Optimal Network Design: Edge Server Placement and Link Capacity Assignment for Delay-Constrained Services

Devyani Gupta
Department of Electronic Systems Engineering
Indian Institute of Science
Bangalore, India
devyani@iisc.ac.in

Joy Kuri
Department of Electronic Systems Engineering
Indian Institute of Science
Bangalore, India
kuri@iisc.ac.in

***Abstract*—A general communication network has a single Data Center (DC) in its "core", which serves as a gateway to the Internet. For delay-constrained services of the kind needed by online gaming, this model does not suffice because the propagation delay between the subscriber and the DC may be too high. This requires some servers to be located close to the network edge. Thus, the question of the optimal placement of these edge servers arises. To lower the network design cost, it is also essential to ensure good traffic routing, so that aggregate traffic on each link remains as low as possible. This enables lower capacity assignment on each link and thereby minimizes design cost. In this paper, we study a novel joint optimization problem of network design cost minimization. Edge server placement cost and link capacity assignment cost constitute the total cost. The problem formulated is a large Integer Linear Program (ILP). Unlike others, we provide an *exact* solution in reasonable time. To achieve this, we apply the Column Generation (CG) technique. The results show a 40% improvement in the design cost, when solved through CG, over other heuristics.**

***Index Terms*—Network Design Problem, Mobile Edge Computing, Integer Linear Program, Column Generation**

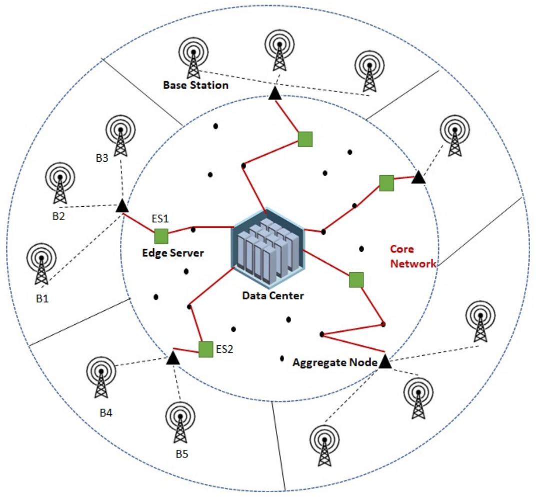


Fig. 1: Illustration Topology.

I. Introduction

We study a problem of optimal network design for delay-constrained applications.

Modern communication networks have a Data Centre at the "core." It is a large group of computer systems which are used for remote storage, processing and distribution of huge amount of data [1]. Generally there is a single DC in a network that serves as a gateway to the Internet. Therefore, all the nodes in the network must be connected to the DC.

With the advent of 4G (and now 5G), services like Mobile TV, video calls, live-streaming videos, gaming and buffer-less HD video streaming are provided to customers, along with voice and data services. These services require high bandwidths and low latency. For instance, to play a YouTube video in 4K resolution, a sustained speed of 20 Mbps is recommended [2]. An acceptable ping for online gaming is around 40-60 ms and if it is greater than 600 ms, some games reject the connection entirely [3]. For delay-constrained services of the kind needed by gaming applications, this model (single DC network) does not suffice because the propagation delay between the subscriber's node and the DC may be too high. The need for ensuring low latency services becomes much more pronounced in the current COVID-19 pandemic situation, which has forced people to spend far more time online and caused a sharp increase in global peak traffic – up to 47% [4]. This requires some servers to be located close to the network edge. These are called Edge Servers (ES). This is illustrated in Fig. 1. Subscribers are connected to a Base Station (BS) which in turn is connected to the DC via a backbone network. It is possible that a service request with a tight delay requirement arriving at B1 cannot be handled by the DC because the propagation delay between B1 and the DC might be too large. Thus, ESs are placed near the edge so that good quality service is provided to subscribers with stringent delay requirements. In the figure, it can be seen that an edge server ES1 is placed which caters to the requests received by base stations B1, B2 and B3.

We view the network as an uncapacitated graph where node and links are given, but the link capacities are not specified. Network traffic is modelled as a **fluid flow**. A solution to the design problem must indicate (a) the locations of the edge servers, and (b) the capacity (from a given set of possible capacity values) to be assigned to each link. This can also be seen from Fig. 1. The links which route ES to DC (shown in red) have high capacity while other links which do not carry traffic are assigned zero or minimum bandwidth (for readability we have not shown uncapacitated links in the picture).

