Research On multicriteria optimization

Ilgam Latypov, Yuri Dorn

***Abstract*—Network optimization is a vital task that arises in various graph-based problems. Numerous algorithms have been proposed to address this challenge, each focusing on different criteria such as robustness, cost, and more. This paper introduces a novel approach utilizing Shapley Value (SV) from cooperative game theory for multi-criteria network topology formation. A comparison is made with a greedy algorithm using the example of the Maximum Concurrent Flow (MCF) problem, demonstrating the potential superiority and efficiency of the proposed method. Furthermore, a framework for applying the suggested algorithm is provided.**

***Index Terms*—Network, Graph, Flow ....**

I. Introduction

CONSTRUCTING an optimal network topology is a significant task that arises in any problem represented as a graph. There exists a multitude of algorithms proposed to solve this task for a single criterion [RS96] [KS98]. However, in practice, the optimality of the network is required by a variety of criteria: the network needs to be robust, cost-effective, and perform well according to the desired metrics. Therefore, constructing a topology that optimizes multiple criteria is a significant challenge. There are not so many papers devoted to this task.

In a broader sense, the problem can be cast in the following form:

$$\begin{gathered}\min[f_1(x), ..., f_k(x)] \\ \text{subject to } x \in X\end{gathered} \tag{1}$$

Here, $f_i(x)$ represents the objective functions, and x denotes the one-hot encoding vector of selected edges. We also introduce the notation $f = (f_1, ..f_k)$ - the vector of objective functions. The set X could signify various constraints, such as the total edge cost being less than a given budget. Different concepts of solutions could be understood as min, including the Pareto front [H.R92], which involves presenting a set of solutions for experts to choose from. However, this method does not suit our requirements, as the solution must be determined solely by the program. The alternative approach is to treat the problem as an optimization task. A brief overview of some methods can be found in the article [OCO13]. With this concept, an aggregator is used for the objective functions. We explore the possibility of applying the Shapley Value concept from cooperative game theory to create a representation of the objective functions. Multi-criteria aspects will be ensured through the application of the aggregator for the objective functions.

Through the example of the Maximum Concurrent Flow problem and a stability-based objective function derived from λ_2, we demonstrate that our method performs comparably to the greedy algorithm. Furthermore, the computational efficiency of our approach makes it viable for practical applications.

When working with the Shapley Value, two challenges arise that need to be addressed: 1) computing the Shapley Value itself, which often involves an NP-hard problem; 2) applying the Shapley Value - how to construct a network topology. Our objective is to make advancements in addressing the second challenge.

The rest of this paper is organized as follows. In Section 2, we provide a literature review of various algorithms for network topology optimization. Section 3 introduces the concept of Shapley Value from cooperative game theory and describes the method we use to compute it. In Section 4, we present the greedy algorithm and our proposed algorithm, highlighting the importance of the choice of function for SV. Section 5 describes our experimental setup and presents the numerical results obtained. The computational experiments are discussed in Section 6. Finally, we conclude our work in Section 7 and discuss potential directions for future research.

II. Literature Review

Numerous algorithms have been proposed for network topology optimization. In the work of [GH07], the concept of Pareto front is utilized. [HS18] develop an algorithm to explore the Pareto front in a discrete formulation. They introduce an aggregator for objective functions, a modified Chebyshev norm. However, their approach involves searching for the minimum of the aggregated function, which can be a complex task. [LF18] presents a decision-making framework. [Kam06] introduce the 'average rank' method, akin to averaging players' importance across multiple objectives. Their approach shares similarities with the concept of Shapley Value. Notable Shapley Value-based methods include the work by [JKJ08], who use a series of cooperative games to form network topology, and [Leo14], who apply an algorithm similar to our proposed method for neuron pruning in neural networks. Nonetheless, our study addresses the problem of topology selection in a broader context.

III. Shapley Value

In cooperative game theory, players can collaborate to achieve a joint payoff, which needs to be distributed among the participating players. Let $\mathcal{N}$ denote the set of all players, and $\mathcal{S} \subset \mathcal{N}$ represent a subset of players. We introduce a function

$$v : 2^{\mathcal{N}} \to \mathbb{R}_+$$

which is the value function of the considered game. At this stage, we refrain from making any strong assumptions about the function. It represents the outcome or payoff of the collective game. Game theory offers various ways to distribute the payoff among players. Literature has shown that good properties are satisfied by it [CEW11], which is why we consider it. The payoff of an individual player is denoted as $\phi_i(\mathcal{N})$, where $i \in \mathcal{N}$. According to the SV,

$$\phi_i(v) = \sum_{S \subseteq \mathcal{N} \setminus \{i\}} \frac{|S|!(n - |S| - 1)!}{n!} (v(S \cup \{i\}) - v(S)) \quad (2)$$

As can be observed, the number of terms in the sum exponentially increases with the number of players. This poses the first obstacle in using SV. To overcome this issue, approximate computation methods are used.

