Finding Non-Redundant Simpson's Paradox in Multidimensional Data

Yi Yang
Duke University
Durham, NC, USA
owen.yang@duke.edu

Jian Pei
Duke University
Durham, NC, USA
j.pei@duke.edu

Jun Yang
Duke University
Durham, NC, USA
junyang@cs.duke.edu

Jichun Xie
Duke University
Durham, NC, USA
jichun.xie@duke.edu

ABSTRACT

Simpson's paradox has broad impact across many scientific domains. Existing detection methods overlook a key issue: many detected paradoxes may be *redundant*, arising from equivalent data subsets, identical subpopulation partitions, or correlated outcome variables, thereby obscure insights and increase computational cost. In this paper, we present a framework for finding *non-redundant* Simpson's paradoxes by formalizing three sources of redundancy—sibling child, separator, and statistic equivalence—and showing that pairwise redundancy forms an equivalence relation. We further propose a concise representation that groups redundant paradoxes and develop efficient algorithms combining depth-first population materialization with redundancy-aware discovery. Experiments on real and synthetic datasets show that redundancy is prevalent (over 40% in some cases), while our methods scale to millions of records, achieve up to 6.72× speedup over brute-force approaches and identify robust paradoxes, enabling efficient discovery, compact summarization, and clear interpretation in multidimensional data.

PVLDB Reference Format:
Yi Yang, Jian Pei, Jun Yang, and Jichun Xie. Finding Non-Redundant Simpson's Paradox in Multidimensional Data. PVLDB, 19(9): 2466 - 2479, 2026.
doi:10.14778/3819518.3819564

PVLDB Artifact Availability:
The source code, data, and/or other artifacts have been made available at https://github.com/Owen-Yang-18/non-redundant-simpson-paradox.

1 INTRODUCTION

Simpson's paradox [30, 40] refers to the reversal of an observed association between two variables when data is disaggregated into sub-populations, and is a classic and widely studied phenomenon in statistics, probability, and data science [2, 7, 14, 21, 22, 25, 29, 32, 38, 39, 42, 48, 49, 52, 59]. It has been recognized for more than a century and continues to play a central role in fields such as epidemiology, genomics, economics, machine learning, and causal inference [8, 12, 17, 22, 26, 28, 34, 44, 45, 54, 56], where decisions depend critically on understanding relationships in multidimensional data.

From a data management perspective, Simpson's paradox challenges OLAP query processing and interactive data exploration.

Proceedings of the VLDB Endowment, Vol. 19, No. 9 ISSN 2150-8097.
doi:10.14778/3819518.3819564

Table 1: Loan approval rates for Moving purposes and House purposes with data partitioned by Region.

Region	Purpose = Moving	Purpose = House
Leinster	50.00%	50.00%
Ulster	25.00%	50.00%
Northern-Irl	14.29%	22.67%
Connacht	0.00%	5.26%
Munster	0.00%	20.00%
Overall approval rate	**22.22%**	20.00%

When drill-down operations expose reversed trends, the association at the aggregate level can be misleading. This issue has been studied in database bias auditing systems [35–37] and is closely related to population lattice exploration and data cube analysis [23, 33, 47].

Example 1.1 (Simpson's paradox). Consider loan applications in the Irish Loan Approval dataset.[1] Moving loans appear to have a higher approval rate than House loans (22.22% vs. 20.00%; Table 1). However, when the data is partitioned by *Region*, the trend reverses in every subgroup. For example, in Ulster, House loans are approved twice as often as Moving loans (50% vs. 25%). More generally, for every region r, $P(\text{Approved} \mid \text{Region} = r, \text{Purpose} = \text{Moving}) \leq P(\text{Approved} \mid \text{Region} = r, \text{Purpose} = \text{House})$. This reversal after stratification is a classic instance of Simpson's paradox. □

Despite its importance, an overlooked issue is that instances of Simpson's paradox can be highly *redundant*. In high-dimensional data, different selections or partitions may involve the same records or induce equivalent sub-populations, causing multiple paradoxes to reflect the same phenomenon. Such redundancy also matters for *data quality assessment* and schema understanding: repeated paradoxes often reveal functional dependencies or strong attribute correlations (e.g., if *RegionCode* maps one-to-one to *Region*, then paradoxes partitioned by these attributes are redundant). Although these paradoxes appear syntactically different, they arise from the same population structure. Treating them as distinct inflates reported results and obscures the key insights analysts seek.

One may wonder whether redundant paradoxes occur merely in theory or in isolated cases. However, our empirical analysis using real-world datasets in different domains, reported in Section 5.2, shows that redundant Simpson's paradoxes account for 20.3–47.8% of all observed paradoxes.

One may also ask why Simpson's paradoxes must be discovered comprehensively rather than checked on demand. We argue that discovering all *non-redundant* paradoxes serves broader goals. In exploratory data analysis and OLAP navigation [23, 33, 47], analysts drill down without knowing which paths expose reversals;

[1] https://www.kaggle.com/datasets/ikpeleambrose/irish-loan-data

comprehensive discovery makes these paths explicit. For data quality assessment [35–37], it reveals structural properties invisible when paradoxes are examined in isolation, as redundant paradoxes often indicate functional dependencies or strong correlations (e.g., one-to-one mappings between *RegionCode* and *Region*). Our experiments (Section 5.2) show that this approach identifies attributes that frequently induce paradoxes, strongly correlated attributes, and attribute pairs with one-to-one mappings. Finally, in high-stakes domains such as healthcare, genomics, and policy-making [6, 9, 28, 45], comprehensive discovery helps avoid decisions based on misleading aggregate trends. For example, a SNP may appear to increase disease risk overall, but when conditioning on another stratifying SNP, the association can weaken, disappear, or even reverse [8, 54].

Identifying *non-redundant* Simpson's paradoxes poses several technical challenges. First, the search space of potential paradoxes grows exponentially with the number of attributes, making brute-force enumeration computation-expensive. Second, redundancies can arise in multiple ways, as analyzed in Section 3.1, and distinguishing among them requires careful formalization. Third, even after redundancies are recognized, a principled method is needed to group redundant paradoxes and produce concise, non-overlapping representations without information loss.

To address these challenges, our key idea is to exploit the structure of the population lattice and leverage convexity properties of coverage equivalence classes. By representing redundant paradoxes as equivalence classes, we can eliminate redundancy while preserving completeness. This approach enables the discovery of all instances of Simpson's paradox while organizing them into concise representations that capture their essential semantics.