Given the above, the network provider's problem is to design a network that minimizes cost. The cost arises from the following two factors:

- Installing edge servers on nodes – the closer a node to the edge, the higher the cost of installing an edge server [5], [6]. This is because installation of these edge servers requires infrastructure (AC, power supply, etc.) of its own. It is costlier to find space near the subscriber or where the population is more.
- Assigning capacities from a given finite set of possible capacity values to links. It is obvious that high capacity links will cost more.

The total cost is the sum of the two. This is a **joint optimization problem**, in which both edge server placement and link capacity assignment must be obtained simultaneously. Solving the problems sequentially will lead to sub-optimal cost. An important point to note is that the cost optimization requires optimal traffic routing as well. An arbitrary routing choice may lead to excess flow on some links, which would imply unnecessarily high link capacities and consequent high cost.

Our study of the literature (Table I) shows that the joint optimization problem of edge server placement and link capacity selection has not been studied earlier. Some authors have studied either the edge server placement problem independently without considering routing [5], [7] or just the capacity assignment problem (CAP) [8], [9]. Thus, to the best of our knowledge, this paper is the first to study the joint optimization problem with the objective of minimizing the total cost. The problem formulated is a large Integer Linear Program (ILP) [10].

Moreover, when faced with an ILP, most papers propose heuristics to provide a near-optimal solution quickly. In contrast, our aim is to provide the exact optimal solution in very reasonable time. We have used Column Generation (CG) technique [11] as our solution approach.

This paper makes the following *contributions*:

- We study a novel joint optimization problem of minimizing the total network design cost, consisting of edge sever placement and capacity assignment costs. The solution provides both edge server placement locations and link capacity values in one shot.
- The problem formulated is a large ILP. We have used Column Generation (CG) as the solution technique to provide the exact solution quickly. Even though CG is known for decades, this is the first time it is being deployed in the novel design problem considered.
- The major challenge while working with CG is to come up with the *'CG Equivalent'* of the ILP formulated. The constraint set of the *Pricing Problem* (PP) must capture the required feasible set exactly for CG to produce true global optima.
- We have applied CG only to a part of the problem, i.e. the edge server placement and traffic routing part. These two aspects are captured in the notion of "configuration" that we introduce. The number of configurations possible is extremely large, and this is where CG plays an important role. As the problem considered is novel, there is no solution in the literature with which our proposed method can be compared directly. We have studied the solutions to the two related problems that other works consider **in isolation** – Edge Server Placement and Link Capacity Assignment problems. We have captured the core ideas in other authors' solutions in our proposed heuristic *Farthest Node Shortest Path* (FNSP) algorithm. Results show an improvement of 40% in the total cost compared to FNSP, when CG is used.

The rest of the paper is organized as follows. Section II provides the state of the art. Section III formally describes the mathematical model, and the Column Generation technique is explained in Section IV. Section V shows the power of CG when compared with other heuristics. Section VI concludes the paper.

II. Related Work

Table I shows the state of the art. We can broadly classify the papers in two categories; Edge server Placement Problem (ESPP) and Capacity Assignment Problem (CAP).

Our work is different from the literature in terms of the problem definition and also in terms of the solution approach. Most authors focus on providing solutions quickly through heuristics at the cost of optimality. On the contrary, we aim to provide *exact* solutions in comparably short times.

III. System Model and Problem Formulation

We are given the network node connections and many subscribers wanting to subscribe to a gaming service. A subscriber has a target delay requirement that must be met not to affect the gaming experience. All subscribers with roughly the same target delay requirement are grouped into an aggregate. This can also be understood from Fig. 1. Base stations B1, B2, and B3 are grouped together, and the aggregate traffic enters the core network through a single node called "Aggregate Node" or "Source Node." The subscribers using the same source node are collectively called "Aggregate-subscriber." The network design problem deals with these aggregate traffic flows. One DC is present in the core network, which acts as the Internet gateway for all the nodes. Therefore, each network node must be connected to DC. The propagation delay between the aggregate-subscriber and the DC could be high, and if the gaming server is placed at the DC, then the experience could be poor. In such cases, we need to place an edge server corresponding to each aggregate-subscriber so that the aggregate's delay constraint is met. There can be many ways of placing an edge server, and there can be

