Beyond Shapley: Efficient Computation of Asymmetric Shapley Values

Ezequiel Companeetz[1], Santiago Cifuentes[2], Sergio Abriola[2]

[1] Departamento de Computación, Facultad de Ciencias Exactas y Naturales, UBA
[2] Instituto de Ciencias de la Computación (ICC), CONICET

Abstract. We address the problem of explainability in machine learning models through feature attribution methods. In particular, we consider a variant of Shapley values known as Asymmetric Shapley Values (ASV), which enables the incorporation of causal knowledge into model-agnostic explanations through the use of a causal graph. We show that in certain contexts in which the computation of SHAP is #P-hard, the exact computation of ASV can be done in polynomial time. To extend this algorithmic result, we introduce a notion of equivalence classes over the topological orderings of the underlying causal graph, which is useful to reduce the time to compute ASV. In particular, we present a polynomial-time algorithm (in the number of equivalence classes) to compute it whenever the causal graph is a rooted directed tree. Finally, we develop an algorithm for approximating ASV in arbitrary causal DAGs which relies on a procedure to sample topological orderings uniformly at random. To implement this sampling mechanism we leverage known algorithms as well as simpler alternatives. Our experimental results demonstrate the practical viability of the proposed approach in realistic causal structures.

Keywords: Explainability, Asymmetric Shapley Values (ASV), SHAP

Más allá de Shapley: Algoritmos Eficientes para calcular los Shapley Values Asimétricos

Abstract. Abordamos el problema de la explicabilidad en modelos de aprendizaje automático mediante métodos de *feature attribution*. En particular, consideramos una variante de los *Shapley values* conocida como *Asymmetric Shapley Values* (ASV), que permite incorporar conocimiento causal en explicaciones *model-agnostic* mediante el uso de un grafo causal. Mostramos que, en ciertos contextos en los que el cálculo de SHAP es #P-hard, el cálculo exacto de ASV puede realizarse polinomial. Para extender este resultado algorítmico, introducimos una noción de clases de equivalencia sobre los ordenamientos topológicos del grafo causal subyacente, lo cual resulta útil para reducir el tiempo de cómputo de ASV.

En particular, presentamos un algoritmo en tiempo polinomial (en el número de clases de equivalencia) para calcularlo cuando el grafo causal es un árbol dirigido enraizado. Finalmente, desarrollamos un algoritmo para aproximar ASV en DAGs causales arbitrarios, basado en un procedimiento que muestrea ordenamientos topológicos uniformemente al azar. Para implementar este mecanismo de muestreo consideramos algoritmos conocidos así como alternativas más simples. Nuestros resultados experimentales demuestran la viabilidad práctica del enfoque propuesto en estructuras causales realistas.

Keywords: Explicabilidad, Asymmetric Shapley Values (ASV), Shap, Órdenes topológicos

1 Introduction

The field of XAI (*Explainable Artificial Intelligence*) aims to develop methods capable of making model decisions more interpretable (Mersha et al., 2024). Within this field, one of the most influential approaches is feature attribution, where each input variable is assigned a score reflecting its contribution to the prediction. In this line of work, SHAP is one of the most well-known methods (Lundberg & Lee, 2017). It is based on the Shapley values, a concept originating in cooperative game theory (Shapley et al., 1953). Intuitively, Shapley values provide a fair way to distribute the total value achieved by a coalition among its participants, according to their average marginal contribution when joining. Translated to machine learning, features play the role of players, and the value assigned to each feature aims to quantify its contribution to the model's prediction for a given instance.

Despite its popularity, SHAP has important limitations. On the one hand, depending on the family of ML models under consideration and the underlying distribution of the data, the computation of the Shapley values can be tractable or not. For example, it is well-known that for product distributions the complexity of computing these scores is equivalent to the complexity of computing the average value of the model (Van den Broeck et al., 2022). In particular, this implies that there exist efficient algorithms for the case where the models are given, for example, by decision trees or by restricted families of circuits (Arenas et al., 2023), while the problem is intractable whenever the models are more expressive, as in the case of neural nets. For more general families of distributions the problem becomes much harder: for instance, when considering Naive Bayes distributions, the computation of this score is intractable already when considering trivial models that only depend on one feature (Van den Broeck et al., 2022).

On the other hand, there are also concerns about the quality of the explanations it produces. In particular, it is not always clear how to interpret, in the context of AI models, the axioms inherited from game theory, and several works point out that these explanations may fail to adequately capture relationships

such as dependence, relevance, or causality between variables (Fryer et al., 2021; Huang & Marques-Silva, 2023; Marques-Silva, 2023). More generally, there is no absolute consensus on what should be considered an "explanation" in AI: many current techniques are pragmatic approximations to a much broader problem related to understanding, causality, and human interpretability (Lipton, 2017; Miller, 2019).