We make four main contributions in this paper. First, we formally define three sources of redundancy – sibling child, separator, and statistic equivalence – and show that pairwise redundancy is an equivalence relation. Second, we propose a concise representation framework that groups redundant paradoxes into compact, systematic summaries. Third, we develop efficient algorithms that combine depth-first materialization of the base table and redundancy-aware paradox discovery. Last, we demonstrate through experiments on both real-world and synthetic datasets that redundant paradoxes are common in practice, our methods scale efficiently, and the identified paradoxes (and redundancies) are structurally robust.

2 PRELIMINARIES

2.1 Basic Notations

Consider a base table T with n *categorical attributes* $\{X_1, \ldots, X_n\}$ and m *label attributes* $\{Y_1, \ldots, Y_m\}$. Each categorical attribute X_i $(1 \le i \le n)$ has a finite domain $\mathrm{Dom}(X_i)$, and each label attribute Y_i $(1 \le i \le m)$ is binary; our results extend to multi-class labels, but we focus on the binary case for simplicity. Each record $t \in T$ is an $(n+m)$-tuple $(t.X_1, \ldots, t.X_n, t.Y_1, \ldots, t.Y_m)$.

A **population** s is an n-tuple $(s[1], \ldots, s[n])$ with $s[i] \in \mathrm{Dom}(X_i) \cup \{*\}$, where $*$ is a wildcard like `ALL` in data cube terminology [16, 19, 47, 58]. A population defines a subset of records via its **coverage** $\mathrm{cov}_T(s) = \{t \in T \mid t.X_i = s[i] \ \vee \ s[i] = *, \ 1 \le i \le n\}$, and we omit the subscript T when the context is unambiguous.

Given populations s and s', s is a **parent** of s' (and s' a **child** of s), denoted $s \succ\!\cdot\ s'$, if there exists an attribute X_j such that $s[j] = *$ and $s'[j] \neq *$, and $s[i] = s'[i]$ for all $i \neq j$. Then $\mathrm{cov}(s) \supseteq \mathrm{cov}(s')$. We call X_j the **differential attribute** and $s'[j]$ the **differential value**, and write $s' = s\langle X_j \rightarrow s'[j]\rangle$. A population may have multiple parents and children.

A population s is an **ancestor** of s' (and s' a **descendant** of s), denoted $s \succ s'$, if for all i, either $s[i] = *$ or $s[i] = s'[i]$, and for at least one i, $s[i] \neq s'[i]$. In this case $\mathrm{cov}(s) \supseteq \mathrm{cov}(s')$. We write $s \succeq s'$ if $s \succ s'$ or $s = s'$. Two populations s_1 and s_2 are **siblings** if they are children of a common parent under the same differential attribute with different values; then $\mathrm{cov}(s_1) \cap \mathrm{cov}(s_2) = \emptyset$.

For a non-empty population s, the **frequency statistic** of label Y_i is the conditional probability $P(Y_i \mid s) = \frac{|\mathrm{cov}_T(s) \cap \{t \in T \mid t.Y_i = 1\}|}{|\mathrm{cov}_T(s)|}$.

All populations form a lattice under $\succ\!\cdot$. Let $\mathcal{P}$ be the set of all populations and $\mathcal{E} \subseteq \mathcal{P}$. Two populations in $\mathcal{E}$ are *directly connected* if one is a parent of the other, and are *connected* if linked by a sequence of such relations. A subset $\mathcal{E}$ is **convex** if all its populations are connected and, whenever $s \succ s'$ are in $\mathcal{E}$, all populations that are descendants of s and ancestors of s' also belong to $\mathcal{E}$. Figure 1 shows the population lattice of a table.

2.2 Simpson's Paradox

Simpson's Paradox [30, 40] describes the counterintuitive phenomenon in which the type of relationship (e.g., positive, negative, or independent) between two variables reverses when the population is partitioned into sub-populations. In this subsection, we formalize Simpson's Paradox in multidimensional data, provide an illustrative example, and briefly review its well-known variants.

Definition 2.1. Consider a population s and two sibling child populations $s_1 = s\langle X_j \rightarrow u_1\rangle$ and $s_2 = s\langle X_j \rightarrow u_2\rangle$ with the differential attribute X_j $(1 \le j \le n)$, where $u_1, u_2 \in \mathrm{Dom}(X_j)$ $(u_1 \neq u_2)$ are the respective *differential values*. Let $X \in \{X_1, \ldots, X_n\} \setminus \{X_j\}$ be a **separator attribute** and $Y \in \{Y_1, \ldots, Y_m\}$ be a label attribute. The tuple (s_1, s_2, X, Y) is called an **association configuration (AC)**. An AC is a **Simpson's Paradox** if the following holds:

(1) $P(Y|s_1) \ge P(Y|s_2)$;
(2) For every separator attribute value $v \in \mathrm{Dom}(X)$ with $\mathrm{cov}(s_1\langle X \rightarrow v\rangle) \neq \emptyset$ and $\mathrm{cov}(s_2\langle X \rightarrow v\rangle) \neq \emptyset$:
$$P(Y|s_1\langle X \rightarrow v\rangle) \le P(Y|s_2\langle X \rightarrow v\rangle);$$
(3) Either the inequality in (1) is strict or all inequalities in (2) are strict. □

The inequality directions in (1) and (2) may be reversed simultaneously.

Example 2.2. In the table in Figure 1, $((*, b_1, *), (*, b_2, *), A, Y_1)$ is an association configuration. $P(Y_1|B = b_1) = 0.5 < P(Y_1|B = b_2) = \frac{2}{3}$. However, $P(Y_1|A = a_i, B = b_2) = P(Y_1|A = a_i, B = b_1)$ for $i = 1, 2$. Thus, $((*, b_1, *), (*, b_2, *), A, Y_1)$ is a Simpson's paradox. □

When Y is a multi-class label, that is $\mathrm{Dom}(Y) = \{y_1, \ldots, y_k\}$ with $k > 2$, Definition 2.1 extends by replacing $P(Y \mid s)$ with $P(Y{=}c \mid s)$ for each class $c \in \mathrm{Dom}(Y)$, yielding the AC $(s_1, s_2, X, Y{=}c)$. A k-class label can therefore generate up to k distinct ACs.

The partitioning in Definition 2.1 also generalizes to a set of multiple separator attributes.

