Heuristic Approach for Optimal Resource Allocation of IoT Networks

by
Hakki Eren Arkangil
M.S., Computer Engineering, Bogazici University, 2019

Cmpe 524 Computer Network Design
Term Project Submitted to the
Institute for Graduate Studies in
Science and Engineering

Bogazici University
2019

TABLE OF CONTENTS

1) INTRODUCTION

Rapid development of information technology has contributed the number of the IoT devices and it is expected that more and more devices are inter-connected via the IoT in the next years. The IoT networks generally consist of two steps. Firstly, the IoT devices connect a gateway. Gateways collects the data from connected IoT devices and then send them to the cloud servers as a whole. The second connection is between gateways and the cloud server. The collected data by the gateways can not divided during the transmission process to the cloud servers. Each gateway have different number of connected devices and collected data size. The cloud servers have capacity constraints with respect to number of IoT devices that they can hold. The amount of the data they process shows the efficiency of a cloud server. Therefore, resource allocation is one of the most important tasks in cloud computing. It consists of assigning resources to each incoming user request in such a way that the user requirements are met and specific goals of the cloud provider are satisfied. The wider adoption of cloud computing and virtualization technologies has led to cluster sizes ranging from hundreds to thousands of nodes for mini and large data centers respectively [5]. Optimal assignment of IoT requests in the cloud server is a NP-hard problem and the main objective of this paper is seeking a way to send maximum data to a cloud server while not exceeding its capacity constraint.

2) RELATED WORKS

Ceri, Martella and Pelagatti (2013) researched this problem for the distributed database and try to solve it with branch and bound algorithm. They compared their result based the required CPU time [1].

Amarante, Cardoso, Roberto and Celestino Jr. (2013) look this problem from a different standpoint by making the resource allocation more specific and worked on cloud computing area [2].

Talari, Shafie-khah, Siano and Loia (2017) showed that the importance of more efficient file allocation [3].

Lyazidi, Aitsaad and Langar (2016) clearly explained the working principles of sensor nodes and how this can end up with the resource allocation problem [4].

3) PROBLEM STATEMENT AND FORMULATION

Clouds are multiple data centers made up of compute and storage resources connected by a network such as IoT network. Connection between IoT devices and cloud server is provided by the gateways. Therefore, gateways have a crucial role in an IoT network. The service demand by IoT devices is sent to cloud through gateways. All of cloud resources are virtualized into one big shared pool of resources in a cloud server. Then, the cloud server can adapt to meet the demands of IoT services which have been sent through gateways and availability of each resource. The demand source can vary from smart-home IoT devices or temperature sensors to the healthcare.

Cloud servers are assumed as public in this model. Public cloud is a type of cloud hosting in which resources of the cloud are rented to many clients. In the model cloud resources will be provided to gateways which are obliged to forward the data of the connected IoT devices. This can cause resource issues such as congestion or failure due to resource capacity of the cloud server. This drawbacks can effect the efficiency of the network. There are two levels of connection between a gateway and cloud server. The platform level where virtual machines are allocated and user tasks are scheduled for execution on them; and the infrastructure level where physical machines are allocated and virtual machines are scheduled to run on them. Each Physical Resource or server (PR) will host a set of Virtual Machines (VM). We are interested in the optimal assignment of gateways to cloud server VMs and trying to utilize the total amount of data in the cloud server with virtual machines constraints. The following informations on virtual machine architecture in the cloud network will help understanding the cloud constraints. The multi-layer framework includes: 1. A physical resource layer which is composed of data centers that host PMs in the form of host machines. The PMs are interconnected by switches (SWs). 2. A virtual resource layer which lies above the physical resource layer to virtualize the physical resources as VMs for better resource management. 3. An application layer which lies above the virtual resource layer to provide a variety of services to users. These include SaaS, PaaS and IaaS [6]. Solving the problem of resource allocation in cloud while maximizing data efficiency is a very compelling issue. This problem is known as NP-hard [7,8] and has been largely studied in the context of cloud computing [9,5,6]. The objective of the paper is to provide solution regarding data efficient resource allocation in cloud.

The number of the IoT devices connected to a gateway will be given and it is denoted by wi. The information of gateway number is provided and it is denoted by n. The total amount of the data sent by IoT services to gateways is also given and it is represented by di, while xi is the decision variable. In the objective function below we want to maximize the amount of the data which is processed in the cloud server. Each IoT device has a service demand and thereby a cost for cloud hosting. For simplicity we assume each IoT device asks only for one virtual machine service from the cloud server. Therefore total service demand of a gateway is equal to wi. If the capacity of the cloud server is not sufficient to allocate a new gateway xi will be 0 and the gateway will not be connected to the cloud. The total capacity of the cloud is denoted by c. Note that, the gateways which could not be assigned any cloud service should not wait forever, this problem will be questioned in the next papers. My suggestion for this problem is to leave the gateways unassigned until they reach a data size threshold.

