figure 2. Average bandwidth to the cluster heads

To emphasise the importance of phase two and three, in this section we compare *BASP* to *Naive K-Means*. BASP groups nodes according to their bandwidth. The nodes assigned to a selected cluster heads in the phase two, are re-assigned in phase three (e.g., BASP) and clusters are formed according to their bandwidth and not geo-location.

Our experiment is comprised of 5 runs and the presented results are averaged over all the runs. Each run consists of 15 repetitions. Figure 2 depicts the average bandwidth to the cluster heads obtained with *Naive K-Means* algorithm and our *BASP* algorithm. Figure reveals that for any number of k, our *BASP* algorithm outperforms the *Naive K-Means* algorithm. For k=2 the average bandwidth to the cluster head is increased from 18.3 Mbps (obtained with Naive K-Means) to 27.7 Mbps (obtained with our BASP algorithm) i.e., 40% increase. The biggest increase of 50% is when k=7. Based on the observations from the Figure 2, the gap between two algorithms is growing as k increases. K increases as network grows.

Note that our heuristics enables us to select nodes (cluster heads) that provide much higher bandwidth than any other random or naive approach. But, if we were about to look for the optimum bandwidth within the clusters (i.e., optimum average bandwidth for the cluster), then this problem would end up to be an NP-hard. Finding the solution is NP-hard, because finding the optimum entails running our algorithm for all the combinations of size k from a set of size n . This is a combinatorial problem that becomes intractable even for small sizes of k or n (e.g., $k = 5$, $n = 54$). For instance, if we would like to find the optimum bandwidth for a cluster of size k=3, then the algorithm would need to run for every possible (non repeating) combination of size 3 from the set of size 54. That is for 54 nodes we would end up having 25K combinations ($choose(54,3)$), or 25K possible nodes to start with. We managed to do this and the optimum average bandwidth obtained for $k = 3$ was 62.7 Mbps. The optimum bandwidth obtained for $k = 2$ was 49.1 Mbps, and for $k = 1$ was 16.9 Mbps. However, the computation time took very long in a machine with Intel Atom N2600 CPU and 4 GB of RAM (65 hours for $k = 3$, 30 minutes for $k = 2$ etc.), comparing to BASP where it took 23 seconds for $k = 3$ and 15 seconds for $k = 2$. To summarize, BASP is able to achieve good bandwidth performance with very low computation complexity.

IV. Related Work

Data centers: Choreo [9] is a measurement-based method for placing applications in the cloud infrastructures to minimize an objective function such as application completion time. Volley [10] is a system that performs automatic data placement across geographically distributed datacenters of Microsoft.

Distributed Clouds: The work in [11] proposes efficient algorithms for the placement of services in distributed cloud environment. The algorithms need input on the status of the network, computational resources and data resources which are matched to application requirements. In [12] authors propose a selection algorithm to allocate resources for service-oriented applications and the work in [13] focuses on resource allocation in distributed small datacenters. Some recent work focuses on service migration in the distributed clouds. The authors in [14] study the dynamic service migration problem in mobile edge-clouds that host cloud-based services at the network edge.

Our context is a real CN such as Guifi.net and this differs from traditional data centers (cluster) environments. Devices and the network in data center are homogeneous whether in our case are very heterogeneous. In terms of demand distribution, in clusters there are load balancers whether in CNs the demand comes directly from the edge so there are no central load balancers. Topology is different, in clusters fat tree or leaf-spine topology is used whether in CNs we have a mesh topology. Underlying physical layer in clusters is wired whether in CNs it is wireless. Clusters are more reliable and robust to failures and CNs are not.

V. Conclusion

The prime motivation of the paper was to assess the current service placement in effect in a representative Guifi.net CN, analyse its inefficiencies, and propose and evaluate a feasible and effective solution to improve it. CNs provide a perfect scenario to deploy and use community services in contributory manner. Previous work done in CNs has focused on better ways to design the network to avoid hot spots and bottlenecks, but did not related to schemes for network-aware placement of service instances.

However, as services become more network-intensive, they can become bottle-necked by the network, even in well-provisioned clouds. In the case of CN clouds, network awareness is even more critical due to the limited capacity of nodes and links, and an unpredictable network performance. Without a network aware system for placing services, locations with poor network paths may be chosen while locations with faster, more reliable paths remain unused, resulting ultimately in a poor user experience.

