UNIFYING FEATURE ATTRIBUTION: BRIDGING ATTENTION MECHANISMS AND SHAPLEY VALUES

Anonymous authors
Paper under double-blind review

ABSTRACT

Machine learning (ML) applications have become ubiquitous in various domains, including and large language models and computer vision. However, the increasing prevalence of ML models has also emphasized the need for explainable artificial intelligence (XAI). In particular, it has become crucial to develop techniques that can help explain how these models work and why they make certain decisions. This is especially important in cases where these models are used to make critical decisions that can have significant consequences on individuals or society as a whole.

1 INTRODUCTION

In the pursuit of transparent and interpretable machine learning models, understanding feature attribution methods is pivotal. Feature attribution elucidates the contribution of individual input elements to model predictions, fostering model interpretability. However, existing methods grapple with challenges, such as non-uniqueness and interpretability gaps. In this paper, we present a cohesive framework by leveraging attention mechanisms and Shapley values.

The attention mechanism, initially designed for natural language processing, has proven instrumental in capturing contextual relationships within input data. Simultaneously, Shapley values, rooted in cooperative game theory, provide a principled approach to distributing contributions among features. We propose a novel integration of attention and Shapley values, unifying their strengths to yield robust and interpretable feature attributions.

Contribution:
Our approach addresses the challenges faced by existing methods, offering a more nuanced understanding of feature importance. Through empirical evaluations on diverse datasets, we demonstrate the effectiveness and versatility of our proposed method in enhancing interpretability across various machine learning models.

Max: Quaily of Writing

Behrooz+Max: Our Contributions?

2 RELATED WORKS

The concept of attention flows was introduced by Abnar & Zuidema (2020) and presented a new perspective on how XAI could leverage attention flows. However, the authors of the paper chose to focus on simple ideas that only required attention weights, which could be easily implemented in any architecture employing self-attention mechanism. While they compared attention flows with other concepts like attention rollout, they did not employ maximum flow to obtain the attribution of each token in language models. In the

classification task, they conducted an evaluation of the weights obtained from raw attention, attention rollout, and attention flow for the CLS embedding over input tokens in all attention layers for a selected set of examples. In the masked language modeling task, a similar analysis was performed on the quantities derived from the MASK embedding to the two potential references.

In a recent study, Ethayarajh & Jurafsky (2021) attempted to establish a systematic connection between attention flows and XAI. While their work focused on demonstrating that attention flows can be interpreted as Shapley values under certain conditions, they did not account for the non-uniqueness of such flows. It is worth noting that, although maximum flow problem has the unique optimal value, there can be multiple feasible flows that attain this optimal value.

This highlights the need for further research to explore the method to resolve the issue of non-uniqueness in attention flows and its impact on XAI.

3 PRELIMINARIES

We will define preliminary terms and definitions here that will be used in the subsequent sections. This study will primarily utilize these concepts.

3.1 GRAPH NETWORK FLOW

3.1.1 MAXIMUM FLOW

Definition 3.1 (Maximum Flow). *Given a network $G = (V, E, \boldsymbol{u})$, where u_{ij} is the capacity for the edge $(i, j) \in E$, a flow is characterized as a function $f : E \to \mathbb{R}^{\geq 0}$ s.t.*

$$\begin{aligned} &f_{ij} \leq u_{ij} \quad \forall (i,j) \in E \quad \textit{(capacity constraints)} \\ &\sum_{j:(i,j)\in E} f_{ij} - \sum_{j:(j,i)\in E} f_{ji} = 0, \quad \forall i \in V, i \neq s, t \quad \textit{(flow conservation constraints)} \end{aligned} \tag{1}$$

The value of a flow in a given network $G = (V, E, s, t, \boldsymbol{u})$ is established as the sum of flows into the source vertex s, denoted as $\mathrm{val}(f) = \sum_{v:(v,s)} f_{vs}$, and a maximum flow is identified as a feasible flow with the highest attainable value.