A. Literature Review on Shapley Value Approximation

Several algorithms have been proposed for the computation of the Shapley Value (SV). Some of these algorithms leverage the game's structure and exploit patterns to accurately calculate SV [MAS+13], [MS62]. However, in our case, we require methods for approximating the SV, as the value function is treated as a black box. The work by [MCFH21] enumerates popular algorithms for SV approximation. All of these methods operate through the sampling of random player subsets. Additionally, this work introduces novel approaches for SV approximation.

In [KBH23], the SVARM algorithm and stratified SVARM are introduced, which demonstrate improved convergence estimates compared to existing prevalent methods. In [LNB20], the authors demonstrate that, in certain instances, it is possible to efficiently compute $v(S \cup \{i\}) - v(S)$, thereby reducing the computation time for SV.

For our numerical experiments, we decided to use SVARM [KBH23], as it has been proven to be unbiased, with derived variance in SV values based on the number of iterations. Moreover, it often outperforms alternatives and is straightforward to implement.

IV. Proposed Method

Let us initially examine the greedy algorithm for constructing the desired topology. This algorithm will lead us to the idea behind the proposed method. A straightforward approach is to add edges to the graph in a way that maximizes the total value function. However, for many graph-related tasks, meeting the requirements of the function necessitates a sufficient number of edges in the graph. For instance, calculating the shortest path between vertices requires the presence of a path. Therefore, let us consider a greedy algorithm that begins with a graph containing all possible edges. It iteratively removes edges to minimize the impact on the value function. The algorithm is detailed in IV.

After applying the algorithm, we obtain an order in which edges can be deleted. Furthermore, we acquire a trade-off $v(S) = v(|S|)$ for "optimal" solutions.

Next, let us describe the concept of using Shapley Value for the addressed problem. The above-mentioned greedy algorithm ranked edges in a greedy manner. Our aim is to rank them with certain weights, using Shapley Value as these weights [CEW11].

Algorithm 1 Greedy Deletion Algorithm

Input : $\mathcal{N}, v$;
deleted $\leftarrow \emptyset$ {List of deleted players} ;
$S \leftarrow \mathcal{N}$ {Non-deleted players};
while $v(S) \neq 0$ and $S \neq \emptyset$ **do**
  $i \leftarrow \arg\max_i v(S \backslash i)$;
  deleted $\leftarrow$ deleted $\cup\ i$;
  $S \leftarrow S \backslash i$;
end while
return deleted =0

Next, let us describe the concept of using Shapley Value for the addressed problem. The above-mentioned greedy algorithm ranked edges in a greedy manner. Our aim is to rank them with certain weights, using Shapley Value as these weights [CEW11].

Before describing the proposed method, we demonstrate why the knapsack problem is not suitable in this context. Consider players as objects and SV as their values, with the budget representing the total edge cost. Let the graph have the form shown in Figure 1. The value function is a binary function, equal to 1 if there exists a path of length 2 from source to target within the subgraph; otherwise, it is 0. Upon computation, all edge SV values are equal, yet the knapsack will select in the written order. As a result, the cost of the solution is twice as worse as the optimal.

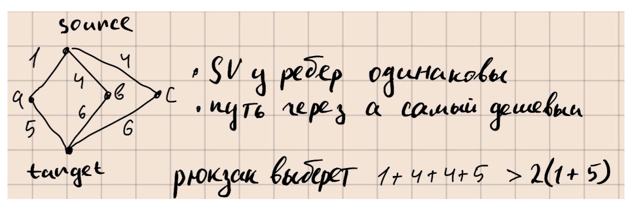

Fig. 1. Example of knapsack application

Now, let's delve into the proposed algorithm.

Algorithm 2 SV Deletion

Input : $\mathcal{N}, v$
$S \leftarrow \mathcal{N}$ {Players that remain}
deleted $\leftarrow \emptyset$ {List of deleted players}
arr $\leftarrow \arg\mathrm{sort}(SV(v))$ {SV(v) – an array of SV values. Sort players in ascending order of SV}
for $i \in$ arr **do**
  if v(S $\backslash \{i\}) > 0$ **then**
    $S \leftarrow S \backslash \{i\}$;
    deleted $\leftarrow$ deleted $\cup \{i\}$;
  end if
end for
return deleted =0

V. Computational Experiments

Let's consider two games related to the maximum concurrent flow

1) Case with two metrics: graph cost and total flow cost.
2) Case with three metrics: graph cost, flow cost, and the second eigenvalue of the graph's Hamiltonian, related to graph stability.