We proposed a low-complexity service placement heuristic called BASP to maximise the bandwidth allocation in deploying a CNMC. We presented algorithmic details, analysed its complexity, and carefully evaluated its performance with realistic settings. Our experimental results show that BASP consistently outperforms the currently adopted random (e.g., Naive K-Means) placement in Guifi.net by 35%. Moreover, as the number of services k increases, the gain tends to increase accordingly.

As a future work, we plan to look into service migration, i.e, the controller needs to decide which CNMC should perform the computation for a particular user, with the presence of user mobility and other dynamic changes in the network. In this problem, the user may switch between CNMCs thus another question is whether we should migrate the service from one CNMC to another cloud when the user location or network condition changes.

Acknowledgement

This work was supported by the EMJD-DC program, by the EU H2020 UMobile project (H2020-645124), by the EU H2020 netCommons project (H2020-688768) and by the Spanish Government under contract TIN2013-47245-C2-1-R. This work was also supported by the national funds through Fundação para a Ciência e a Tecnologia with reference UID/CEC/50021/2013.

References

[1] M. Selimi, A. M. Khan, E. Dimogerontakis, F. Freitag, and R. P. Centelles, "Cloud services in the guifi.net community network," *Computer Networks*, vol. 93, Part 2, pp. 373 – 388, 2015.

[2] A. Lertsinsrubtavee, L. Wang, A. Sathiaseelan, J. Crowcroft, N. Weshsuwannarugs, A. Tunpan, and K. Kanchanasut, "Understanding internet usage and network locality in a rural community wireless mesh network," in *Proceedings of the Asian Internet Engineering Conference*, ser. AINTEC '15. New York, NY, USA: ACM, 2015, pp. 17–24.

[3] M. Selimi, N. Apolónia, F. Olid, F. Freitag, L. Navarro, A. Moll, R. Pueyo, and L. Veiga, "Integration of an assisted p2p live streaming service in community network clouds," in *2015 IEEE 7th International Conference on Cloud Computing Technology and Science (CloudCom)*, Nov 2015, pp. 202–209.

[4] M. Selimi, D. Vega, F. Freitag, and L. Veiga, "Towards network-aware service placement in community network micro-clouds," in *Proceedings of the 22nd International European Conference on Parallel and Distributed Computing (EuroPar 2016).* Springer, Aug. 2016.

[5] L. Wang, S. Bayhan, J. Ott, J. Kangasharju, A. Sathiaseelan, and J. Crowcroft, "Pro-diluvian: Understanding scoped-flooding for content discovery in information-centric networking," in *Proceedings of the 2nd International Conference on Information-Centric Networking*, ser. ICN '15. New York, NY, USA: ACM, 2015, pp. 9–18.

[6] L. Cerdà-Alabern, A. Neumann, and P. Escrich, "Experimental evaluation of a wireless community mesh network," in *Proceedings of the 16th ACM International Conference on Modeling, Analysis and Simulation of Wireless and Mobile Systems*, ser. MSWiM '13. New York, NY, USA: ACM, 2013, pp. 23–30.

[7] D. Vega *et al.*, "A technological overview of the guifi.net community network," *Computer Networks*, vol. 93, Part 2, pp. 260 – 278, 2015.

[8] "Stirling Number of the Second Kind," http://mathworld.wolfram.com/StirlingNumberoftheSecondKind.html.

[9] K. LaCurts *et al.*, "Choreo: Network-aware task placement for cloud applications," in *Proceedings of the 2013 Internet Measurement Conference*, ser. IMC '13. New York, NY, USA: ACM, 2013, pp. 191–204.

[10] S. Agarwal *et al.*, "Volley: Automated data placement for geo-distributed cloud services," in *Proceedings of the 7th USENIX Conference on Networked Systems Design and Implementation*, ser. NSDI'10. Berkeley, CA, USA: USENIX Association, 2010, pp. 2–2.

[11] M. Steiner and B. e. a. Gaglianello, "Network-aware service placement in a distributed cloud environment," in *Proceedings of the ACM SIGCOMM 2012 Conference*, ser. SIGCOMM '12. New York, NY, USA: ACM, 2012, pp. 73–74.

[12] 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.

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

[14] S. Wang, R. Urgaonkar, M. Zafer, T. He, K. Chan, and K. K. Leung, "Dynamic service migration in mobile edge-clouds," *CoRR*, vol. abs/1506.05261, 2015.