TABLE I: Literature

Type	Author	Objective	Capacity or Delay Constraint	Highlights
Edge Server Placement Problem	Wang *et al.* [7]	min weighted sum of workload difference and access delay	No threshold for both	One edge server can cater to multiple BS-s. All edge servers must be allocated exactly on BS. No routing from BS to DC. Abstract level formulation. CPLEX used to provide *Exact* solution
	Li *et al.* [5]	min total energy consumption	Upper bound on both	There are direct connection between BSs and edge servers and hence euclidean distance is taken to measure access delay. No routing from BS to DC is considered. Proposed PSO based heuristic which gives near-optimal solution
	Lahderanta *et al.* [12]	min the weighted distances between edge servers and the APs	Lower and Upper bound on edge server capacity	The workload of an AP can also be handled by multiple edge servers. The proposed PACK algorithm uses allocation-relocation process to find near-optimal solution.
	Kang *et al.* [13]	min weighted sum of workload and total delay	No threshold for both	Designed geographic clustering and collaborating scheduling mechanism to reduce delay without increasing the workload.
	Bhatta *et al.* [14]	min total cost of placing edge server and min total latency	Upper bound on number of cloudlet placed	Genetic algorithm based approach is used to find near-optimal solution.An initial random placement is chosen and improved until 'R' best solutions are found. One is finally chosen from the set.
	Cao *et al.* [15]	Six objectives. min {delay, load, energy, . . .}	Not explicitly considered	They have focused of reasonably clustering the RSUs so that deployment cost is reduced. Improved algorithm PCMaLIA is used to obtain near-optimal solution.
	Li *et al.* [16]	max Profit by minimizing total energy consumption.	Upper bound for both	Same formulation and solution approach as [5]
	Y. Chen *et al.* [17]	min weighted sum of workload,query balancing and distance for latency	Not explicitly considered	Proposed QIP from small data-set and TAKG heuristic for large data-set. This is a general assignment formulation of edge servers to BS and routing has not been considered.
Capacity Assignment Problem	Roberts [8]	min network cost	May or may not be present	The flow on each link is given as input. In the algorithm, some capacity is assigned to each link and by choosing "AddFast" or "DropFast" criterion minimum cost link capacities are obtained.
	Lin *et al.* [9]	min total cost of delay and fixed and variable cost associated with each link	Threshold on links	They have formulated a 0-1 non-linear program.The proposed Genetic Algorithm takes all the edges as one vector and tries to generate offspring by crossover and mutation process. It, then, keeps improving the vector through fitness function.
	S.G. Chen *et al.* [18]	min capacity assigned to each link such that network survives under link failure	No restrictions on both	The proposed algorithm provides exact solution to the non-linear problem. It identifies the critical links through analysis and assigns the largest capacity to them. Based on reliability, capacity of other links are determined.
	Courtain *et al.* [19]	min net cost between each pair of nodes	Threshold on capacity on some links	The first extension provides an algorithm to find the expected net cost between all pair of nodes. The second extension provides an algorithm to constrained problem where capacity constraint is considered on some links.
	Shen *et al.* [20]	min Total cost (delay + computation + links)	Threshold on link capacity	Tabu search algorithm is used. It is iterative process which finds best solution in the neighbourhood through local search. It may or may not give true optima. Also, the traffic requirement between pair of nodes is given.
	Crainic *et al.* [21]	min Total cost (fixed + transportation)	Threshold on link capacity	Sequential (simplex + CG) and Parallel (independent and cooperative) Tabu search strategies are explored. The formulation and solution approach is similar to ours.
Both	This paper	min weighted sum of cost of edge server placement and cost of link capacity selection	Threshold on both	We have formulated an large ILP which minimises the cost. The solution technique is *Column Generation* which provides *exact* solution in comparable times.

multiple paths from the source node to DC. Each such path, along with the edge server node location, is what we call a "*Configuration*." The same path but with different edge server node locations constitutes two different configurations. Similarly, two different paths with the same edge server node location form two different configurations.

The objective of the paper is to obtain a minimal design cost. This cost comprises two components - (a) Edge server placement cost and (b) Link capacity assignment cost. The cost of placing an edge server on a node is given. Edge server located on a node close to the edge incurs higher cost [5], [6]. Link capacity costs are given. The available capacities are finite and discrete. It is obvious that higher capacity links cost more. Also, for a fixed capacity, longer links have a higher cost.

An arbitrary choice can lead to traffic overlapping on some links. This may increase congestion and force some links to acquire more bandwidth. Thus, we aim to optimally place the edge servers and assign capacity to each link to minimize the total cost. The notations used in the formulation are shown in Table II.