The problem also will be worked for multiple cloud servers. This is an another optimization problem known as Bin packing problem.

Variables: $xi = 1$ if item i is selected, otherwise $xi = 0$, for $i = (0, 1, 2 \ldots n)$

1) $maximize \sum_{i=1}^{n} di\ xi$

Subject to:

2) $\sum_{i=0}^{n} w_i\ x_i \leq C$

$xi \in \{0,1\}$ xi $\in \{0,1\}$ for $i = (0, 1, 2 \ldots n)$

In order to solve this combinatorial optimization problem this paper suggests an efficient algorithm based on heuristic approach. I give more information about this algorithm in the section IV. Thanks to this algorithm, the processed data size in the cloud server is utilized.

4) GREEDY ALGORITHM BASED SOLUTION TECHNIQUE

The aim of the paper is to propose an algorithm that takes the available capacity constraints described in Section III as inputs, and find an optimal or near-optimal solution as a network topology which includes the assignment of gateways to a cloud server .

The "knapsack" problem is NP-hard; i.e., there is no optimization technique that solves the "knapsack" problem in linear time complexity. Considering the difficult optimization problem at hand, it is not possible to guarantee the optimal solution in reasonable run times. Therefore, techniques that offer near-optimal solutions within acceptable run times are required. One such method that finds a sub-optimal solution in reasonable time without searching the whole solution space is Greedy Algorithm (GA).

The algorithm takes 3 parameter as inputs which are number of gateways, cloud server capacity and data size-device number pairs and returns the optimal combination which maximize processed data size in the cloud server. Di and wi are the data and service demand values of the relevant gateway. For each gateway we will obtain an integer number which is represented by R. N is the number of gateways.

```
Greedy {
Input: Numbers of gateway with their data and IoT information
Output: Index of gateways and sum of the their values

    For i=1 to n
        Compute di/wi ratio

    ->Sort objects in NON-INCREASING order di/wi = R.

For i=1 to n
   ->Keep connecting to cloud till whole devices can be taken
   ->If the R is lower than 0.035:
      Do not assign this object to the cloud.
   ->If the cloud is out of capacity or we can't send gateway data fully:
     Note which object is that (get index) and break out of loop

}
```

As it can clearly seen above, while n is increasing, the problem become more complicated. This paper also suggests giving R a threshold, thus the algorithm can eliminate gateways which contains low data size and big service demand. The cost of assigning these devices is high. We get more efficient results by ignoring them.

5) PERFORMANCE EVALUATION

The simulations will be conducted using Gurobi Optimizer. I will compare the results of the greedy algorithm to the GUROBI and analyze how far my algorithm is from the optimal solution. The algorithm will be tested for different gateway numbers with different cloud server capacities. The number of the gateways defines the problem size. Therefore the experiments will be done for 4,10,15,20,22,25,27,30,32,35,37 and 40 gateways for lightly, random and heavily loaded enviroments. Heavily loaded enviroments only consists of gateways which have more than 35 connected IoT devices while lightly loaded enviroments includes gateways which have less than 35 IoT device. Random enviroments are just the combination of them. The capacity constraints of the cloud is increased gradually from 100 to the 300 for heavy and from 60 to 225 for lightly loaded environments.

In order to find correct comprasion values of our algorithm , the each group of experiments are performed using different parameter values on different data sets. Experiments are repeated 30 times for each data and parameter set. Comparative study of experiments has been done and results are taken and shown in table 1 and table 2 given below. Form the tables in the Appendix A it is clearly shown that our algorithm gives almost the same results with the Gurobi Optimizer.

Table 1: Heavily loaded Environment

Number of Gateway	Greedy Algorithm(mean)	Gurobi Optimizer(mean)
4	328.5	331
10	513	566
15	674.5	695
20	807.5	857.5
22	824	897.5
25	963	982.5
27	1041	1083.5
30	1160	1179
32	1133	1186.5
35	1238	1238
37	1241.5	1282
40	1474.5	1490

Table 2: Random Loaded Environment

Number of Gateway	Gurobi Optimizer(mean)	Greedy Algorithm(mean)
4	422.5	422.5
10	1032	1032
15	2095.5	2077
20	2018	2009
22	2391	2370
25	2580	2572
27	3391	3391
30	3137	3132
32	3600.5	3587.5
35	3746.5	3742.5
37	4434.5	4423
40	4317.5	4317.5