The Asymmetric Shapley Values (ASV from now on) are a proposal introduced by Frye et al., 2019 to overcome some of these limitations. This variant incorporates causal information into the computation of attribution scores by restricting attention to permutations of variables that respect a causal graph. In this way, instead of treating all features symmetrically, priority is given to the more fine-grained ones (in the sense that the information they provide is not present in the other variables). This can produce explanations that are more faithful to the data-generating process and can help better detect phenomena such as bias or discrimination. Moreover, although ASV was primarily proposed to improve explanatory quality, it naturally raises the question of whether this additional restriction may also provide computational benefits.

Therefore, in this paper we address the problem of efficiently computing the ASV scores, which was not explored thoroughly in previous literature. We will prove that for some families of models for which computing the Shapley values is intractable there are still polynomial time algorithms for computing the ASV. Inspired by this initial result, we develop an algorithm to compute the ASV exactly in more general contexts by exploiting the fact that many permutations of the features can be grouped in a unique equivalence class to simplify computations. Finally, we also propose a flexible approximation scheme that allows to approximate the ASV as long as it is possible to (1) Sample a topological order of the causal graph uniformly at random, and (2) Sample a data point with respect to the data distribution after conditioning a subset of the features to have specific values. We instantiate all these general algorithms with particular examples of causal graphs and machine learning models and show experimentally that they are efficient.

2 Definitions

Let X be a finite set of features. A (binary) *entity* over X is a mapping $e : X \to \{0, 1\}$ assigning a value to every feature from X. We denote by $\texttt{ent}(X)$ the set of all entities, and we will assume that some probability distribution is provided over this set. A (binary) classifier is a mapping $M : \texttt{ent}(X) \to \{0, 1\}$. We restrict to both binary entities and classifiers only for simplicity: all the results and algorithms we obtain can be extended to more general cases straightforwardly.

Given a model M and an entity e, a *feature attribution score* is a mapping $\phi_{M,e} : X \to \mathbb{R}$. Intuitively, the value $\phi_{M,e}(x)$ indicates the relevance of feature x with respect to prediction $M(e)$. As mentioned before, one of the most studied feature attribution schemes is the SHAP-SCORE (Lundberg & Lee, 2017), which is theoretically grounded on the *Shapley values* (Shapley et al., 1953) from co-

operative game theory. In that context, the Shapley values come up as the "most fair" way to distribute some capital earned during a team game. More precisely, given a finite set of players $\mathcal{I}$ and a *characteristic function* $\nu : \mathcal{P}(\mathcal{I}) \to \mathbb{R}$ describing, for each subset of the players, the reward that such coalition would obtain in some fixed game; the shapley values are the unique assignment $\varphi_\nu : \mathcal{I} \to \mathbb{R}$ of reward to each player i satisfying the following desired properties:

1. Efficiency: All the reward is distributed, i.e. $\sum_{i \in \mathcal{I}} \varphi_\nu(i) = \nu(\mathcal{I}) - \nu(\emptyset)$.
2. Symmetry: If $\nu(S \cup \{i\}) = \nu(S \cup \{j\})$ for every $S \subseteq \mathcal{I} \setminus \{i, j\}$, then $\varphi_\nu(i) = \varphi_\nu(j)$.
3. Null Player: If $\nu(S \cup \{i\}) = \nu(S)$ for every $S \subseteq \mathcal{I}$, then $\varphi_\nu(i) = 0$.
4. Linearity: For any pair of characteristic functions ν_1, ν_2 and $a \in \mathbb{R}$ it holds that $\varphi_{a\nu_1+\nu_2}(i) = a\varphi_{\nu_1}(i) + \varphi_{\nu_2}(i)$[3].

Moreover, the Shapley values have an exact closed form: if we refer to players with indices in the range $\{1, \ldots, n\}$, we can write

$$\varphi_\nu(i) = \frac{1}{|\mathcal{I}|!} \sum_{\pi \in perm(\mathcal{I})} [\nu(\pi_{<i} \cup \{i\}) - \nu(\pi_{<i})] \tag{1}$$

where $perm(\mathcal{I})$ denotes the set of all permutations of the number from 1 to n and $\pi_{<i}$ denotes the set of indices before i in the permutation π (i.e. $\pi_{<i} = \{j : \pi(j) < \pi(i)\}$). Fundamentally, Eq. (1) is summing across all possible ways of ordering the players, considering the impact of the inclusion of player i to the reward of the game when adding the players according to each of these orderings.