The **objective** of this paper is to minimize the cost which

TABLE II: Defining System Model

INPUT PARAMETERS	
$G = (V, L)$	physical topology of network with V nodes & L links
$u \in U$	Number of aggregate-subscribers
C_u	set of Configurations associated with aggregate-subscriber, u
β_v	Cost of instantiating a edge server on node. v
η_i	i^{th} choice of link capacity, where $i = 1, 2, \cdots, k$
γ_i	Cost of selecting η_i link capacity
d_l	Propagation delay on link l
τ_u	Maximum allowed delay for aggregate-subscriber u
r_u	Traffic rate for aggregate-subscriber u
RMP PARAMETERS / PP VARIABLES	
$b^c_{1,l}$	$= 1$ if link l is used in the path from source node to edge server node for configuration c ; 0 otherwise.
$b^c_{2,l}$	$= 1$ if link l is used in the path from edge server node to DC node for configuration c ; 0 otherwise.
a^c_v	$= 1$ if edge server is placed on node v in configuration c; 0 otherwise.
VARIABLES	
z_c	$= 1$ if configuration c is chosen ; 0 otherwise.
H^i_l	$= 1$ if i^{th} capacity is chosen for link l ; 0 otherwise.

is given as:

$$\min \sum_{u\in U}\sum_{c\in C_u}\sum_{v\in V}\Big(a^c_v \cdot \beta_v\Big) z_c + \sum_{l\in L}\sum_{i=1}^{k} \gamma_i \cdot H^i_l \qquad (\alpha)$$

such that,

$$\sum_{c\in C_u} z_c = 1 \quad ; u \in U \qquad (1)$$

Exactly one configuration is selected for each aggregate-subscriber.

$$\sum_{i=1}^{k} H^i_l = 1 \quad ; l \in L \qquad (2)$$

For each link, exactly one capacity from the given set is assigned.

$$\sum_{c\in C_u}\Big(\sum_{l\in L} d_l \cdot b^c_{1,l}\Big) z_c \le \tau_u \ ; \ u \in U \qquad (3)$$

The total propagation delay from source node to edge server node should be less than the target value.

$$\sum_{u\in U} r_u \sum_{c\in C_u} (b^c_{1,l} + b^c_{2,l}) \cdot z_c \le \sum_{i=1}^{k} \eta_i \cdot H^i_l \quad ; l \in L \qquad (4)$$

The cumulative traffic flowing on each link should be less than the capacity allocated to that link.

$$z_c \in \{0,1\} \quad ; c \in C_u, \ u \in U \qquad (5)$$

$$H^i_l \in \{0,1\} \quad ; l \in L, \ i = 1, \dots, k \qquad (6)$$

The variables are binary which makes the formulation an Integer Linear Program (ILP). We refer to this problem as the **Master Problem**.

IV. Solution Approach

The Master Problem (α) is a large ILP and cannot be solved directly. When confronted with a problem like this, the usual approach is to devise heuristics that yield quick, albeit sub-optimal, solutions. Approaches include PSO [5], [16], Genetic Algorithm [9], [14], PACK Algorithm [12], etc. Most of these are probabilistic approaches and may not provide *exact* optimal solutions. Further, performance and quality of solution depend on a bunch of parameters that must be chosen; for example, in Genetic Algorithms, parameters like mutation probability, crossover probability must be assigned values. Our goal, however, is to pursue exact optimal solutions, without a massive computational burden.

We apply the Column Generation (CG) technique [11] which provides *exact* solutions in quick time. CG has evolved from the *Simplex Method* [22]. It divides the original problem – the *Master Problem (MP)*– into two subsections called the *Restricted Master Problem (RMP)* and the *Pricing Problem (PP)*.

Let λ and ν be the equality constraint and inequality constraint dual variables. Therefore, for our formulation λ_u and λ_l are the dual variables associated with (1) and (2). And, ν_u and ν_l are the dual variables associated with (3) and (4). The RMP is the same as MP but with fewer variables and objective of PP is the LHS of the constraint of the dual of RMP and is given as:

$$PP_{(u\in U)} = \min \sum_{v\in V}\Big(a_v \cdot \beta_v\Big) - {\lambda_u}^1 + \nu_u \sum_{l\in L} d_l \cdot b_{1,l} + \sum_{l\in L} \nu_l (r_u \cdot b_{1,l}) + \sum_{l\in L} \nu_l (r_u \cdot b_{2,l}) \qquad (\beta)$$

such that,

$$\sum_{v\in V\setminus\{S\}} a_v = 1 \qquad (7)$$

where S is the set of source nodes. The edge server must be placed on exactly one of the nodes except source nodes.

$$\sum_{l\in v^+} b_{1,l} = s \ ; \quad v \in S, + = outgoing\ links \qquad (8)$$

The value of s depends whether the source is active or inactive. $s = 1$ if it is an active source; 0 otherwise. The PP is solved per aggregate-subscriber. When PP for aggregate-subscriber u_1 is considered, the source nodes of other aggregate-subscribers does not send service requests. Hence, they are termed as 'inactive' sources. The above constraint ensures that exactly one of the links leaving an active source node must be active.