A. MCF with 2 Metrics

To apply the algorithm, we need a value function. Let's demonstrate that it can be composed of simple considerations, specifically as a product of functions corresponding to different metrics.

In this case, there are two metrics: graph cost and total flow cost. Both values should be minimized. Therefore, we aim for the value function to increase as these metrics decrease. The graph cost lies within a certain interval $[0, \text{max\_cost}]$. Let's consider the component responsible for the graph cost, denoted as $g(c) = 1 + \left(1 - \frac{c}{\text{max\_cost}}\right)$. The addition of 1 ensures that the function does not become 0 when all edges are used. The total flow cost exists within a ray $[\text{min\_flow\_cost}, \infty]$. We'll use the function $h(f) = \frac{\text{min\_flow\_cost}}{f}$ to represent the flow cost. This function has a desirable property: if flows are not satisfied, then $h(f) = 0$, implying that the value of a graph unable to accommodate all flows is zero. A simple multiplication of these functions does not work in experiments. However, when their powers are multiplied, the experiments yield the desired result. The resulting value function is given by:

$$v(S) = \left(\frac{1}{2}\left(2 - \frac{c}{\text{max\_cost}}\right)\right)^{\alpha} * \left(\frac{\text{min\_flow\_cost}}{f}\right)^{\beta} \quad (3)$$

Let's examine the algorithm's performance under different parameters. Figure 2 on the left depicts the trade-off between the considered metrics. It is evident that the algorithm performs better with higher powers. Meanwhile, the quality of the greedy algorithm does not vary significantly.

B. MCF with 3 Metrics

In this case, we introduce a stability metric for the graph, denoted as $r(\lambda_2)$. Greater stability corresponds to a larger λ_2, where $\lambda_2 \in [-1, 1]$. Let's consider $r(\lambda_2) = \frac{2-\lambda_2}{2}$.

$$v_2(S) = \left(\frac{1}{2}\left(2 - \frac{c}{\text{max\_cost}}\right)\right)^{\alpha} * \left(\frac{\text{min\_flow\_cost}}{f}\right)^{\beta} * \left(+\frac{2-\lambda_2}{2}\right)^{\gamma} \quad (4)$$

VI. Results Analysis

When the functions are raised to a power, they stretch along the y-axis, causing the differentiating properties of the components of the value function to increase. In other words, if we consider two topologies such that $v_1 > v_2$, raising the components to a power will increase the ratio $\frac{v_1}{v_2}$. Consequently, the first topology becomes even more preferable.

VII. Conclusion

In this work, we presented an algorithm for constructing graph topology based on multiple criteria using Shapley Value. We also proposed assumptions about the method of constructing a value function for Shapley Value computation, which were confirmed in practice. Additionally, we demonstrated that the proposed algorithm performs at least as well as the greedy algorithm. Furthermore, we developed a framework for working with Shapley Value.

References

[CEW11] Georgios Chalkiadakis, Edith Elkind, and Michael Wooldridge. Computational aspects of cooperative game theory. In *Computational Aspects of Cooperative Game Theory*, 2011.

[GH07] C. Gomes and G. Huiban. Multiobjective analysis in wireless mesh networks. In *2007 15th International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems*, pages 103–108, 2007.

[H.R92] H.R.Varian. *MicroeconomicAnalysis*. W.W.Norton Company, 1992.

[HS18] Tim Holzmann and J Cole Smith. Solving discrete multi-objective optimization problems using modified augmented weighted tchebychev scalarizations. *European Journal of Operational Research*, 271(2):436–449, 2018.

[JKJ08] Farshad Javadi, M. Rubaiyat Kibria, and Abbas Jamalipour. Bilateral shapley value based cooperative gateway selection in congested wireless mesh networks. In *IEEE GLOBECOM 2008 - 2008 IEEE Global Telecommunications Conference*, pages 1–5, 2008.

[Kam06] Noriaki Kamiyama. Network topology design with multiple criteria. *Networks 2006. 12th International Telecommunications Network Strategy and Planning Symposium*, pages 1–6, 2006.

[KBH23] Patrick Kolpaczki, Viktor Bengs, and Eyke Hüllermeier. Approximating the shapley value without marginal contributions, 2023.

[KS98] Rajesh M. Krishnaswamy and Kumar N. Sivarajan. Design of logical topologies: A linear formulation for wavelength routed optical networks with no wavelength changers. *Proceedings. IEEE INFOCOM '98, the Conference on Computer Communications. Seventeenth Annual Joint Conference of the IEEE Computer and Communications Societies. Gateway to the 21st Century (Cat. No.98*, 2:919–927 vol.2, 1998.