In the context of machine learning explainability, the Shapley values are instantiated by thinking of the features as players and considering the characteristic function

$$\nu_{M,e}(S) = \mathbb{E}[M(e')|S, e] = \sum_{e' \in \mathtt{ent}(X)} M(e') \Pr[e'|\{e'(x) = e(x) : x \in S\}] \tag{2}$$

which represents the expected value of the model M when restricting the features from S to have the value dictated by entity e. Intuitively, if some prediction $M(e) = 1$ depends strongly on the fact that some feature x has value $e(x) = 1$, then it will hold that $\nu_{M,e}(S)$ is close to 1 if $x \in S$ and far from 1 otherwise. For example, if the model M merely outputs the value of x (i.e. $M(e) = e(x)$) and we assume that the value of x is independent of the other features and uniformly distributed between 0 and 1, then $\nu_{M,e}(S) = 1$ if $x \in S$ while $\nu_{M,e}(S) = \frac{1}{2}$ otherwise. Thus, it is expected that $\nu_{M,e}$ will give a bigger value to those features more relevant for the prediction.

As mentioned in the introduction, the Shapley values may fail to distinguish features that are heavily correlated. For example, if some feature j depends deterministically of some other feature i it can happen that both features receive

[3] The characteristic function $a\nu_1 + \nu_2$ is naturally defined as $(a\nu_1 + \nu_2)(S) = a\nu_1(S) + \nu_2(S)$.

the same score, even though i is intuitively more relevant for *any* model and prediction. Note that the *Symmetry* property essentially implies that this should always happen whenever two features share the same information. Thus, to avoid this problem it is somewhat necessary to relax that axiom and define a different score. This is the idea underlying the *Asymmetric Shapley Values*.

Definition 1 (Asymmetric Shapley Values (Frye et al., 2019)). *Given a model M, an entity e, a feature x and some weight function $w : perm(X) \to \mathbb{R}_{\geq 0}$ satisfying $\sum_{\pi \in perm(X)} w(\pi) = 1$, we define the Asymmetric Shapley Value of x as*

$$\phi^{ASV}_{M,e}(x) = \sum_{\pi \in perm(X)} w(\pi) \left[\nu_{M,e}(\pi_{<i} \cup \{i\}) - \nu_{M,e}(\pi_{<i})\right]$$

where $\nu_{M,e}$ is defined in Eq. (2)

This definition is derived from the corresponding notion of asymmetric shapley values from cooperative game theory. The function w is useful for weighting some orderings above others. For example, if some feature j is a function of i it is possible to introduce this fact into the score by assigning weight zero to all orderings in which i comes after j. Note that the Shapley values are obtained by taking $w(\pi) = \frac{1}{|X|!}$, and we will refer to this particular case as $\phi^{Shap}_{M,e}$.

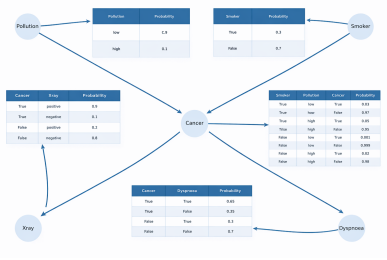
Frye et al., 2019 proposes to define the weight function w by considering a *causal DAG* C over the features. More precisely, C is a directed acyclic graph whose nodes are the set of features X and where an edge directed from i to j indicates that j depends on i. Then, they define the weight function

$$w_C(\pi) = \frac{\mathbb{I}_{\{\pi \text{ is a toposort}\}}}{|topos(C)|} \tag{3}$$

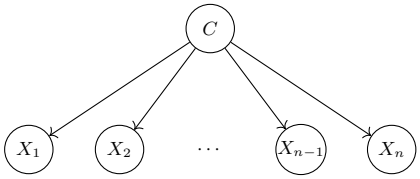
where $topos(C)$ is the set of all topological orderings of C. This weight function generalizes the idea mentioned earlier of considering only those orderings consistent with the known causality of the features. In the rest of this paper we will be interested in developing efficient algorithms for computing the ASV from Def. 1 whenever the weight function is given by Eq. (3).

Moreover, we will consider that the distribution of the space of entities is described by a Bayesian Network N. We refer to the reader to standard references for their precise formal definitions (Pearl, 1986) while here we describe them conceptually: a bayesian network N defines a distribution for $\texttt{ent}(X)$ with a DAG whose nodes are the features from X. Every node x has a set of parents $p(x) \subseteq X$, and the network provides, for each way of assigning the features in $p(x)$, the conditional distribution of x. See Figure 1a for an example. Given N and any entity e it is possible to compute the probability of e by traversing the DAG of N in any topological sort multiplying the corresponding conditional probabilities.