[1]MATLAB 2020b is used for computation. MATLAB takes constraint (1) as $1 - \sum_{c\in C_u} z_c = 0$. Hence, the coefficient of λ_u is negative in (β)

$$\sum_{l \in v^-} b_{1,l} \;-\; \sum_{l \in v^+} b_{1,l} \;=\; a_v \;;\; v \in M \tag{9}$$

Here M is the set of intermediate nodes. All nodes excluding source node and DC nodes are termed as intermediate nodes. This is a continuity constraint. Suppose the edge server is on node v. Then, $(b_{1,l})$ on all the outgoing links from that node must be zero. Moreover, if the edge server is not on node v, then there are two possibilities — either that node is not a part of the chosen path or that node is a part of the chosen path. In the latter case, exactly one incoming and one outgoing link must be active.

$$\sum_{l \in v^-} b_{1,l} \;= 1;\; v \in D \tag{10}$$

where D is the DC node. This constraint ensures that exactly one of the links coming into the DC node is active.

$$\sum_{l \in v^+} b_{2,l} = 0 \;;\quad v \in S, + = outgoing\ links \tag{11}$$

The edge server cannot be placed on a source node. Therefore, $b_{2,l}$ on all the outgoing links from a source should be 0.

$$\sum_{l \in v^+} b_{2,l} \;-\; \sum_{l \in v^-} b_{2,l} \;=\; a_v \;;\; v \in M \tag{12}$$

Again, this is a continuity constraint. Suppose the edge server is on node v. Then, $(b_{2,l})$ on all incoming links to that node must be zero. Moreover, if the edge server is not on node v, then there are two possibilities — either that node is not a part of the chosen path or that node is a part of the chosen path. In the latter case, exactly one incoming and one outgoing link must be active.

$$\sum_{l \in v^-} b_{2,l} \;\leq\; 1 \;;\; v \in D \tag{13}$$

The edge server can be placed on an intermediate node or on DC node. Therefore, at most one of the incoming links to DC must be active.

$$a_v \in \{0,1\} \;;\; v \in V \tag{14}$$

$$b_{1,l} \;\; b_{2,l} \in \{0,1\} \;;\; l \in L \tag{15}$$

$$V = \{S \cup M \cup D\} \tag{16}$$

The variables here are also binary, which makes the PP an ILP.

It may be noted that a_v, $b_{1,l}$ and $b_{2,l}$ had appeared as constant coefficients in the constraints of the RMP (Primal Problem); in the PP, they are the variables.

- $b_{1,l} = 1$ if link l is used in the path from source node to edge server node ; 0 otherwise.
- $b_{2,l} = 1$ if link l is used in the path from edge server node to DC node ; 0 otherwise.
- $a_v = 1$ if edge server is placed on node v ; 0 otherwise.

V. Results

We have taken the 18-node 39-link topology for evaluation of our formulation. We assume this network to be the backbone network of an Internet Service Provider (ISP) and it has one DC in it.

In our model, we have varied the aggregate-subscribers from 4 to 10 and observed the change in the cost and the link capacities. In Fig. 2, the 'Setup' is defined as the specific parameter values which are taken while evaluation. For instance, the 'Setup A' in Fig. 2(b) is defined as follows:

- Allowed Delay for six aggregate-subscribers is $\{1, 2, 8, 3, 4, 5\}$ ms.
- Node placement cost for edge servers is taken randomly between Rs. $[50, 1000] \times 10^3$.
- Link capacity choices are $\{1, 5, 20, 50\}$ Gbps.
- Link capacity selection cost for above capacities is taken to be Rs. $\{5, 25, 100, 250\} \times 10^3$.
- Link propagation delay is taken randomly between $[0, 9]$ ms.
- Traffic rate for six aggregate-subscribers is $\{1, 2, 3, 4, 5, 6\}$ Gbps.

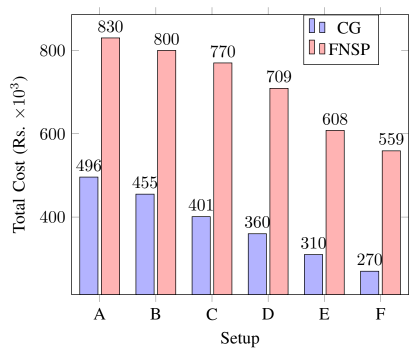
For comparison and performance evaluation of our proposed technique, we explored various heuristics like PSO from [5], PACK from [12], Automata-based algorithm from [8], Genetic Algorithm from [9], etc. to name a few.