	A	B	C	Y_1
t_1	a_1	b_1	c_1	0
t_2	a_1	b_1	c_1	0
t_3	a_1	b_2	c_1	0
t_4	a_2	b_1	c_2	1
t_5	a_2	b_1	c_2	1
t_6	a_2	b_2	c_2	1
t_7	a_2	b_2	c_2	1

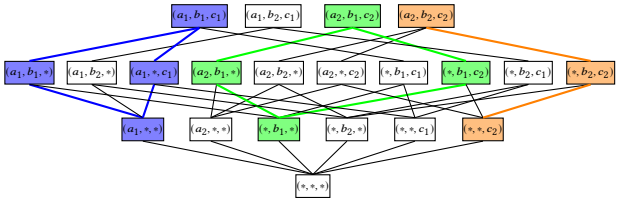


Figure 1: A data table $T(A, B, C, Y_1)$ and the Hasse diagram of the population lattice with respect to the parent-child relation $\succ$. A parent is placed lower than its child. The blue and green subsets are convex, while the orange subset is non-convex.

Definition 2.3 (Multi-attribute Separator). Let X_j be the differential attribute such that $s_1 = s\langle X_j \shortrightarrow u_1\rangle$ and $s_2 = s\langle X_j \shortrightarrow u_2\rangle$, and let $\mathbf{X} \subseteq \{X_1, \ldots, X_n\} \setminus \{X_j\}$ be a set of separator attributes, where $\mathrm{Dom}(\mathbf{X}) = \prod_{X_i \in \mathbf{X}} \mathrm{Dom}(X_i)$ is the Cartesian product of their domains. The AC $(s_1, s_2, \mathbf{X}, Y)$ is a **Simpson's Paradox** if:

(1) $P(Y \mid s_1) \geq P(Y \mid s_2)$;
(2) For every $\mathbf{v} \in \mathrm{Dom}(\mathbf{X})$ with $\mathrm{cov}(s_1\langle \mathbf{X} \shortrightarrow \mathbf{v}\rangle) \neq \emptyset$ and $\mathrm{cov}(s_2\langle \mathbf{X} \shortrightarrow \mathbf{v}\rangle) \neq \emptyset$:

$$P(Y \mid s_1\langle \mathbf{X} \shortrightarrow \mathbf{v}\rangle) \leq P(Y \mid s_2\langle \mathbf{X} \shortrightarrow \mathbf{v}\rangle);$$

(3) Either the inequality in (1) is strict or all inequalities in (2) are strict.

When $|\mathbf{X}| = 1$, this reduces to Definition 2.1. □

This paper assumes binary labels and a single separator attribute, though our results extend directly to the general cases above.

Several variants of Simpson's paradox have been studied, including Association Reversal (AR) [38] as formalized in Definition 2.1, Yule's Association Paradox (YAP) [57], Amalgamation Paradox (AMP) [15], and Averaged Association Reversal (AAR) [2, 49]. YAP, AMP, and AAR are all special cases of AR. We refer readers to Sprenger and Weinberger [43] for an in-depth review, and our framework naturally extends to these variants.

3 REDUNDANCY AMONG INSTANCES OF SIMPSON'S PARADOX

In practice, multiple populations in a table may have identical coverage, leading to different association configurations that capture essentially the same paradoxical behavior. For example, in Table 2, $\mathrm{cov}(a_1, *, *, *) = \mathrm{cov}(a_1, *, c_1, *) = \{t_1, t_2, t_3\}$. When a table has many attributes but relatively sparse records, such overlaps are common [5, 10, 18, 23, 33]. This incidental identicality can generate multiple Simpson's paradoxes that are redundant. In this section, we formalize **redundancy** through three types of equivalences that give rise to it, and then unifying them into a single definition.

3.1 Three Types of Redundancies

Redundancy may arise from three distinct sources.

3.1.1 Sibling Child Equivalence. When sibling populations have identical coverage, their corresponding paradoxes are redundant.

Lemma 3.1 (Sibling child equivalence). *Consider two association configurations, $p = (s_1, s_2, X, Y)$ and $p' = (s_1', s_2', X, Y)$, where $\mathrm{cov}(s_1) = \mathrm{cov}(s_1')$ and $\mathrm{cov}(s_2) = \mathrm{cov}(s_2')$. If p is a Simpson's paradox, then p' is also a Simpson's paradox.* □

Table 2: Data table $T(A, B, C, D, Y_1, Y_2)$ containing 7 records.

	A	B	C	D	Y_1	Y_2
t_1	a_1	b_1	c_1	d_1	0	0
t_2	a_1	b_1	c_1	d_1	0	0
t_3	a_1	b_2	c_1	d_2	0	0
t_4	a_2	b_1	c_2	d_1	1	1
t_5	a_2	b_1	c_2	d_1	1	1
t_6	a_2	b_2	c_2	d_2	1	1
t_7	a_2	b_2	c_2	d_2	1	1

Example 3.2 (Sibling child equivalence). Consider Table 2. It can be verified that $((*, b_1, *, *), (*, b_2, *, *), A, Y_1)$ is a Simpson's paradox. It can also be verified that $((*, *, *, d_1), (*, *, *, d_2), A, Y_1)$ is a Simpson's paradox due to sibling child equivalence—$\mathrm{cov}(*, b_i, *, *) = \mathrm{cov}(*, *, *, d_i)$ for $i = 1, 2$. □

3.1.2 Separator Equivalence. When two separator attributes produce identical partitioning of the sibling populations, the resulting Simpson's paradoxes are redundant.

Lemma 3.3 (Separator equivalence). *Consider two ACs $p = (s_1, s_2, X, Y)$ and $p' = (s_1, s_2, X', Y)$, where $X \neq X'$ and there exists a one-to-one mapping $f : \mathrm{Dom}(X) \mapsto \mathrm{Dom}(X')$ such that for every $v \in \mathrm{Dom}(X)$ and $s \in \{s_1, s_2\}$, $\mathrm{cov}(s\langle X \shortrightarrow v\rangle) = \mathrm{cov}(s\langle X' \shortrightarrow f(v)\rangle)$. If p is a Simpson's paradox, then p' is also a Simpson's paradox.* □

Example 3.4 (Separator equivalence). In Table 2, $((*, b_1, *, *), (*, b_2, *, *), A, Y_1)$ is a Simpson's paradox. Consider the mapping $f(a_1) = c_1$, $f(a_2) = c_2$. Since $\mathrm{cov}(a_1, b_i, *, *) = \mathrm{cov}(*, b_i, c_1, *)$ and $\mathrm{cov}(a_2, b_i, *, *) = \mathrm{cov}(*, b_i, c_2, *)$ for $i = 1, 2$, partitioning by A and by C produce identical sub-populations. Therefore $((*, b_1, *, *), (*, b_2, *, *), C, Y_1)$ is also a Simpson's paradox by separator equivalence. □

3.1.3 Statistic Equivalence. When label attributes are dependent, their Simpson's paradoxes may be redundant. Intuitively, if a Simpson's paradox under Y where the trends $P(Y \mid s_1) \geq P(Y \mid s_2)$ and $P(Y \mid s_1\langle X \shortrightarrow v\rangle) \leq P(Y \mid s_2\langle X \shortrightarrow v\rangle)$ also hold for Y', then it is also a Simpson's paradox under Y'. We identify three sufficient conditions for this equivalence.