One of the simplest types of Bayesian Networks is the Naive Bayes, which was already mentioned in regard to intractability results for the Shapley values. This network has a particular structure, in which a single node is parent to all the others and no other pairs of nodes are connected. See Figure 1b for an example.



(a) A cancer prediction Bayesian network.



(b) A Naive Bayes network.

Fig. 1: Examples of Bayesian network structures.

A key point for our purposes is that computing the characteristic function (from Eq. (2)) under a Bayesian network distribution requires probabilistic inference. Since exact inference in Bayesian networks is #P-HARD [4] in general (Cooper, 1990), we focus on restricted networks such as polytrees, where inference can be performed in polynomial time.

Sometimes we will identify the network itself as a causal graph. Thus, the network will provide both the distribution necessary to compute the characteristic function $\nu_{M,e}$ and the causal graph necessary to define the weight function w^C. This identification is not always appropriate. In general, a Bayesian network represents a joint probability distribution and its conditional independencies, but its directed edges do not necessarily encode causal relations. Therefore, using the same graph as both a probabilistic model and a causal graph is only justified when the network is assumed to reflect the underlying causal structure. Otherwise, this should be understood merely as a modeling simplification.

3 Theoretical Results

3.1 Tractability of ASV

As mentioned before, the computation of the Shapley Values is intractable (in particular, #P-HARD) even for trivial models if the underlying distribution is represented by a Naive Bayes network. We show that this is not the case for the ASV whenever we use the bayesian network as the causal graph[5].

Theorem 1. *Let $\mathcal{F}$ be any family of models. Then, the ASV can be computed in polynomial time for the family $\mathcal{F}$ under Naive Bayes distributions (where*

[4] #P is a class of *counting* problems. The #P-hard problems from #P are the hardest problems in this class, and it is known that if they can be solved in polynomial time then P = NP. Then, under the usual theoretical assumption, #P-hard problems cannot be solved efficiently.

[5] Due to space limitations, we omit all proofs and defer the complete arguments to the supplementary material.

the causal graph is the network itself) if and only if the Shapley values can be computed in polynomial time for the family $\mathcal{F}$ under product distributions[6].

Theorem 1 shows that, in contrast to the Shapley values, the ASV can be computed under Naive Bayes distributions for simple models such as decision trees. Intuitively, the reason for this is that the causality imposed by the distribution is mimicked by the way that only certain permutations are considered, and thus many symmetries can be exploited. We believe this motivates the idea of studying the complexity of computing the ASV whenever the distribution is also used as the causal graph.

3.2 Equivalence classes of topological sorts

The Shapley values can also be written in closed form as

$$\phi^{Shap}_{M,e}(x) = \sum_{S \subseteq X \setminus \{x\}} \frac{|S|!(|X| - |S| - 1)!}{|X|!} \left[\nu_{M,e}(S \cup \{x\}) - \nu_{M,e}(S)\right]$$

This rewriting is obtained by observing that many permutations give rise to the same set $\pi_{<x}$: more precisely, if $|\pi_{<x}| = s$ there are $s!(|X|-s-1)!$ permutations π' satisfying $\pi'_{<x} = \pi_{<x}$. Computing the Shapley values using this formula reduces the number of times that $\nu_{M,e}$ has to be computed, although the number of calls remains exponential on the number of features.

We will follow a similar strategy to compute the ASV. We define a notion of equivalence between permutations.

Definition 2 (Equivalence relation for permutations). *Let C be a causal graph for features X and fix some feature $x \in X$. We define an equivalence relation $\sim_x$ over $topos(C)$ as*

$$\pi^1 \sim_x \pi^2 \iff \pi^1_{<x} = \pi^2_{<x}.$$

Let Λ^x_C be the set of equivalence classes. It is immediate from the definition that

$$\phi^{ASV}_{M,e}(x) = \frac{1}{|topos(C)|} \sum_{[\pi] \in \Lambda^x_C} |[\pi]| \left[\nu_{M,e}(\pi_{<x} \cup \{x\}) - \nu_{M,e}(\pi_{<x})\right] \tag{4}$$

Therefore, instead of invoking $\nu_{M,e}$ $2|topos(C)|$ times it is enough to compute it $2|\Lambda^x_C|$ times. This value can be at most $2^{|X|}$, but the hope is that in practical scenarios it is smaller. From now on, we will restrict to the case in which causal graphs are given by *rooted directed trees*, polytrees with nodes that have one or fewer parents. In this case, note that we can split the nodes in three groups with respect to x: those that are descendants of x, those that are ancestors of x, and the *unrelated* ones. Observe that descendant/ancestors are never/always in the

[6] With our notation, a product distribution over $\texttt{ent}(X)$ is a distribution that factorizes as $\Pr[e] = \prod_{x \in X} \Pr[x = e(x)]$.

set $\pi_{<x}$, for any x. Thus, the structure of the equivalence classes depends only on the unrelated nodes. See Figure 2 for a visual explanation.