Behrooz: Check definition of val(f)

3.1.2 MINIMUM-COST CIRCULATION

Definition 3.2 (Minimum Cost Circulation). *Given a network $G = (V, E, \boldsymbol{u}, \boldsymbol{l}, \boldsymbol{c})$ with $|V| = n$ vertices and $|E| = m$ edges, where c_{ij} is the cost, $l_{i,j}$ and $u_{i,j}$ are respectively the lower and upper capacities for the edge $(i, j) \in E$, a circulation is f function $f : E \to \mathbb{R}^{\geq 0}$ s.t.*

$$\begin{aligned} &l_{ij} \leq f_{ij} \leq u_{ij}, && \forall (i,j) \in E \\ &\sum_{j:(i,j)\in E} f_{ij} - \sum_{j:(j,i)\in E} f_{ji} = 0, && \forall i \in V. \end{aligned} \tag{2}$$

The min-cost circulation problem is to find a circulation f that minimizes the cost function $\sum_{(i,j)\in E} c_{ij} f_{ij}$.

The formulation of minimum-cost circulation problem can be algebraically written as the following linear programming (LP) problem:

$$\min_{\substack{\boldsymbol{B}^\top \boldsymbol{f}=\boldsymbol{0} \\ l_e \le f_e \le u_e \forall e \in E}} \boldsymbol{c}^\top \boldsymbol{f} \quad \text{i.e.} \quad \arg\min_{\substack{\boldsymbol{B}^\top \boldsymbol{f}=\boldsymbol{0} \\ \boldsymbol{l} \le \boldsymbol{f} \le \boldsymbol{u}}} \boldsymbol{c}^\top \boldsymbol{f} \tag{3}$$

where $B_{m\times n}$,is the edge-vertex incidence matrix. For a directed graph, the entries of the matrix $\boldsymbol{B}$ are defined as follows:

$$b_{ev} = \begin{cases} -1, & \text{if vertex } v \text{ is the tail of edge } e, \\ 1, & \text{if vertex } v \text{ is the head of edge } e, \\ 0, & \text{if edge } v \text{ is not incident to vertex } e. \end{cases}$$

3.2 SHAPLEY VALUES

Definition 3.3 (Shapley Values). *Shapley values are defined through value functions ϑ of subsets S of players in a set $N = \{1, 2, \ldots, n\}$. Specifically, the Shapley value $\phi_j(\vartheta)$ of player (or feature) j in a game with total payoff v is given by:*

$$\phi_j(\vartheta) = \sum_{S \subseteq N \setminus \{j\}} \frac{|S|!(n - |S| - 1)!}{n!} (\vartheta(S \cup \{j\}) - \vartheta(S)) \tag{4}$$

Remark 3.4. *Shapley values are the unique linear additive explanation that satisfies four fairness-based axioms (efficiency, symmetry, linearity (additivity), and null player)*

Please see Appendix A for details.

3.3 MULTI-HEAD ATTENTION MECHANISM

Given an input sequence $\boldsymbol{X} \in \mathbb{R}^{t\times d}$, where d is the dimensionality of the model's input vectors, the multi-head self-attention mechanism computes attention weights for each element in the sequence.

- **Linear Transformation**:

  $$\boldsymbol{Q}_i = \boldsymbol{X}\boldsymbol{W}_i^Q, \quad \boldsymbol{K}_i = \boldsymbol{X}\boldsymbol{W}_i^K, \quad \boldsymbol{V}_i = \boldsymbol{X}\boldsymbol{W}_i^V$$

  where $\boldsymbol{Q}_i, \boldsymbol{K}_i, \boldsymbol{V}_i \in \mathbb{R}^{t\times d_k}$, and $\boldsymbol{W}_i^Q, \boldsymbol{W}_i^K, \boldsymbol{W}_i^V \in \mathbb{R}^{d\times d_k}$.

- **Scaled Dot-Product Attention**:

  $$\text{Attention}_i(\boldsymbol{Q}_i, \boldsymbol{K}_i, \boldsymbol{V}_i) = \text{softmax}\left(\frac{\boldsymbol{Q}_i \boldsymbol{K}_i^T}{\sqrt{d_k}}\right) \boldsymbol{V}_i$$

  where the division by $\sqrt{d_k}$ is for scaling. Here Attention_i represent the operation for the i-th head, and the attention scores matrix $\widetilde{\boldsymbol{A}}_i$ for the i-th head is defined as:

  $$\widetilde{\boldsymbol{A}}_i = \text{softmax}\left(\frac{\boldsymbol{Q}_i \boldsymbol{K}_i^T}{\sqrt{d_k}}\right) \tag{5}$$

Here, n is the sequence length, d is the dimensionality of the model's input vectors, and d_k is the dimensionality of the key vectors. The resulting matrix $\widetilde{\boldsymbol{A}}_i$ has dimensions $\widetilde{\boldsymbol{A}}_i \in \mathbb{R}^{t \times t}$, where each element $\widetilde{\boldsymbol{A}}_i(j,k)$ represents the attention weight assigned to the j-th element when processing the k-th element in the sequence.