Lemma 3.5 (Statistic equivalence). *Consider two ACs $p = (s_1, s_2, X, Y)$ and $p' = (s_1, s_2, X, Y')$ such that $Y \neq Y'$. If p is a Simpson's paradox and if any of the following (sufficient, and progressively less restrictive) conditions hold, then p' is also a Simpson's paradox:*

(1) *For every $t \in \mathrm{cov}(s_1) \cup \mathrm{cov}(s_2)$, $t.Y = t.Y'$;*
(2) *For every $s \in \{s_1, s_2\}$, $P(Y|s) = P(Y'|s)$ and for every $v \in \mathrm{Dom}(X)$, $P(Y|s\langle X \shortrightarrow v\rangle) = P(Y'|s\langle X \shortrightarrow v\rangle)$;*

Table 3: Data table $T(A, B, C, D, Y_1, Y_2, Y_3, Y_4)$ with 11 records.

	A	B	C	D	Y_1	Y_2	Y_3	Y_4
t_1	a_1	b_1	c_1	d_1	0	0	0	0
t_2	a_1	b_1	c_1	d_1	0	0	0	0
t_3	a_1	b_1	c_1	d_2	0	0	0	0
t_4	a_1	b_2	c_1	d_1	0	0	0	0
t_5	a_1	b_2	c_1	d_2	0	0	0	0
t_6	a_2	b_1	c_2	d_1	0	0	1	1
t_7	a_2	b_1	c_2	d_1	1	1	0	1
t_8	a_2	b_1	c_2	d_2	1	1	1	1
t_9	a_2	b_2	c_2	d_1	0	0	1	1
t_{10}	a_2	b_2	c_2	d_2	1	1	0	1
t_{11}	a_2	b_2	c_2	d_2	1	1	1	1

(3) $\text{sign}(P(Y|s_1) - P(Y|s_2)) = \text{sign}(P(Y'|s_1) - P(Y'|s_2))$, *and for every* $v \in \text{Dom}(X)$, $\text{sign}(P(Y|s_1\langle X \shortrightarrow v\rangle) - P(Y|s_2\langle X \shortrightarrow v\rangle)) = \text{sign}(P(Y'|s_1\langle X \shortrightarrow v\rangle) - P(Y'|s_2\langle X \shortrightarrow v\rangle))$. □

Example 3.6 (Statistic equivalence). Consider Table 3, an extension of Table 2, and $p = ((*, b_1, *, *), (*, b_2, *, *), A, Y_1)$. $P(Y_1|(*, b_1, *, *)) = 0.33 < P(Y_1|(*, b_2, *, *)) = 0.40$. Partitioning by A yields $P(Y_1|(a_1, b_1, *, *)) = P(Y_1|(a_1, b_2, *, *)) = 0$ and $P(Y_1|(a_2, b_1, *, *)) = P(Y_1|(a_2, b_2, *, *)) = 0.67$, confirming that p is a Simpson's paradox.

It follows that $((*, b_1, *, *), (*, b_2, *, *), A, Y_2)$ is statistic equivalent to p (Case 1), since $t.Y_1 = t.Y_2$ for every record t.

Moreover, $((*, b_1, *, *), (*, b_2, *, *), A, Y_3)$ is statistic equivalent to p (Case 2), as the frequency statistics of Y_3 match those of Y_1 across all relevant populations, even though $Y_1 \neq Y_3$ for some records.

Similarly, $((*, b_1, *, *), (*, b_2, *, *), A, Y_4)$ is statistic equivalent to p (Case 3), since the signs of the frequency-statistic differences coincide for Y_1 and Y_4 at the aggregate and sub-population.

Finally, note that Y_4 could be constructed so that the sign of the frequency-statistic difference at the aggregate matches that of Y_1, while for one sub-population the difference is negative for Y_1 but zero for Y_4; despite mismatch, Simpson's Paradox is still preserved with Y_4, showing that Case 3 is sufficient but not necessary. □

3.2 Equivalence Classes of Simpson's Paradoxes

Redundant Simpson's paradoxes may arise from more than one of the equivalence types, sometimes combining sibling child, separator, and statistic equivalence within the same instances.

Example 3.7 (Redundancy). $((*, b_1, *, *), (*, b_2, *, *), A, Y_1)$ is a Simpson's paradox in Table 2. It can be verified that $((*, *, *, d_1), (*, *, *, d_2), C, Y_2)$ is also a Simpson's paradox, and redundant due to sibling child, separator, and statistic equivalence simultaneously. □

Motivated by the above observation, we integrate the three equivalences into a unified notion.

Definition 3.8 (Redundancy). Two distinct paradoxes are:

- **sibling child equivalent** if they satisfy Lemma 3.1;
- **separator equivalent** if they satisfy Lemma 3.3;
- **statistic equivalent** if they satisfy at least one of the conditions in Lemma 3.5.

They are **redundant** if any of the above hold. □

Theorem 3.9 (Equivalence). *Pairwise redundancy of Simpson's paradoxes is an equivalence relation.* □

Because pairwise redundancy is an equivalence relation, the set of all Simpson's paradoxes can be partitioned into equivalence classes. Each class corresponds to a group of mutually redundant paradoxes. In some cases, an equivalence class may contain only a single instance when no redundancy is observed.