Table 3: Lightly loaded environment

Number of Gateway	Gurobi Optimizer(mean)	Greedy Algorithm(mean)
4	574	574
10	898	880.5
15	1238.5	1237
20	1671	1633.5
22	1626.5	1622
25	1782.5	1758
27	1925	1849.5
30	2292	2232.5
32	2301	2290.5
35	2289	2286
37	2522.5	2508.5
40	2651	2651

6) CONCLUSION

The GA results are promising, since for the cases studied, the quality of GA results almost the same with Gurobi optimizer. Although the greedy algorithm never could beat Gurobi Optimizer mean of the difference between their results are maximum 75.5 gigabyte. That means the proposed algorithm is 4 per cent far from the optimal solution in the worst case and it is observed in the most of the experiments both algoritms give the same result. Greedy algorithm is easy to implement, and it is time saving. However, future works still aim implementing improved methods to find global optimum.

APPENDIX A

Figure 1: Heavily Loaded Environment

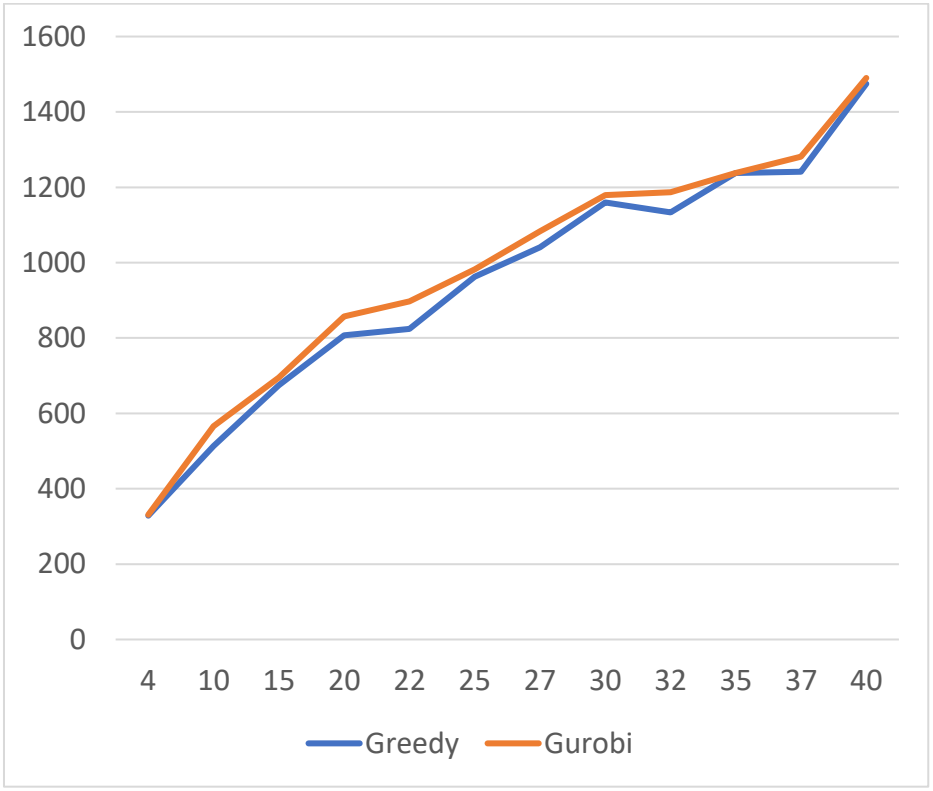


Figure 2: Lightly Loaded Environment

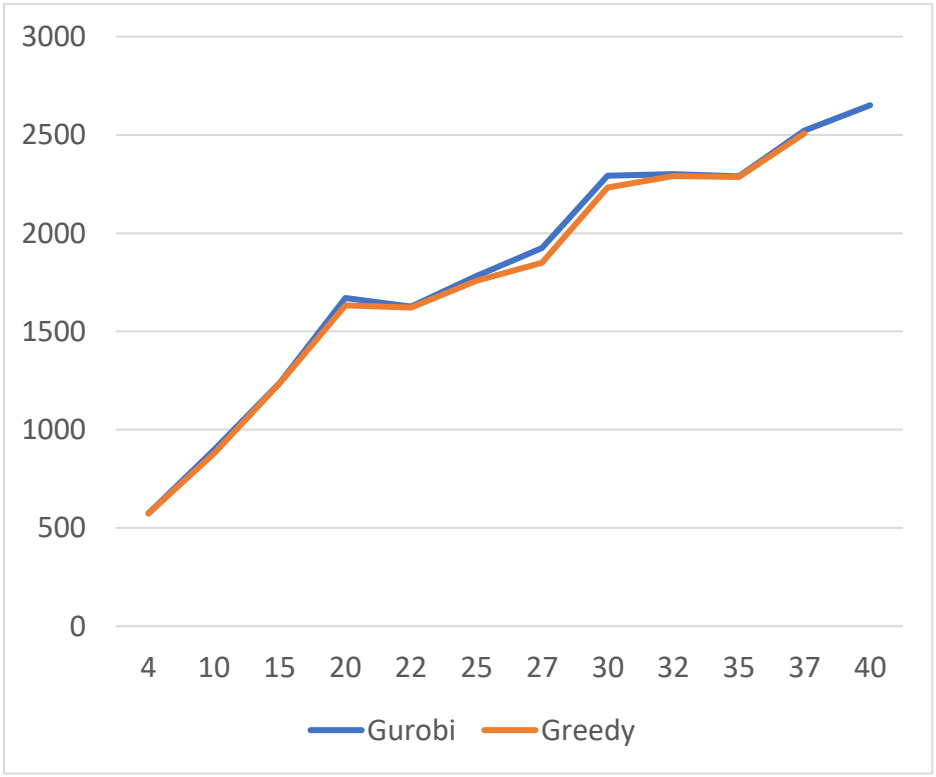
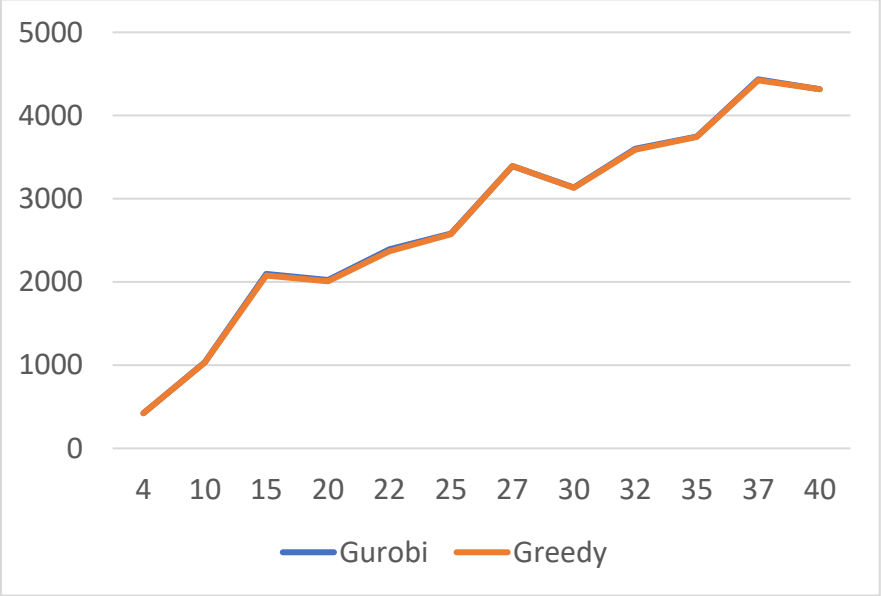


Figure 3: Random Loaded Environment



APPENDIX B

Listings of software is given to the Institute. The diskette/CD contains files such as the source code, one or more sample input and corresponding output separately.

REFERENCES

1) Optimal File Allocation in a Computer Network: a Solution Method Based on the Knapsack Problem. Ceri S., Martella G., Pelagatti G. (2013). Milano, Italy