[Leo14] Florin Leon. Optimizing neural network topology using shapley value. In *2014 18th International Conference on System Theory, Control and Computing (ICSTCC)*, pages 862–867, 2014.

[LF18] Christophe Labreuche and Simon Fossier. Explaining multi-criteria decision aiding models with an extended shapley value. In *IJCAI*, pages 331–339, 2018.

[LNB20] Phuoc Hoang Le, Tri-Dung Nguyen, and Tolga Bektaş. Efficient computation of the shapley value for large-scale linear production games. *Annals of Operations Research*, 2020.

[MAS+13] Tomasz P. Michalak, Karthik. V. Aadithya, Piotr L. Szczepański, Balaraman Ravindran, and Nicholas R. Jennings. Efficient computation of the shapley value for game-theoretic network centrality. *Journal of Artificial Intelligence Research*, 2013.

[MCFH21] Rory Mitchell, Joshua N. Cooper, E. Frank, and G. Holmes. Sampling permutations for shapley value estimation. *Journal of machine learning research*, 2021.

[MS62] Irwin Mann and Lloyd S. Shapley. *Values of Large Games, VI: Evaluating the Electoral College Exactly*. RAND Corporation, Santa Monica, CA, 1962.

[OCO13] GO Odu and OE Charles-Owaba. Review of multi-criteria optimization methods–theory and applications. *IOSR Journal of Engineering*, 3(10):01–14, 2013.

[RS96] Rajiv Ramaswami and Kumar N. Sivarajan. Design of logical topologies for wavelength-routed optical networks. *IEEE J. Sel. Areas Commun.*, 14:840–851, 1996.

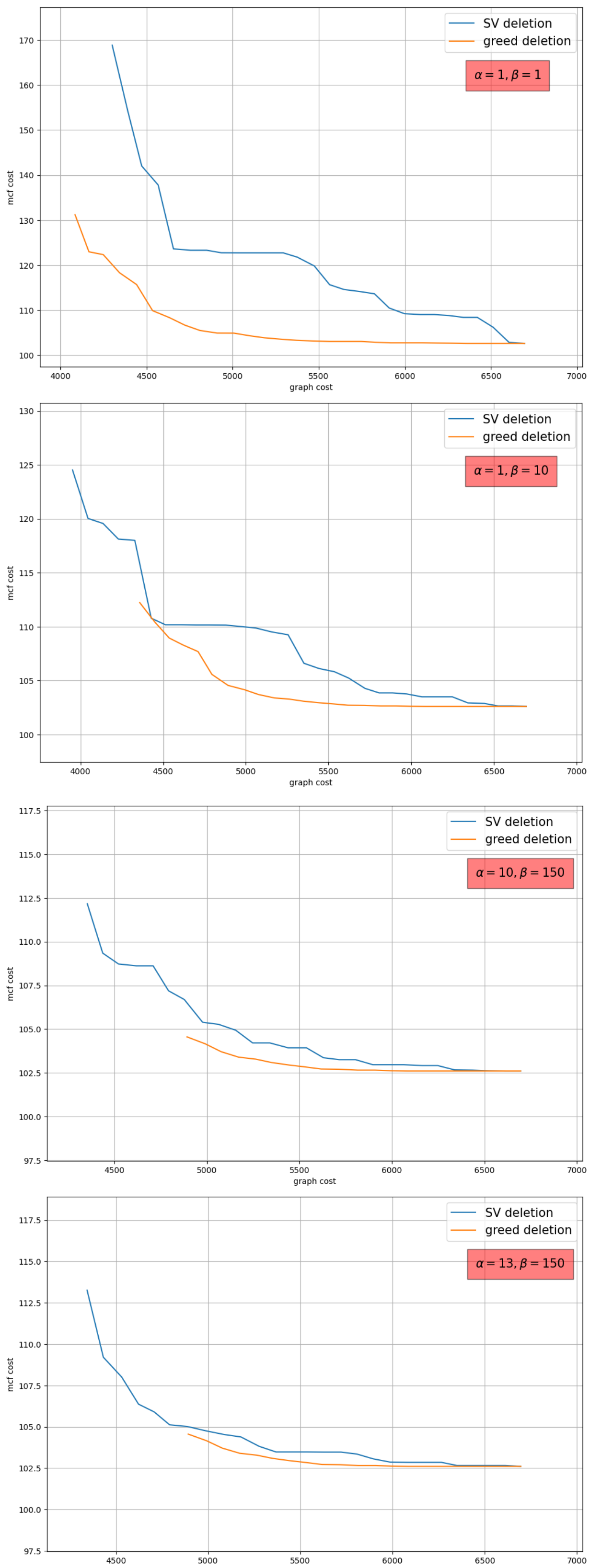

Fig. 2. flow_cost(graph_cost)