Example 3.10 (Equivalence class). Consider Table 2, $p_1 = ((*, b_1, *, *), (*, b_2, *, *), A, Y_1)$ is a paradox. We also obtain the following paradoxes redundant to p_1:

- $p_2 = ((*, *, *, d_1), (*, *, *, d_2), A, Y_1)$ (sibling child equiv.);
- $p_3 = ((*, b_1, *, *), (*, b_2, *, *), C, Y_1)$ (separator equiv.);
- $p_4 = ((*, b_1, *, *), (*, b_2, *, *), A, Y_2)$ (statistic equiv.);
- $p_5 = ((*, *, *, d_1), (*, *, *, d_2), C, Y_1)$ (sibling and separator);
- $p_6 = ((*, *, *, d_1), (*, *, *, d_2), A, Y_2)$ (sibling and statistic);
- $p_7 = ((*, b_1, *, *), (*, b_2, *, *), C, Y_2)$ (separator and stat.);
- $p_8 = ((*, *, *, d_1), (*, *, *, d_2), C, Y_2)$ (all equivalences).

For instance, p_5 is redundant to p_1 by combining sibling child and separator equivalence: replacing $(*, b_i, *, *)$ with $(*, *, *, d_i)$ (since $\text{cov}(*, b_i, *, *) = \text{cov}(*, *, *, d_i)$ for $i = 1, 2$) and replacing separator A with C (since $\text{cov}(a_j, b_1, *, *) = \text{cov}(*, b_1, c_j, *)$ and $\text{cov}(a_j, b_2, *, *) = \text{cov}(*, b_2, c_j, *)$ for $j = 1, 2$).

The set $\{p_1, p_2, \ldots, p_8\}$ forms an equivalence class. □

3.3 Representing Equivalence Classes Concisely

In real datasets with many attributes, the number of redundant paradoxes can be large. We therefore propose a concise representation for an equivalence class of redundant Simpson's paradoxes (or **redundant paradox group** for short). To begin, we make the following observation.

Lemma 3.11 (Product Space). *Each redundant paradox group can be characterized by the product of* $\mathcal{E}_1 \times \mathcal{E}_2 \times \mathbf{X} \times \mathbf{Y}$, *where* $\mathbf{X}$ *is a set of separator attributes,* $\mathbf{Y}$ *is a set of label attributes, and* $\mathcal{E}_1, \mathcal{E}_2$ *are sets of sibling populations, each containing populations with identical coverage. Any choice of* $(s_1, s_2, X, Y) \in \mathcal{E}_1 \times \mathcal{E}_2 \times \mathbf{X} \times \mathbf{Y}$ *where* s_1, s_2 *are siblings is a Simpson's paradox in the redundant paradox group.* □

Example 3.12. The redundant paradox group from Example 3.10 is characterized by the product space $\mathcal{E}_1 \times \mathcal{E}_2 \times \mathbf{X} \times \mathbf{Y}$, where $\mathcal{E}_1 = \{(*, b_1, *, *), (*, *, *, d_1), (*, b_1, *, d_1)\}$, $\mathcal{E}_2 = \{(*, b_2, *, *), (*, *, *, d_2), (*, b_2, *, d_2)\}$, $\mathbf{X} = \{A, C\}$, and $\mathbf{Y} = \{Y_1, Y_2\}$. This product space encompasses multiple paradoxes; for instance, $((*, b_1, *, *), (*, b_2, *, *), A, Y_1)$ and $((*, *, *, d_1), (*, *, *, d_2), C, Y_2)$ are included. However, $((*, b_1, *, d_1), (*, b_2, *, d_2), A, Y_1)$ is not, since these two populations are not valid siblings. □

Next, we show that $\mathcal{E}_1$ and $\mathcal{E}_2$ can be represented more concisely. We partition the set of all populations $\mathcal{P}$ based on coverage, denoted $\mathcal{P}/\equiv_{\text{cov}}$, where each **coverage group** $\mathcal{E} \in \mathcal{P}/\equiv_{\text{cov}}$ contains populations with identical coverage. We show that each such coverage group exhibits a structural property in the population lattice: they form convex subsets. This means that if two populations belong to the same coverage group, then all populations that lie between them in the lattice hierarchy must also belong to that group. Given convexity of any $\mathcal{E} \in \mathcal{P}/\equiv_{\text{cov}}$, we call a population $s \in \mathcal{E}$ an **upper bound** of $\mathcal{E}$ if there is no $s' \in \mathcal{E}$ with $s \prec s'$ (i.e., least descendant); similarly, we call s a **lower bound** of $\mathcal{E}$ if there is no $s' \in \mathcal{E}$ with $s' \prec s$ (i.e., greatest ancestor). We denote the set

of upper bounds of $\mathcal{E}$ by $\mathsf{up}(\mathcal{E})$ and the set of lower bounds of $\mathcal{E}$ by $\mathsf{low}(\mathcal{E})$. Convexity ensures that we can represent coverage groups $\mathcal{E}_1$ and $\mathcal{E}_2$ using only these bounds, thereby avoiding enumerating all members, which can be prohibitively large in high-dimensional datasets. Furthermore, we show that each coverage group has a unique upper bound (though it can have multiple lower bounds).

Property 1 (Convexity of coverage groups). *Let $\mathcal{P}$ be the set of all populations. For each coverage group $\mathcal{E} \in \mathcal{P}/\equiv_{\text{cov}}$, $\mathcal{E}$ is a convex subset of coverage-identical populations. Furthermore, $|\,\mathsf{up}(\mathcal{E})| = 1$ and the least descendant is the unique upper bound.* □

Property 2 (Reconstruction from bounds). *Let $\mathcal{E} \subseteq \mathcal{P}$ be a convex subset of populations. Then $s \in \mathcal{E}$ if and only if there exist $s_l \in \mathsf{low}(\mathcal{E})$ and $\{s_u\} = \mathsf{up}(\mathcal{E})$ such that $s_l \preceq s \preceq s_u$.* □

Using these properties and Lemma 3.11, we can concisely represent each redundant paradox group. We remark that while Property 1 establishes that upper bounds uniquely identify (convex) coverage groups, reconstruction requires lower bounds. Given only the upper bound s_u, enumerating all $s \in \mathcal{E}$ requires verifying coverage equality for each ancestor $s \prec s_u$. With lower bounds $\mathsf{low}(\mathcal{E})$, reconstruction is straightforward and efficient: following Property 2, enumerate all s satisfying $s_l \preceq s \preceq s_u$ for every $s_l \in \mathsf{low}(\mathcal{E})$. Reconstruction is essential because our concise representation must generate all Simpson's paradoxes in the group by enumerating every valid sibling pairs in $\mathcal{E}_1$ and $\mathcal{E}_2$ (as shown in Examples 3.10, 3.12).