To the best of our knowledge, the joint edge server placement and capacity assignment problem has not been studied and therefore there is no heuristic or solution approach available for direct comparison. Based on our survey, we have come up with a potential candidate heuristic for comparison, called *Farthest Node Shortest Path (FNSP) Algorithm.*

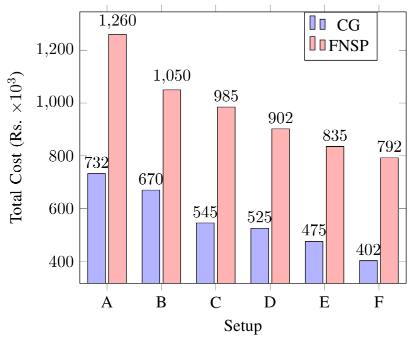
A. Farthest Node Shortest Path (FNSP) Algorithm

This algorithm can be thought as a greedy algorithm. It divides our joint problem into two parts. Firstly, it places the edge server on a node such the delay constraint is met with its maximum value, i.e., on a node as deep as possible inside the core network. Then, it finds the shortest path from edge server node to the DC node. Finally, the edge server placement cost and link capacity selection cost are calculated.

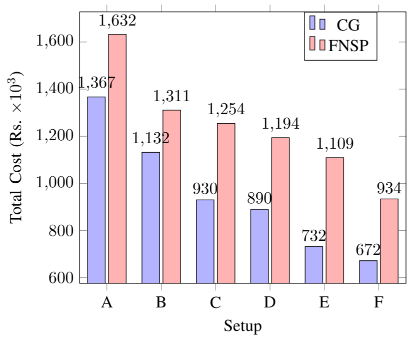
We have compared our solution technique, i.e. Column Generation (CG) with FNSP as shown in Fig. 2. For different Setups (explained previously), we have calculated the total cost incurred. In Fig. 2(a), we have placed four edge servers for four different aggregate-subscribers based on the given target delay. We have increased the target delay and reported the total cost incurred. Thus, we can observe that the total cost reduces along x-axis (Setup) as we know that the placement cost of an edge server reduces as we go deeper into the network. For Setup F, we have kept the acceptable delay so high that even the longest path from source to DC is well within the delay limits. This means that the DC itself can serve the application or service directly. In other words, there is no need to place an additional server on the edge of the core network. We can clearly see from Fig. 2(a) that when compared with FNSP, CG



(a) Placement of 4 edge servers to cater to 4 aggregate-subscribers. The total allowed delay is increased and total cost is calculated.

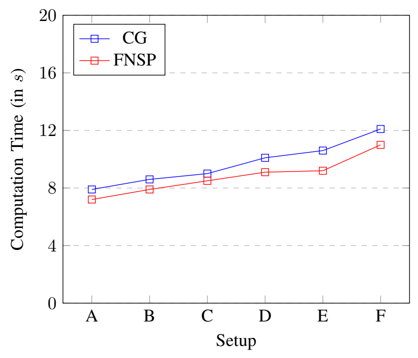


(b) Placement of 6 edge servers to cater to 6 aggregate-subscribers. The total allowed delay is increased and total cost is calculated. Additional 2 aggregate-subscribers are added on previous model.

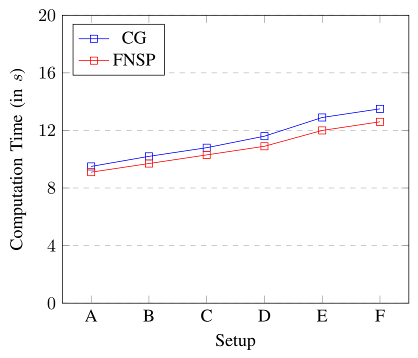


(c) Placement of 10 edge servers to cater to 10 aggregate-subscribers. The total allowed delay is increased and total cost is calculated. Additional 4 aggregate-subscribers are added on previous model.

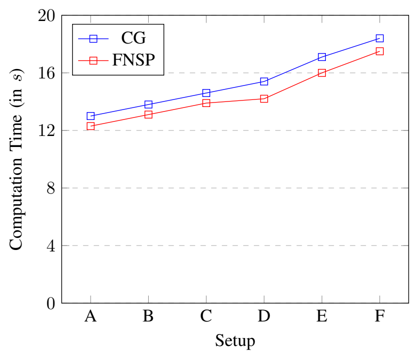
Fig. 2: Total cost incurred in shown in the backbone network. It can be seen that the CG method produces a network design that costs far less than that produced by FNSP. The number of aggregate-subscribers is varied and results are shown for 4-, 6-, and 10-Aggregate-subscribers respectively. The total allowed delay is increased from Setup A to Setup F and and total cost is reported. The cost reduces and is the least in 'Setup F' because the allowed delay is very large and all the aggregate-subscriber services are catered to by the DC directly. This means no edge server is required in 'Setup F'.