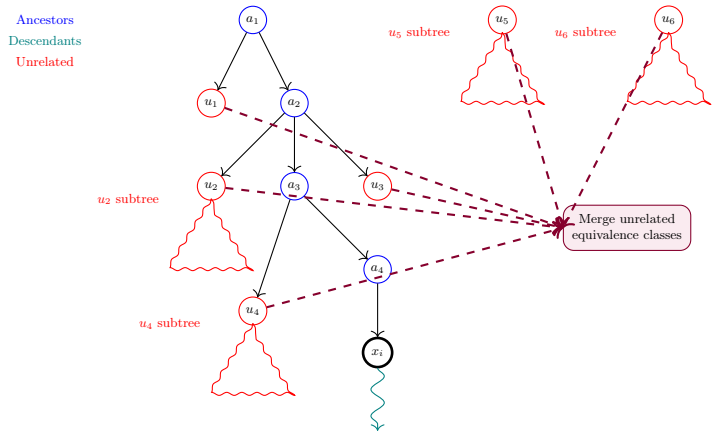


Fig. 2: Graph partition into ancestors, descendants and unrelated nodes with respect to x_i, the feature for which we want to calculate the ASV. The unrelated rooted subtrees can be processed independently and then merged to construct the full equivalence classes.

We have a general bound on the size of Λ_C^x depending only on the causal graph C:

Proposition 1. *Let C be a causal graph shaped as a rooted directed tree with l leaves and depth h. Then, it holds that $|\Lambda_C^x| \leq h^l + 1$*

Proposition 1 indicates that the number of equivalence classes will be small when the tree is shallow. From the point of view of the causal graph, this corresponds to the case in which there aren't long causal dependencies between the features. In particular, there will be many features that are mutually independent.

For the case of directed trees we further develop an algorithm able to compute the set Λ_C^x as well as the number of permutations inside each class (which is required to compute Eq. (4)). The complexity of the algorithm is polynomial on $|\Lambda_C^x|$.

Proposition 2. *There is an algorithm able to compute $|[\pi]|$ for each $[\pi] \in \Lambda_C^x$ given feature x and the causal graph C (shaped as a rooted directed tree) on time polynomial on $|C|$ and $|\Lambda_C^x|$.*

We note that once the set Λ_C^x is computed it is possible to find any value $\phi_{M,e}^{ASV}(x)$ in time linear on $|\Lambda_C^x|$. Thus, even though this step can be costly (because the bound from Proposition 1 implies that the algorithm may take exponential time on the size of the causal graph) is must be done only once. Moreover,

it is independent of the model M, and thus it can be reused even when the model is updated.

3.3 Approximating ASV through sampling

A simple way to compute $\phi^{Shap}_{M,e}(x)$ consist on sampling a small subset $K \subseteq perm(X)$ at random and then approximating the score as

$$\phi^{Shap}_{M,e}(x) \sim \frac{1}{|K|} \sum_{\pi \in K} [\nu_{M,e}(\pi_{<x} \cup \{x\}) - \nu_{M,e}(\pi_{<x})]$$

For the ASV the analogous procedure consists on sampling topological orderings at random. In the next statement we formalize this strategy. Moreover, we observe that the algorithm is robust to error both at the level of the sampling procedure and the computation of $\nu_{M,e}$.

Proposition 3. *Assume access to a procedure $\mathcal{P}_{samp}$ that samples topological orderings from a DAG with a distribution ε_{samp}-close to the uniform one*[7]*, and to a procedure $\mathcal{P}_\nu$ that computes, given any subset of features $S \subseteq X$, a ε_ν-approximation of the value $\nu_{M,e}(S)$. Then, it is possible to compute, with high probability, a $O(\varepsilon_{samp} + \varepsilon_\nu)$-approximation of $\phi^{ASV}_{M,e}(x)$ using $O(1/\varepsilon_\nu^2)$ calls to $\mathcal{P}_{samp}$ and $\mathcal{P}_\nu$.*

Proposition 3 states that we only need to be able to sample the topological orderings *almost* uniformly to compute the ASV. Also, it shows that if we approximate $\nu_{M,e}$ the overall *additive*-error of the estimation of $\phi^{ASV}_{M,e}(x)$ is also controlled. Note that implementing the procedure $\mathcal{P}_\nu$ is often easy: by definition $\nu_{M,e}(S)$ is the expected value of a random variable, and thus if we can sample from the set of entities after conditioning that $s = e(s)$ for every $s \in S$ we are done. This is the case for the product distribution, but also for any bayesian networks whose underlying graph is a polytree, because then the inference process can be implemented in polynomial time.