Definition 3.13 (Concise representation). A redundant paradox group characterized by $\mathcal{E}_1 \times \mathcal{E}_2 \times \mathbf{X} \times \mathbf{Y}$ can be represented as:

$$(\mathsf{up}(\mathcal{E}_1), \mathsf{low}(\mathcal{E}_1), \mathsf{up}(\mathcal{E}_2), \mathsf{low}(\mathcal{E}_2), \mathbf{X}, \mathbf{Y}),$$

where $\mathsf{up}(\mathcal{E}_1)$ and $\mathsf{up}(\mathcal{E}_2)$ are the (unique) upper bounds of $\mathcal{E}_1$ and $\mathcal{E}_2$, and $\mathsf{low}(\mathcal{E}_1)$ and $\mathsf{low}(\mathcal{E}_2)$ are their lower bounds. □

By Property 2, this representation is precise: all populations in $\mathcal{E}_1$ and $\mathcal{E}_2$ (and thus all redundant Simpson's paradoxes in the group) can be reconstructed from the bounds.

Example 3.14 (Concise representation). From Example 3.10, the redundant paradox group can be represented as:

- $\mathsf{up}(\mathcal{E}_1) = \{(*, b_1, *, d_1)\}$, $\mathsf{up}(\mathcal{E}_2) = \{(*, b_2, *, d_2)\}$,
- $\mathsf{low}(\mathcal{E}_1) = \{(*, b_1, *, *), (*, *, *, d_1)\}$, $\mathbf{X} = \{A, C\}$,
- $\mathsf{low}(\mathcal{E}_2) = \{(*, b_2, *, *), (*, *, *, d_2)\}$, $\mathbf{Y} = \{Y_1, Y_2\}$.

All eight redundant Simpson's paradoxes from Example 3.10 are captured. For instance, $((*, b_1, *, *), (*, b_2, *, *), A, Y_1)$ is valid because $(*, b_1, *, *) \in \mathcal{E}_1$ and $(*, b_2, *, *) \in \mathcal{E}_2$ are siblings. By contrast, $((*, b_1, *, d_1), (*, b_2, *, d_2), A, Y_1)$ is not valid, since these two populations are not siblings. □

4 FINDING NON-REDUNDANT SIMPSON'S PARADOXES

Building on the concise representation developed in Section 3, our framework proceeds in two phases: (1) *materialization*, which enumerates all non-empty populations and organizes them into convex coverage groups; and (2) *paradox discovery*, which detects all instances of Simpson's paradox and simultaneously constructs concise representations of redundant paradox groups.

Algorithm 1 Brute-force Materialization

Input: Data table $T = (\{X_i\}_{i=1}^n, \{Y_j\}_{j=1}^m)$
Output: Materialized populations

```
for each record t ∈ T do
    for each ancestor s of t do
        Update cov(s) and P(Y|s) for each Y ∈ {Y_1, ..., Y_m}.
```

Algorithm 2 DFS-based Materialization

Input: Data table $T = (\{X_i\}_{i=1}^n, \{Y_j\}_{j=1}^m)$, coverage threshold θ
Output: Coverage-based partitioning $\mathcal{P}/\equiv_{\text{cov}}$ of populations, where each group $\mathcal{E}$ is represented by $(\mathsf{up}(\mathcal{E}), \mathsf{low}(\mathcal{E}))$; frequency STATS for each $\mathcal{E} \in \mathcal{P}/\equiv_{\text{cov}}$, indexed by $\mathsf{up}(\mathcal{E})$.

```
Initialize the set G of candidate coverage groups and STATS;
DFS((*, *, ..., *), T, 0);                          ▷ updates G and STATS
for each unique upper bound u such that (u, _) ∈ G do
    L ← {s | (u, s) ∈ G ∧ ¬∃s' : ((u, s') ∈ G ∧ s' ≺ s)};
    Add (u, L) as a coverage group in P/≡_cov;
return P/≡_cov, STATS;
function DFS(s, T', k):                             ▷ updates G and STATS
    d ← s;
    for each attribute X_i (1 ≤ i ≤ n) with s[i] = * do
        if ∃ v ∈ Dom(X_i) : cov(s) = cov(s⟨X_i ↦ v⟩) then
            d[i] ← v;
    Add (d, s) to G; STATS(d) ← {P(Y_j|d)}_{j=1}^m;
    for each attribute X_h (k < h ≤ n) with d[h] = * do
        for each v ∈ Dom(X_h) do
            T'_v ← {t | t ∈ T' ∧ t.X_h = v};
            if |T'_v| ≥ θ · |T| then
                DFS(d⟨X_h ↦ v⟩, T'_v, h);
```

4.1 Materialization

The first step in our algorithm materializes all non-empty populations and computes their coverage and frequency statistics. A brute-force approach (Algorithm 1) iterates over each record $t \in T$ and updates all its 2^n ancestor populations s with $s[i] \in \{t.X_i, *\}$. This approach is inefficient for two reasons: it repeatedly processes identical populations across records, resulting in $|T| \times 2^n$ updates, and it does not organize populations into convex coverage groups, thereby missing opportunities to avoid redundant materialization.

To overcome these issues, we adopt a depth-first search (DFS) strategy adapted from [5, 18, 23]. The DFS directly constructs coverage groups—convex sets of populations with identical coverage—while computing their frequency statistics. The procedure is summarized in Algorithm 2.

4.1.1 DFS-based Population Materialization. The algorithm constructs the population lattice (see Figure 1) in a bottom-up manner. It starts from the root population $s_{\text{root}} = (*, *, \ldots, *)$, which covers the entire table T, and progressively materializes populations with smaller coverage, moving upward in the lattice.

At each DFS step, the goal is to identify a candidate convex coverage group. Starting from a lower-bound population s, we compute its upper bound s'. We initialize $s' = s$ and scan the records in $\text{cov}(s)$: for each attribute X_i with $s[i] = *$, if all records share a value $v \in \text{Dom}(X_i)$, we set $s'[i] = v$; otherwise, $s'[i] = *$. This construction guarantees $\text{cov}(s') = \text{cov}(s)$.

Example 4.1. In Table 2, let $s = (a_1, *, *, *)$ with $\text{cov}(s) = \{t_1, t_2, t_3\}$. All records share c_1 on attribute C, but not on B or D, yielding the upper bound $s' = (a_1, *, c_1, *)$. □

By Property 2, all populations s'' with $s \geq s'' \geq s'$ share the same coverage and statistics, and thus need not be explicitly materialized. This substantially reduces the computation. Once s' is identified, the pair (s, s') becomes a candidate coverage group. The DFS then recurses on each child $\hat{s}$ of s', using $\hat{s}$ as the next lower bound.