(a) Execution time for placing 4 edge servers for each Setup in Fig. 2(a).



(b) Execution time for placing 6 edge servers for each Setup in Fig. 2(b).



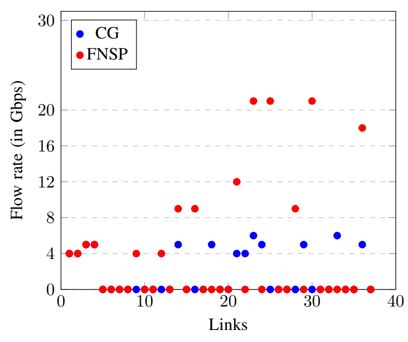
(c) Execution time for placing 10 edge servers for each Setup in Fig. 2(c).

Fig. 3: Execution time in MATLAB 2020b to compute the optimal solutions for various cases considered in Fig. 2.

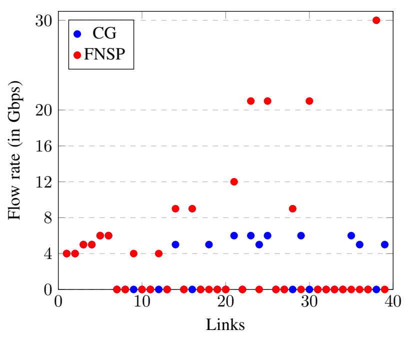
performs better as it produces true optima as compared to near optimal solution produced by FNSP. The overall improvement in total cost is close to 40%. Similar results can be seen in Fig 2(b),(c) wherein we have increased the aggregate-subscribers from four to six and ten respectively. In each of the above mentioned scheme, CG outperforms FNSP and the reducing value to total cost is observed along Setups because we have increased the allowed delay along Setups.

FNSP finds the shortest path in later part of the algorithm. This makes some links more crowded than the other. This can be observed in Fig. 4. On the contrary, our mathematical model finds a balanced path from source to DC via edge server node such that the traffic is evenly distributed among the network links.

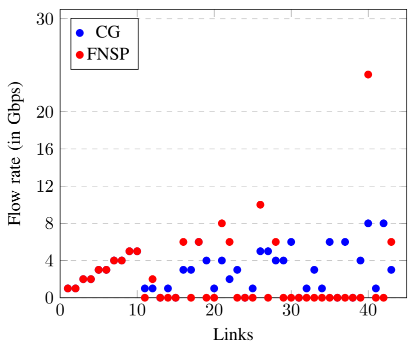
CG solves large ILP problems by an iterative process and starts with fewer variables. In our model, we have applied CG to the first part of the problem only, i.e. the edge server placement part. This is because the number of configurations is very large when compared to the number of capacity choices available per link. In our calculations, we have taken one configuration per aggregate-subscriber as the starting point of CG. For instance, in Fig. 2(a), we have taken four configurations as starting point for CG. The initial set of RMP has many variables out of which four corresponds to configurations of edge server placement. One of the benefits of using CG is that it arrives at the optimal solution quickly. Fig. 3 shows the execution time of CG and FNSP. It can be observed that CG takes slightly longer to produce the optimal solution than the solution produced by FNSP. As we move along x-axis, the execution time increases in Fig. 3. This is because



(a) Traffic seen on each link in scheme Fig.2(a) for Setup A.



(b) Traffic seen on each link in scheme Fig.2(b) for Setup A.



(c) Traffic seen on each link in scheme Fig.2(c) for Setup A.

Fig. 4: Traffic flows seen in Setup for the 4-, 6-, and 10-Aggregate-subscriber cases. FNSP causes some links to carry more traffic whereas CG provides balanced flow in the network which leads to lower design costs.

we have increased the propagation delay along Setups. As a result, the feasible set also increases. Therefore it takes a little longer to compute. But overall, the percentage increase in computation time of CG is far less than the percentage improvement obtained in the total cost from FNSP.

VI. Conclusion

We have studied a novel joint optimization problem to minimize the total design cost incurred from edge server placement and link capacity assignment. The problem formulated is a large ILP. We have applied the Column Generation technique to obtain the true optima, unlike others who have tried to find a near-optimal solution through the heuristic approach. CG is an old technique but its application in network design problems is novel. It is challenging to come up with the correct "*CG equivalent*" of the formulated problem. We tried to compare the power of CG with common heuristics like PSO and Genetic algorithm. These algorithms are probabilistic and boil down to something similar to our proposed algorithm (FNSP). We have analyzed three schemes (Fig. 2) where we increase the aggregate-subscribers from 4 to 10, and the results show that CG performs much better than FNSP in reasonable time (Fig. 3). Fig. 4 shows that our formulation provides balanced traffic routing as compared to FNSP. Thus, the simultaneous placement of the edge server and selection of link capacity offers better results when solved through Column Generation.