Now we describe two ways to implement the procedure $\mathcal{P}_{samp}$. The first one can be understood by looking at Figure 3. The idea is that we can sample a topological ordering uniformly using a recursive procedure in which at each step we pick the left-most node of the ordering. In particular, in the beginning we must pick one of the sources of the causal graph (i.e. one of the nodes with in-degree 0). We should weight them according to the number of topological orderings that start with each respective node, and thus we effectively reduce the problem of sampling a topological ordering to the problem of counting the number of topological orderings. Even though this problem is intractable in general (Brightwell & Winkler, 1991), there are heuristic algorithms that can approximate this value or even exact algorithms for restricted cases (Kangas et al., 2016).

[7] To measure distance between distributions we use the 1-norm, i.e. given two distributions $\mathcal{D}_1$ and $\mathcal{D}_2$ over a finite domain Ω we define the distance between $\mathcal{D}_1$ and $\mathcal{D}_2$ as $||\mathcal{D}_1 - \mathcal{D}_2||_1 = \sum_{o \in \Omega} |\mathcal{D}_1(o) - \mathcal{D}_2(o)|$.

In particular, we develop a simple algorithm to count topological orders applicable to polytrees whose complexity depends on the maximum degree of a node. This algorithm computes the number of topological orderings of a polytree by performing a DFS over its underlying undirected graph and recursively combining the results of each node's unvisited neighbors. Instead of only counting the number of orderings of each subtree, the algorithm keeps track of the positions in which a node may appear, together with how many topological orderings realize each position (this additional information is necessary to correctly handle nodes with multiple parents). At each step, the algorithm enumerates the valid relative orders of the current node and its unvisited neighbors, combines the recursive results of those neighbors, and merges equal positions. In this way, it computes all possible placements of the node and the corresponding number of topological orderings, yielding an exact counting algorithm for arbitrary polytrees.

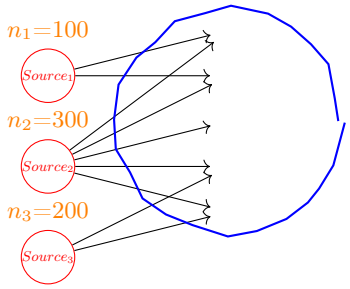


Fig. 3: Illustration of the first step of the sampling algorithm. The candidate source nodes that may appear first in the ordering are shown in red. Each source node s is annotated with the number of topological orderings n_s in which it appears first. We will assign a probability to each s as $n_s/topos(C)$.

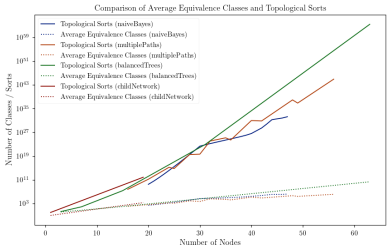
Alternatively, we consider known algorithms from the literature able to approximate uniform sampling of topological orderings from arbitrary DAGs. We implement an algorithm based on random walks (Bubley & Dyer, 1999) with a polynomial (although of high degree) complexity. We note that there are more modern and asymptotically better algorithms (Huber, 2006), but we believe that the hidden constants in them make them inefficient for the input sizes we consider.

4 Experimental results

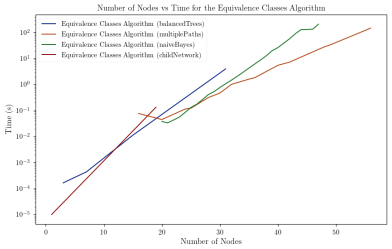
We evaluate the proposed methods by comparing them against naive baselines in terms of computational cost and scalability. Experiments were conducted on two real Bayesian networks, *Cancer* and *Child*, obtained from the `bnlearn` repository. Both were used as data distributions and as causal DAGs. The *Child* network was modified into a polytree by removing edges that introduce cycles in the underlying undirected graph, where we computed the new conditional tables

through marginalization. Datasets were generated by sampling from these networks, and decision trees were trained accordingly. For ASV computation, the prediction variable was removed from the network to avoid trivial inference. To analyze the performance of the algorithms that compute the equivalence classes of topological sorts we also created artificial networks shaped as (1) Balanced binary trees, (2) Naive-bayes distributions, and (3) Trees composed by merging directed paths on their root (what we call `multiplePaths`).[8]

Equivalence Classes vs. Topological Orders We compare the naive ASV computation based on iterating all topological orders with our approach based on equivalence classes. The key quantities are the sizes of $topos(G)$ and Λ_G^x, as well as the cost of constructing each set.