Example 4.2. From Example 4.1, $(s = (a_1, *, *, *), s' = (a_1, *, c_1, *))$ forms a convex group. DFS explores children of s', such as $(a_1, b_1, c_1, *)$, whose coverage is non-empty and strictly smaller, and continues recursively. □

The recursion stops when a population covers fewer than $\theta \cdot |T|$ records (Section 4.1.3) or when the top of the lattice is reached.

4.1.2 Coverage Group Merging. DFS may discover the same coverage group through different lower bounds. We merge candidate groups that share the same upper bound. Each merged group has a unique upper bound and possibly multiple lower bounds.

Example 4.3. In Table 2, $(a_1, *, *, *)$, $(*, *, c_1, *)$, and $(a_1, *, c_1, *)$ all cover $\{t_1, t_2\}$. DFS may reach $(a_1, *, c_1, *)$ via $(a_1, *, *, *)$ or $(*, *, c_1, *)$. After merging, the upper bound is $(a_1, *, c_1, *)$ and valid lower bounds are $(a_1, *, *, *)$ and $(*, *, c_1, *)$. □

4.1.3 Population Pruning. Populations with very small coverage are of limited analytical usefulness. We introduce a pruning threshold $0 \leq \theta \leq 1$: any population with coverage size below $\theta \cdot |T|$ is not materialized. The DFS prunes such populations and their descendants, further reducing run time.

Example 4.4. In Table 2, let $\theta = 0.4$. Population $(*, *, c_1, *)$ covers $\{t_1, t_2, t_3\}$ and is retained since $3/7 > 0.4$. If $\theta = 0.6$, this population and its descendants, such as (a_1, b_1, c_1, d_1), are pruned. □

Theorem 4.5 (Completeness). *Algorithm 2 materializes all non-empty populations that satisfy the coverage threshold. After group merging, it yields maximal convex coverage groups of coverage-identical populations; no population outside a group shares the same coverage as any population within it.* □

4.2 Finding Redundant Paradox Groups

The discovery of Simpson's paradoxes is a two-step process: (1) systematically enumerating all possible ACs, and (2) evaluating each AC against Definition 2.1. Algorithm 3 illustrates a brute-force method: for each non-empty population s, we enumerate all sibling child pairs (s_1, s_2), and then combine each pair with every valid separator attribute and label attribute to form candidate ACs. Each AC is then checked against Definition 2.1.

This exhaustive search has two major drawbacks. First, it does not organize discovered paradoxes into redundant paradox groups. Second, it wastes computation by (i) repeatedly iterating over populations with identical coverage, and (ii) evaluating ACs that are redundant to already discovered paradoxes.

We therefore design optimizations to avoid repeated computation and concisely represent redundant paradox groups.

Algorithm 3 Brute-force finding of Simpson's paradoxes

Input: Materialized populations S from $T = (\{X_i\}_{i=1}^{n}, \{Y_j\}_{j=1}^{m})$
Output: All instances of Simpson's paradox

```
for each population s ∈ S do
    for each X_i with s[i] = * do
        for each pair v_1, v_2 ∈ Dom(X_i) where v_1 ≠ v_2 do
            s_1 ← s⟨X_i → v_1⟩;   s_2 ← s⟨X_i → v_2⟩;
            for each i' ≠ i with s[i'] = * do
                for each label attribute Y do
                    Evaluate (s_1, s_2, X_i', Y) using Definition 2.1;
```

4.2.1 Iteration over Coverage Groups. Instead of iterating over every non-empty population in Algorithm 3, we exploit the convex coverage groups discovered from Algorithm 2. From each coverage group, it suffices to consider only one representative population – specifically, its unique upper bound (Property 1) – since all populations in the coverage group have identical coverage. This helps avoid repeated computation over such populations.

4.2.2 Constructing Redundant Paradox Groups. Even after iterating over coverage groups, many ACs remain redundant due to three types of equivalences. First, *sibling child equivalence*: coverage groups contain multiple populations beyond their upper bounds, each forming valid sibling pairs that generate equivalent paradoxes. For example, valid sibling pairs in coverage groups $\mathcal{E}_1$ and $\mathcal{E}_2$ can create sibling child equivalent ACs. Second, *separator and statistic equivalence*: different separator attributes produce identical partitions, and different labels may be perfectly correlated. For example, for siblings $s_1 = (*, b_1, *, *)$ and $s_2 = (*, b_2, *, *)$ from Table 2, attributes A and C partition the data identically and Y_1, Y_2 are perfectly correlated, so (s_1, s_2, A, Y_1) and (s_1, s_2, C, Y_2) are redundant.

We propose two strategies that exploit the redundancy conditions from Section 3.2 to construct redundant paradox groups while maintaining concise representations (Section 3.3).

Construction by sibling child equivalence. Once a Simpson's paradox $p = (s_1, s_2, X, Y)$ is identified, all paradoxes sibling child equivalent to p can be inferred without evaluation. Thanks to Section 3.3, we can also encode the entire set of such paradoxes concisely. This strategy is formalized below and implemented by Algorithm 4.

Proposition 4.6. *Let $p = (s_1, s_2, X, Y)$ be a Simpson's paradox, where s_1 and s_2 belong to coverage groups $\mathcal{E}_1$ and $\mathcal{E}_2$ in $\mathcal{P}/\equiv_{\text{cov}}$, respectively. Then for any $(s_1', s_2') \in \mathcal{E}_1 \times \mathcal{E}_2$ such that s_1' and s_2' are siblings, the AC $p' = (s_1', s_2', X, Y)$ is also a Simpson's paradox and redundant with respect to p.* □

Example 4.7 (Construction by sibling child equivalence). In Table 2, consider the Simpson's paradox $p = ((*, b_1, *, *), (*, b_2, *, *), A, Y_1)$ from Example 3.2. With materialization by Algorithm 2, populations $(*, b_1, *, *)$ and $(*, b_2, *, *)$ belong to coverage groups $\{(*, b_1, *, *), (*, *, *, d_1), (*, b_1, *, d_1)\}$ and $\{(*, b_2, *, *), (*, *, *, d_2), (*, b_2, *, d_2)\}$, respectively. Valid sibling pairs across these groups yield additional paradoxes, such as $((*, *, *, d_1), (*, *, *, d_2), A, Y_1)$, which are redundant to p. These can be directly included in the same redundant paradox group without further evaluation. Furthermore, instead of enumerating these paradoxes, the group can