References

[1] "Data Center." https://www.cisco.com/c/en_in/solutions/data-center-virtualization/what-is-a-data-center.html, 2014.

[2] "YouTube Help." https://support.google.com/youtube/answer/78358?hl=en.

[3] C. Jiang, A. Kundu, S. Liu, R. Salay, X. Xu, and M. Claypool, "A survey of player opinions of network latency in online games," 2020.

[4] "COVID-19 Impact on Internet Performance." https://www.internetsociety.org/wp-content/uploads/2020/12/Asia-Covid-report-EN-March-2021.pdf, 2020.

[5] Y. Li and S. Wang, "An energy-aware edge server placement algorithm in mobile edge computing," in *2018 IEEE International Conference on Edge Computing (EDGE)*, pp. 66–73, IEEE, 2018.

[6] S. Khirman and P. Henriksen, "Relationship between quality-of-service and quality-of-experience for public internet service," in *In Proc. of the 3rd Workshop on Passive and Active Measurement*, vol. 1, 2002.

[7] S. Wang, Y. Zhao, J. Xu, J. Yuan, and C.-H. Hsu, "Edge server placement in mobile edge computing," *Journal of Parallel and Distributed Computing*, vol. 127, pp. 160–168, 2019.

[8] T. D. Roberts, *Learning automata solutions to the capacity assignment problem.* PhD thesis, Carleton University, 1997.

[9] X.-H. Lin, Y.-K. Kwok, and V. K. Lau, "A genetic algorithm based approach to route selection and capacity flow assignment," *Computer Communications*, vol. 26, no. 9, pp. 961–974, 2003.

[10] E. K. Chong and S. H. Zak, *An introduction to optimization.* John Wiley & Sons, 2004.

[11] G. Desaulniers, J. Desrosiers, and M. M. Solomon, *Column generation*, vol. 5. Springer Science & Business Media, 2006.

[12] T. Lähderanta, T. Leppänen, L. Ruha, L. Lovén, E. Harjula, M. Ylianttila, J. Riekki, and M. J. Sillanpää, "Edge computing server placement with capacitated location allocation," *Journal of Parallel and Distributed Computing*, vol. 153, pp. 130–149, 2021.

[13] S. Kang, L. Ruan, S. Guo, W. Li, and X. Qiu, "Geographic clustering based mobile edge computing resource allocation optimization mechanism," in *2019 15th International Conference on Network and Service Management (CNSM)*, pp. 1–5, IEEE, 2019.

[14] D. Bhatta and L. Mashayekhy, "Generalized cost-aware cloudlet placement for vehicular edge computing systems," in *2019 IEEE International Conference on Cloud Computing Technology and Science (CloudCom)*, 2019.

[15] B. Cao, S. Fan, J. Zhao, S. Tian, Z. Zheng, Y. Yan, and P. Yang, "Large-scale many-objective deployment optimization of edge servers," *IEEE Transactions on Intelligent Transportation Systems*, 2021.

[16] Y. Li, A. Zhou, X. Ma, and S. Wang, "Profit-aware edge server placement," *IEEE Internet of Things Journal*, 2021.

[17] Y. Chen, Y. Lin, Z. Zheng, P. Yu, J. Shen, and M. Guo, "Preference-aware edge server placement in the internet of things," *IEEE Internet of Things Journal*, 2021.

[18] S.-G. Chen, "An optimal capacity assignment for the robust design problem in capacitated flow networks," *Applied Mathematical Modelling*, vol. 36, no. 11, pp. 5272–5282, 2012.

[19] S. Courtain, P. Leleux, I. Kivimäki, G. Guex, and M. Saerens, "Randomized shortest paths with net flows and capacity constraints," *Information Sciences*, vol. 556, pp. 341–360, 2021.

[20] J. Shen, F. Xu, and P. Zheng, "A tabu search algorithm for the routing and capacity assignment problem in computer networks," *Computers & operations research*, vol. 32, no. 11, pp. 2785–2800, 2005.

[21] T. G. Crainic and M. Gendreau, "Cooperative parallel tabu search for capacitated network design," *Journal of Heuristics*, vol. 8, no. 6, pp. 601–627, 2002.

[22] "Simplex Method." https://nptel.ac.in/content/storage2/courses/105108127/pdf/Module_3/M3L3_LN.pdf.