(a) Number of equivalence classes vs. topological orders.



(b) Runtime for computing equivalence classes.

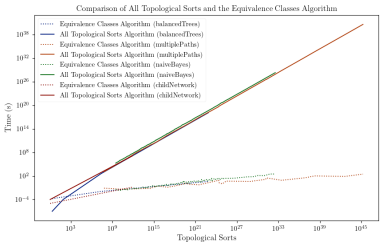
Figure 4a shows that the number of equivalence classes grows significantly slower than the number of topological orders. For instance, in *Child*, the graph has 7.41×10^{11} topological orders but only around 2003 equivalence classes on average[9]. Similar gaps are observed in balanced trees, where the difference can reach several orders of magnitude. At the same time, the cost of computing equivalence classes remains moderate for graphs of practical size, typically under a few seconds. For example, in *Child*, all equivalence classes are computed in less than one second. This reduction directly decreases the number of evaluations of the characteristic function, making the equivalence class approach substantially more efficient.

Performance of Equivalence Class Algorithms We now analyze the cost of computing equivalence classes. As shown in Figure 4b, the computation remains efficient for moderate graph sizes, typically under a few seconds. However, as graph size increases, the runtime grows alongside the number of equivalence classes. We can see in Figure 5b that the runtime is strongly correlated with the number of equivalence classes. This is useful not only for explaining the observed growth,

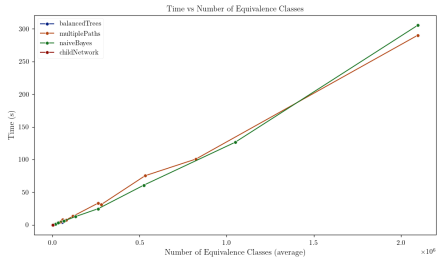
[8] All experiments were run on an Intel i5-7500 CPU @ 3.40GHz with 16 GB of RAM, using Ubuntu 22.04 and Python 3.12. The main libraries used were `pgmpy`, `sklearn`, `networkx`, and `shap`.

[9] We mean the average value of $|\Lambda_G^x|$ over the set of features.

but also from a practical point of view. Since we also have a procedure to count the number of equivalence classes beforehand, we can use it as a rough predictor of the computational cost of the exact algorithm before actually running it.



(a) Runtime comparison between computing equivalence classes and enumerating all topological orders.[10]



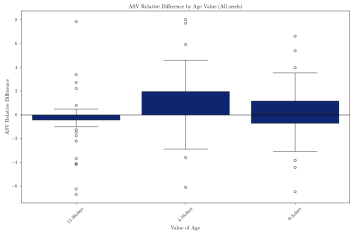
(b) Runtime as a function of the number of equivalence classes.

Figure 5a highlights the main advantage of our approach. While enumerating all topological orders quickly becomes infeasible (e.g., requiring days for large graphs), computing equivalence classes remains tractable.

Sampling-Based Approximation of Topological Orders We now evaluate the sampling-based approach.

# **Samples**	**Time (s)**
100	0.9270
1000	1.1170
10000	4.7170
20000	7.7170
30000	10.8387

(a) Sampling times for topological orders in *Child* using our developed algorithm.



(b) Relative error between approximate and exact ASV values in *Child*.

Table 6a shows that when using our algorithm the sampling remains efficient even for large numbers of samples. Moreover, the runtime is not linear in the number of samples, as multiple candidates are generated simultaneously during recursive calls due to our efficient implementation. We conclude that in the bounded-degree setting (observe that in all the examples considered here most nodes have small degree) our algorithm is more efficient than the alternative options such as Huber, 2006. Nevertheless, we recall that their algorithm is more general, as it applies beyond polytrees and runs in polynomial time.

[10] To estimate runtime, we measured the time required to generate the first 1000 topological orders and extrapolated using the total number of orders in each graph, since full execution was not computationally feasible.

Finally, we evaluate the approximation error of ASV using sampled orders. Figure 6b shows that using 1000 samples yields an average relative error of around 8%. Larger ASV values exhibit smaller relative error, while larger deviations correspond to very small true values. These results indicate that sampling provides a practical trade-off between accuracy and computational cost.

Remarks on Mean Prediction Computation In our experiments, the cost of evaluating the characteristic function is not the dominant factor. This happens because for decision trees under Bayesian Network distributions with a polytree structure mean prediction can be computed exactly in an efficient way. In general models this might not be the case, but nonetheless due to Proposition 3 if sampling is used instead of exact computation the overall error of the approximation should not be extremely bad.