- **Concatenation and Linear Projection**:

$$\text{MultiHead}(\boldsymbol{X}) = \text{Concat}(\text{Attention}_1, \text{Attention}_2, \ldots, \text{Attention}_h)\boldsymbol{W}_O$$

where $\text{MultiHead}(\boldsymbol{X}) \in \mathbb{R}^{t \times d}$, and $\boldsymbol{W}_O \in \mathbb{R}^{h \cdot d_k \times d}$.

For a transformer architecture with l attention layers, the attention scores at each layer can be defined as multi-head attention score:

$$\widehat{\boldsymbol{A}} = \text{Concat}(\widetilde{\boldsymbol{A}}_1, \widetilde{\boldsymbol{A}}_2, \cdots, \widetilde{\boldsymbol{A}}_h) \in \mathbb{R}^{h \times t \times t} \tag{6}$$

We can extend this concept to a transformer architectures itself as following:

$$\boldsymbol{A} = \text{Concat}(\widehat{\boldsymbol{A}}_1, \widehat{\boldsymbol{A}}_2, \cdots, \widehat{\boldsymbol{A}}_l) \in \mathbb{R}^{l \times h \times t \times t} \tag{7}$$

wehre $\widehat{A}_j \in \mathbb{R}^{h \times t \times t}$ is the multi-head attention score for j-th attention layer.

3.3.1 Attribution Flows

One way to integrate the attention mechanism into XAI is by implementing a flow network representation of the graph. This representation assigns capacities to the edges of the graph that correspond to attention weights or similar values. By implementing a maximum flow algorithm, we can discover the flow from any node in any layer to any input node, which can help us estimate the quantity of attention (attribution) directed towards the input nodes.

To determine the maximum flow from all output nodes to all input nodes, we must utilize the concept of multi-commodity flow. This involves creating a super-source s and a super-target t with a capacity u_∞. The connectivity between layers is established by means of attention weights or similar values, resulting in the formation of a layered network.

To formalize this process, assume we have the transformer architecture with l attention layers , input sequence $X \in \mathbb{R}^{t \times d}$ and its attribution tensor $\boldsymbol{F} \in \mathbb{R}^{l \times t \times t}$. we create the attribution flow matrix $\mathcal{F}$ and its integral version $\widetilde{\mathcal{F}}$ employing either Algorithm 1 or Algorithm 2:

Algorithm 1 Backward Attribution Flow

Require: $\boldsymbol{F}_{l\times t\times t}$: An attribution tensor.
Ensure: Tuple: $(\mathcal{F}, \widetilde{\mathcal{F}})$
  function GET_ATTENTION_FLOW($\boldsymbol{F}$)
    $l, t, \_ \leftarrow \boldsymbol{F}$.shape()
    $\beta_{\min} \leftarrow \min(\boldsymbol{F} > 0)$
    $\beta \leftarrow -\lfloor \log_{10}(\beta_{\min}) \rfloor$
    $\gamma \leftarrow 10^{\beta}$
    $Q_{tl} \leftarrow t * (l+1) + 2$
    $\mathcal{F} \leftarrow$ np.zeros(Q_{tl}, Q_{tl})
    $u_{\infty} \leftarrow t$
    for i in range(t) **do**
      $\mathcal{F}[i+1][0] \leftarrow u_{\infty}$
    end for
    for i in range(t) **do**
      $\mathcal{F}[-1][-i-2] \leftarrow u_{\infty}$
    end for
    for j in range$(0, l)$ **do**
      low, up $\leftarrow t * j + 1, t * (j+1) + 1$
      $\mathcal{F}$[up : up $+ t$, low : low $+ t$] $\leftarrow \boldsymbol{F}[j]$
    end for
    $\widetilde{\mathcal{F}} \leftarrow$ int$(\gamma * \mathcal{F})$
  end function