2) Using the multiple knapsack problem to model the problem of virtual machine allocation in cloud computing. Amarante S., Cardoso A., Roberto F., Celestino J. (2013). 2013 IEEE 16th International Conference on Computational Science and Engineering.

3) A Review of Smart Cities Based on the Internet of Things Concept. Talari S., Shafie-khah M., Siano P.,Loia P. (2017). Lisbon, Portugal.

4) Dynamic Resource Allocation for Cloud-RAN in LTE with Real-Time BBU/RRH Assignment. Lyazidi M., Aitsaadi N., Langar R. (2016). IEEE ICC 2016 - Next-Generation Networking and Internet Symposium

5) Energy efficient resource allocation in cloud computing environments. Ghribi, C. (2014). Informatique, T´el´ecommunications et Electronique de Paris

6) Improving Quality-of-Service in Cloud/Fog Computing through Efficient Resource Allocation. Akintoye S, Bagula A. (2019). ISAT Laboratory, Department of Computer Science, University of the Western Cape, Bellville 7535, South Africa

7) NP-Completeness of Knapsack. Peng R. (2016). CS 3510 Design & Analysis of Algorithms. Georgia Institute of Technology.

8) A well-solvable special case of the bounded knapsack problem. Deineko VG, Woeginger GJ. (2010). Operations Research Letters.

9) A low-level resource allocation in an agent-based Cloud Computing. Bajo J, Prieta F, Corchado J, Rodriguez S. (2016). Applied Soft Computing.