5 Conclusions and open questions

In this work, we studied *Asymmetric Shapley Values* (ASV), a variant of Shapley values that incorporates causal information into feature attribution. We analyzed the computational complexity of ASV and showed tractability results for restricted settings in which computing SHAP was intractable. To make ASV more practical, we developed exact and approximate algorithms based on equivalence classes of topological orderings. Our experiments show that these techniques improve the performance for computing the score and yield accurate approximations.

Overall, our results suggest that ASV is a promising explainability framework when causal structure is relevant, since it can capture dependencies that standard SHAP does not reflect. Our experiments also indicate that ASV can be implemented effectively for several networks, although its scalability depends strongly on the causal graph. While equivalence classes can greatly reduce the number of topological orderings considered, they may still grow quickly in large or dense DAGs, where the tree or polytree assumptions used by the exact algorithms no longer apply and inference can become more expensive. In these cases, the method may need to rely on sampling-based approximations, whose accuracy depends on both topological-order sampling and conditional-expectation estimation. Thus, further optimization is needed for very large or highly connected graphs.

Several directions remain open for future work. On the theoretical side, it would be interesting to generalize the equivalence class algorithm from directed trees to polytrees, and to further study the complexity of computing the ASV in other cases. On the practical side, possible improvements include implementing alternative counting algorithms from the literature, optimizing the current implementation through caching or parallelization, and extending the framework to more general causal models beyond Bayesian networks.

Disclosure of Interests. The authors have no competing interests to declare that are relevant to the content of this article.

References

Arenas, M., Barceló, P., Bertossi, L., & Monet, M. (2023). On the complexity of Shap-score-based explanations: Tractability via knowledge compilation and non-approximability results. *Journal of Machine Learning Research*, *24*(63), 1–58.

Brightwell, G. R., & Winkler, P. (1991). Counting linear extensions. *Order*, *8*, 225–242. https://doi.org/10.1007/BF00383444

Bubley, R., & Dyer, M. (1999). Faster random generation of linear extensions. *Discrete mathematics*, *201*(1-3), 81–88.

Cooper, G. F. (1990). The computational complexity of probabilistic inference using bayesian belief networks. *Artificial Intelligence*, *42*(2–3), 393–405. https://doi.org/10.1016/0004-3702(90)90060-D

Frye, C., Rowat, C., & Feige, I. (2019). Asymmetric Shapley values: Incorporating causal knowledge into model-agnostic explainability. *arXiv preprint arXiv:1910.06358*.

Fryer, D., Strümke, I., & Nguyen, H. (2021). Shapley values for feature selection: The good, the bad, and the axioms. *Ieee Access*, *9*, 144352–144360.

Huang, X., & Marques-Silva, J. (2023). The inadequacy of Shapley values for explainability. *arXiv preprint arXiv:2302.08160*.

Huber, M. (2006). Fast perfect sampling from linear extensions. *Discrete Mathematics*, *306*(4), 420–428. https://doi.org/https://doi.org/10.1016/j.disc.2006.01.003

Kangas, K., Hankala, T., Niinimäki, T., & Koivisto, M. (2016). Counting linear extensions of sparse posets. *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence*, 603–609.

Lipton, Z. C. (2017). The mythos of model interpretability. https://arxiv.org/abs/1606.03490

Lundberg, S., & Lee, S.-I. (2017). A unified approach to interpreting model predictions. https://arxiv.org/abs/1705.07874

Marques-Silva, J. (2023). Logic-based explainability in machine learning. In *Reasoning web. causality, explanations and declarative knowledge: 18th international summer school 2022, berlin, germany, september 27–30, 2022, tutorial lectures* (pp. 24–104). Springer.

Mersha, M., Lam, K., Wood, J., Alshami, A. K., & Kalita, J. (2024). Explainable artificial intelligence: A survey of needs, techniques, applications, and future direction. *Neurocomputing*, *599*, 128111.

Miller, T. (2019). Explanation in Artificial intelligence: Insights from the social sciences. *Artificial Intelligence*, *267*, 1–38. https://doi.org/https://doi.org/10.1016/j.artint.2018.07.007

Pearl, J. (1986). Fusion, propagation, and structuring in belief networks. *Artificial Intelligence*, *29*(3), 241–288.

Shapley, L. S., et al. (1953). A value for n-person games. *Classics in game theory*.

Van den Broeck, G., Lykov, A., Schleich, M., & Suciu, D. (2022). On the tractability of Shap explanations. *Journal of Artificial Intelligence Research*, *74*, 851–886.