Algorithm 2 Forward Attribution Flow

Require: $\boldsymbol{F}_{l\times t\times t}$: An attribution tensor.
Ensure: Tuple: $(\mathcal{F}, \widetilde{\mathcal{F}})$
  function GET_ATTENTION_FLOW($\boldsymbol{F}$)
    $l, t, \_ \leftarrow \boldsymbol{F}$.shape()
    $\beta_{\min} \leftarrow \min(\boldsymbol{F} > 0)$
    $\beta \leftarrow -\lfloor \log_{10}(\beta_{\min}) \rfloor$
    $\gamma \leftarrow 10^{\beta}$
    $Q_{tl} \leftarrow t * (l+1) + 2$
    $\mathcal{F} \leftarrow$ np.zeros(Q_{tl}, Q_{tl})
    $u_{\infty} \leftarrow t$
    for i in range(t) **do**
      $\mathcal{F}[i+1][0] \leftarrow u_{\infty}$
    end for
    for i in range(t) **do**
      $\mathcal{F}[-1][-i-2] \leftarrow u_{\infty}$
    end for
    for j in range$(0, l)$ **do**
      low, up $\leftarrow t * j + 1, t * (j+1) + 1$
      $\mathcal{F}$[up : up $+ t$, low : low $+ t$] $\leftarrow \boldsymbol{F}[j]$
    end for
    $\widetilde{\mathcal{F}} \leftarrow$ int$(\gamma * \mathcal{F})$
  end function

Behrooz: check the algorithm accuracy add source and node indices to algorithms

4 METHODS

4.1 NON-UNIQUENESS OF MAXIMUM FLOW

It should be noted that the problem at hand is not strictly convex, which implies that the existence of a unique solution cannot be guaranteed. Additionally, we have discovered that the maximum flow problem for the previously constructed graph does not possess a unique solution either. Specifically, we have identified two feasible maximum flows for a similar graph created using attention flows. These maximum flows can be obtained using two distinct algorithms, namely Algorithms 1 and 2.

Moreover, we know that if any linear programming has two distinct feasible solutions f_1, f_2, any convex combination $\gamma_1 f_1 + \gamma_2 f_2$ where $\gamma_1 + \gamma_2 = 1$ is a new solution. Consequently, the attention flows has infinite feasible solution and therefore the feature attributions and Shapley values resulted from this approach is ill-defined.

5 MAKE MAXIMUM FLOW UNIQUE

As previously discussed, each network inherently possesses a distinct maximum flow value. However, achieving this maximum value is not confined to a singular feasible flow; rather, multiple feasible flows can attain the same maximum. This variability introduces a challenge in discerning the optimal flow solution. To address this challenge, the introduction of regularization terms into the maximum flow problem proves beneficial. Through this regularization, the objective of the linear programming problem defined in Eq. 11 is transformed into a strictly convex function. This transformation ensures the derivation of a unique

and optimal approximate solution, thereby overcoming the issue posed by the non-uniqueness of feasible flows achieving the maximum value.

Here we introduce two main regularization term and the logic behind them: 1. log-barrier regularization 2. ℓ_1 and ℓ_2 regularization.

5.1 LOG-BARRIER REGULARIZATION

We can reformulate the minimum cost circulation problem as:

$$\underset{\substack{\boldsymbol{B}^\top \boldsymbol{f}=\mathbf{0} \\ h(\boldsymbol{f})\leq \mathbf{0}}}{\arg\min}\ \boldsymbol{c}^\top \boldsymbol{f} \tag{8}$$

where $h(\boldsymbol{f}) = (\boldsymbol{f} - \boldsymbol{l})(\boldsymbol{f} - \boldsymbol{u})$.

Defining the log-barrier function $\phi_\mu(\boldsymbol{f}) = -\mu \sum_{e\in E} \log\left(-h(f_e)\right) = \mu \sum_{e\in E} \left(\log\left(f_e - l_e\right) + \log\left(u_e - f_e\right)\right)$, the original problem can, therefor, be approximated using the log barrier function as the following:

$$\underset{\boldsymbol{B}^\top \boldsymbol{f}=\mathbf{0}}{\arg\min}\ \boldsymbol{c}^\top \boldsymbol{f} + \phi_\mu(\boldsymbol{f}) \tag{9}$$

It is easy to observe that as long as $\mu > 0$ and our initial solution is feasible, the barrier function will ensure that any solution we find using an iterative minimization scheme, such as gradient descent or Newton's method, remains feasible. Moreover, one can show that to get an ε-approximate solution to the Eq. 8 it suffices to set $\mu \leq \frac{\varepsilon}{2m}$ and find the optimal solution to the corresponding problem in Eq. 9.(Boyd & Vandenberghe, 2004; Bubeck, 2015)

Finally, observe that the Hessian of the objective function of Eq. 9 at some point $\boldsymbol{f}$ is equal to the Hessian $\nabla^2\phi_\mu(x)$ of the barrier function. The latter turns out to be a diagonal matrix with

$$\left(\nabla^2\phi_\mu(\boldsymbol{f})\right)_{e,e} = \mu\left(\frac{1}{\left(u_e - f_e\right)^2} + \frac{1}{\left(f_e - l_e\right)^2}\right),$$

for each edge e. Thus, the Hessian of our objective function is positive definite (provided $\mu > 0$), which implies that the objective function is strongly convex and therefore have a unique feasible solution.

There are many preceding study about various numerical methods and their convergence rate which we leave the readers for further investigation (Bertsimas & Tsitsiklis, 1997; Boyd & Vandenberghe, 2004; Bubeck, 2015). It is noteworthy to mention that many other barrier functions can be used instead of log-barrier functions which can enhance the convergence rate of numerical solver for the optimization problem. (Cohen et al., 2020; Lee & Sidford, 2020; Axiotis et al., 2022; van den Brand et al., 2021; Chen et al., 2022)

5.1.1 ℓ_1 AND ℓ_2 REGULARIZATION

L1 regularization encourages sparsity in the model, which means it forces some of the feature weights to be precisely zero. This leads to feature selection, where only a subset of the most relevant features is retained in the model, which is particularly useful when dealing with high-dimensional datasets or when there is a suspicion that many features may not contribute significantly to the model's performance. On the other hand, L2 regularization constrains the magnitude of the weights, preventing them from becoming too large. L2 regularization ensures a more stable and generalized model by penalizing overly complex models with large coefficients (Yen et al.).

Therefore, we can regularize the Eq. 8 as:

$$\underset{\substack{\boldsymbol{B}^\top \boldsymbol{f}=\mathbf{0} \\ h(\boldsymbol{f})\leq \mathbf{0}}}{\arg\min}\ \boldsymbol{c}^\top \boldsymbol{f} + \lambda_1 \|\boldsymbol{f}\|_1 + \lambda_2 \|\boldsymbol{f}\|_2^2 \tag{10}$$

where $\lambda_1, \lambda_2 > 0$. It is is easy to show that to get an ε-approximate solution to the Eq. 8 it suffices to set $\max(\lambda_1, \lambda_2) \leq \frac{\varepsilon}{2}\min(1, \frac{1}{m\|\boldsymbol{u}\|_\infty})$ and find the optimal solution to the corresponding problem in Eq. 10.(Boyd & Vandenberghe, 2004; Bubeck, 2015)

6 INSTANTIATION

Te main information to construct our attention flow is the tensor $\mathcal{A}_{L\times T\times T} = [\mathcal{A}_1, \mathcal{A}_2, \cdots, \mathcal{A}_L]$ where $\mathcal{A}_i = \mathbb{A}$ L is the number of attention layers and T is the number of tokens in our sample. In the previous section

REFERENCES

Samira Abnar and Willem Zuidema. Quantifying Attention Flow in Transformers, May 2020.

Kyriakos Axiotis, Aleksander Madry, and Adrian Vladu. Faster Sparse Minimum Cost Flow by Electrical Flow Localization. In *2021 IEEE 62nd Annual Symposium on Foundations of Computer Science (FOCS)*, pp. 528–539, Denver, CO, USA, February 2022. IEEE. ISBN 978-1-66542-055-6. doi: 10.1109/FOCS52979.2021.00059.

Dimitris Bertsimas and John N. Tsitsiklis. *Introduction to Linear Optimization*. Athena Scientific Series in Optimization and Neural Computation. Athena Scientific, Belmont, Mass, 1997. ISBN 978-1-886529-19-9.

Stephen P. Boyd and Lieven Vandenberghe. *Convex Optimization*. Cambridge University Press, Cambridge, UK ; New York, 2004. ISBN 978-0-521-83378-3.

Sébastien Bubeck. Convex Optimization: Algorithms and Complexity, November 2015. Comment: A previous version of the manuscript was titled "Theory of Convex Optimization for Machine Learning".

Li Chen, Rasmus Kyng, Yang P. Liu, Richard Peng, Maximilian Probst Gutenberg, and Sushant Sachdeva. Maximum Flow and Minimum-Cost Flow in Almost-Linear Time, April 2022.

Michael B. Cohen, Yin Tat Lee, and Zhao Song. Solving Linear Programs in the Current Matrix Multiplication Time, October 2020. Comment: STOC 2019, JACM 2020.

Kawin Ethayarajh and Dan Jurafsky. Attention Flows are Shapley Value Explanations, May 2021. Comment: ACL 2021.

Yin Tat Lee and Aaron Sidford. Solving Linear Programs with Sqrt(rank) Linear System Solves, August 2020. Comment: The merged version of abs/1312.6677 and abs/1312.6713. It contains several new results beyond these prior submissions, including a nearly optimal self-concordant barrier and its relation to Lewis weight.

Lloyd S. Shapley. A Value for N-Person Games. Technical report, RAND Corporation, March 1952.

Jan van den Brand, Yin Tat Lee, Yang P. Liu, Thatchaphol Saranurak, Aaron Sidford, Zhao Song, and Di Wang. Minimum Cost Flows, MDPs, and ℓ_1-Regression in Nearly Linear Time for Dense Instances, August 2021.

Ian E H Yen, Kai Zhong, Cho-Jui Hsieh, Pradeep Ravikumar, and Inderjit S Dhillon. Sparse Linear Programming via Primal and Dual Augmented Coordinate Descent.

A APPENDIX

You may include other additional sections here.

A.1 DEFINITIONS

A.1.1 MULTI-COMMODITY MAXIMUM FLOW

The multi-commodity flow problem is an important variant of the maximum flow problem. This problem involves multiple source-sink pairs, unlike the standard maximum flow problem which only considers one source and one sink. The goal is to find multiple feasible flows, denoted by $f^1(\cdot,\cdot), \ldots, f^r(\cdot,\cdot)$, where each $f^k(\cdot,\cdot)$ represents a feasible flow from the source s_k to the sink t_k. The objective is to ensure that all capacity constraints are satisfied, which are represented by the equation:

$$\sum_{k=1}^{r} f^k(i,j) \leq u(i,j) \quad \forall (i,j) \in E$$

Such a flow is known as a "multi-commodity" flow. A multi-commodity maximum flow problem is to maximize the function $\sum_{k=1}^{r} \sum_{v:(v,s_k)} f^k_{s_k v}$.

To solve the problem of multi-commodity maximum flow, we can simplify it by transforming it into a standard maximum flow problem. This can be achieved by introducing two new nodes - a "super-source" node s and a "super-sink" node t. The "super-source" node s should be connected to all the original sources s_i through edges of finite capacities, while the "super-sink" node t should be connected to all the original sinks t_i with edges of finite capacities:

- Each outgoing edge from the "super-source" node s to each source node s_i gets assigned a capacity that is equal to the total capacity of the outgoing edges from the source node s_i.
- Each incoming edge from an original "super-sink" node t to each sink node t_i gets assigned a capacity that is equal to the total capacity of the incoming edge to the sink node t_i.

It is easy to see that the maximum flow from s to t is equivalent to the maximum sum of flows in a feasible multi-commodity flow in the original network.

A.1.2 MINIMUM COST CIRCULATION PROBLEM

The formulation of minimum-cost circulation problem can be written as the following linear programming (LP) problem:

$$\min_{\substack{\boldsymbol{B}^\top \boldsymbol{f}=\boldsymbol{0} \\ l_e \leq f_e \leq u_e \forall e \in E}} \boldsymbol{c}^\top \boldsymbol{f} \quad \text{i.e.} \quad \arg\min_{\substack{\boldsymbol{B}^\top \boldsymbol{f}=\boldsymbol{0} \\ \boldsymbol{l} \leq \boldsymbol{f} \leq \boldsymbol{u}}} \boldsymbol{c}^\top \boldsymbol{f} \tag{11}$$

It is easy to show that it satisfies strong duality with dual problem:

$$\max_{\substack{\boldsymbol{B}\boldsymbol{y}+\boldsymbol{s}_l-\boldsymbol{s}_u=\boldsymbol{c} \\ \boldsymbol{s}_l,\boldsymbol{s}_u \in \mathbb{R}^E_{\geq 0}}} -\boldsymbol{s}_u^\top \boldsymbol{u} + \boldsymbol{s}_l^\top \boldsymbol{l} \tag{12}$$

where the Lagrangian ($\mathcal{L}$) is defined as:

$$\mathcal{L}(\boldsymbol{f}, \boldsymbol{y}, \boldsymbol{s}_u, \boldsymbol{s}_l) = \boldsymbol{c}^\top \boldsymbol{f} + \boldsymbol{y}^\top (\boldsymbol{B}^\top \boldsymbol{f}) + \boldsymbol{s}_u^\top (\boldsymbol{u} - \boldsymbol{f}) + \boldsymbol{s}_l^\top (\boldsymbol{f} - \boldsymbol{l})$$

Since our primal LP problem is convex, the KKT conditions provide both necessary an sufficient conditions to find optimal primal and dual variables for the minimum cost circulation problem as follows:

1. **Stationarity**:
$$\nabla \mathcal{L}(\boldsymbol{f}, \boldsymbol{y}, \boldsymbol{s}_u, \boldsymbol{s}_l) = \boldsymbol{c} - \boldsymbol{B}\boldsymbol{y} + \boldsymbol{s}_u - \boldsymbol{s}_l = \boldsymbol{0}$$
2. **Complementary Slackness**:
$$\boldsymbol{s}_u^\top (\boldsymbol{u} - \boldsymbol{f}) = \boldsymbol{0} : s_e^+ \cdot (u_e - f_e) = 0, \quad \forall e \in E \quad \text{(for upper-bound capacity constraints)}$$
$$\boldsymbol{s}_l^\top (\boldsymbol{f} - \boldsymbol{l}) = \boldsymbol{0} : s_e^- \cdot (f_e - l_e) = 0, \quad \forall e \in E \quad \text{(for lower-bound capacity constraints)}$$
3. **Primal Feasibility**:
$$\boldsymbol{B}^\top \boldsymbol{f} = \boldsymbol{0} \quad \text{(flow conservation constraint)}$$
$$\boldsymbol{l} \leq \boldsymbol{f} \leq \boldsymbol{u} : \quad l_e \leq f_e \leq u_e, \quad \forall e \in E \quad \text{(capacity constraints)}$$
4. **Dual Feasibility**:
$$\boldsymbol{s}_u, \boldsymbol{s}_l \geq \boldsymbol{0} \quad \text{(non-negativity of slack variables)}$$

Here, $\boldsymbol{y}$ represents the dual variables corresponding to the equality constraint ($\boldsymbol{B}^\top \boldsymbol{f} = \boldsymbol{0}$), and $\boldsymbol{s}_u$ and $\boldsymbol{s}_l$ represent the dual variables corresponding to the upper and lower inequality constraints ($l_e \leq f_e \leq u_e$), respectively.

A.1.3 SHAPLEY VALUES

The Shapley value, introduced by Shapley (1952), concerns the cooperative game in the coalitional form (N, ϑ), where N is a set of n players and $\vartheta : 2^N \to \mathbb{R}$ with $\vartheta(\emptyset) = 0$ is the characteristic function. In the game, the marginal contribution of the player i to any coalition S with $i \notin S$ is considered as $\vartheta(S \cup i) - \vartheta(S)$. These Shapley values are the only constructs that jointly satisfy the Efficiency, Symmetric, Null Player and Additivity axioms:

- **Efficiency:** The Shapley values must add up to the total value of the game, which means $\sum_{i \in N} \phi_i(\vartheta) = \vartheta(N)$
- **Symmetry:** If two players are equal in their contributions to any coalition, they should receive the same Shapley value. Mathematically, if $\vartheta(S \cup \{i\}) = \vartheta(S \cup \{j\})$ for all $S \subseteq N \setminus \{i, j\}$, then $\phi_i(\vartheta) = \phi_j(\vartheta)$.
- **Null Player:** If a player has no impact on any coalition, their Shapley value should be zero. Mathematically, if $\vartheta(S \cup \{i\}) = \vartheta(S)$ for all $S \subseteq N \setminus \{i\}$, then $\phi_i(\vartheta) = 0$.
- **Additivity:** If two games are combined into a larger game, the Shapley value of a player in the combined game is the sum of their Shapley values in the individual games.

By considering these axioms, the attribution of a player i is uniquely given by:

$$\phi_j(\vartheta) = \sum_{S \subseteq N \setminus \{j\}} \frac{|S|!(n - |S| - 1)!}{n!} (\vartheta(S \cup \{j\}) - \vartheta(S))$$