Algorithm 4 Construction by sibling child equivalence

function ConstructBySib($p, \mathcal{P}/\equiv_{\text{cov}}$)
  Input: Simpson's paradox $p = (s_1, s_2, X, Y)$; coverage-based partitioning $\mathcal{P}/\equiv_{\text{cov}}$ of populations
  Output: Concise representation of a set of paradoxes sibling-child-equivalent to p
  Let $\mathcal{E}_1, \mathcal{E}_2 \in \mathcal{P}/\equiv_{\text{cov}}$ be groups containing s_1 and s_2, resp.;
  return $(\text{up}(\mathcal{E}_1), \text{low}(\mathcal{E}_1), \text{up}(\mathcal{E}_2), \text{low}(\mathcal{E}_2), \{X\}, \{Y\})$;

Algorithm 5 Extension by separator/statistic equivalence

function ExtendBySepStat($\tilde{\mathbf{P}}, p'$)
  Input: Concise rep. $\tilde{\mathbf{P}}$ for a sibling-child-equivalent paradox group; new Simpson's paradox $p' = (s'_1, s'_2, X', Y')$ redundant with respect to some paradoxes in $\tilde{\mathbf{P}}$
  Output: Updated $\tilde{\mathbf{P}}$
  $(\text{up}(\mathcal{E}_1), \text{low}(\mathcal{E}_1), \text{up}(\mathcal{E}_2), \text{low}(\mathcal{E}_2), \mathbf{X}, \mathbf{Y}) \leftarrow \tilde{\mathbf{P}}$;
  return $(\text{up}(\mathcal{E}_1), \text{low}(\mathcal{E}_1), \text{up}(\mathcal{E}_2), \text{low}(\mathcal{E}_2), \mathbf{X} \cup \{X'\}, \mathbf{Y} \cup \{Y'\})$;

be concisely represented by:

$$\text{up}(\mathcal{E}_1) = \{(*, b_1, *, d_1)\}, \quad \text{low}(\mathcal{E}_1) = \{(*, b_1, *, *), (*, *, *, d_1)\},$$
$$\text{up}(\mathcal{E}_2) = \{(*, b_2, *, d_2)\}, \quad \text{low}(\mathcal{E}_2) = \{(*, b_2, *, *), (*, *, *, d_2)\},$$
$$\mathbf{X} = \{A\}, \quad \mathbf{Y} = \{Y_1\}$$ □

Many paradoxes differ only by separator or label attributes but are still redundant. Once a paradox p' is identified as separator- or statistic-equivalent to some p in a sibling-child-equivalent group $\mathbf{P}$, the separator and label attributes of p' can be applied to all members of $\mathbf{P}$ without enumeration (Algorithm 5).

Proposition 4.8. *Let $\mathbf{P}$ be a set of sibling-child-equivalent Simpson's paradoxes with separator X and label Y. Suppose (s'_1, s'_2, X', Y'), where $X' \neq X$ or $Y' \neq Y$, is a Simpson's paradox that is redundant w.r.t. some paradox in $\mathbf{P}$. Then for every $p = (s_1, s_2, X, Y) \in \mathbf{P}$, the AC (s_1, s_2, X', Y') is also a redundant Simpson's paradox w.r.t. p.* □

Example 4.9 (Extension by separator and statistic equivalence). Continuing from Example 4.7, suppose we have now identified $p' = ((*, *, *, d_1), (*, *, *, d_2), C, Y_2)$ redundant to the sibling-child-equivalent redundant paradox group $\mathbf{P}$ characterized by $\mathcal{E}_1 \times \mathcal{E}_2 \times \{A\} \times \{Y_1\}$. Proposition 4.8 implies that all combinations from $\mathcal{E}_1$ and $\mathcal{E}_2$ with separator C and label Y_2 are also redundant paradoxes. For example, $((*, b_1, *, *), (*, b_2, *, *), C, Y_2)$ can be added directly. The concise representation becomes $\mathcal{E}_1 \times \mathcal{E}_2 \times \{A, C\} \times \{Y_1, Y_2\}$. □

4.2.3 Complete Algorithm. Finally, we integrate these optimizations into a comprehensive algorithm (Algorithm 6). We iterate only over coverage groups (using upper bounds), constructing a sibling-child-equivalence redundant paradox group as soon as one paradox is found, and extending groups with separator and statistic equivalence when applicable. We maintain a hashmap I where each key is $\text{Sig}(p)$ (see Definition 4.10 below) and the value is the concise representation of a redundant paradox group (though a group may contain only a single paradox if no redundancy is observed).

Algorithm 6 Finding non-redundant Simpson's paradoxes

Input: $\mathcal{P}/\equiv_{\text{cov}}$ and STATS as produced by Algorithm 2
Output: Hashmap I storing concise representations of redundant paradox groups, keyed by Sig

```
Initialize I ← ∅;
for each coverage group E ∈ P/≡cov do
    Let s ← up(E)                          ▷ use upper bound as representative
    for each X_i with s[i] = * do
        for each pair v1, v2 ∈ Dom(X_i) do
            s1 ← s⟨X_i ↦ v1⟩,   s2 ← s⟨X_i ↦ v2⟩;
            for each X_j ≠ X_i with s[j] = * do
                for each label attribute Y do
                    p ← (s1, s2, X_j, Y); ▷ check first if p is included in existing redundant paradox groups
                    if ∃(E1, E2, X, Y) ∈ I : s1 ∈ E1 ∧ s2 ∈ E2 ∧ X_j ∈ X ∧ Y ∈ Y then continue
                    Evaluate p acc. Def. 2.1; compute Sig(p);
                    if p is a Simpson's paradox then
                        if I(Sig(p)) = ∅ then
                            I(Sig(p)) ← ConstructBySib(p, P/≡cov);
                        else
                            I(Sig(p)) ← ExtendBySepStat(I(Sig(p)), p);
return I
```

Definition 4.10 (Signature). Given populations s_1 and s_2, we define their *joint signature* with respect to label attribute Y as a triple:

$$\text{JSig}_Y(s_1, s_2) = \langle \text{cov}(s_1), \text{cov}(s_2), \text{sign}(P(Y|s_1) - P(Y|s_2)) \rangle.$$

For an AC $p = (s_1, s_2, X, Y)$, its **signature** $\text{Sig}(p)$ is defined as:

$$\langle\, \text{JSig}_Y(s_1, s_2), \{\text{JSig}_Y(s_1\langle X \mapsto v\rangle, s_2\langle X \mapsto v\rangle) \mid v \in \text{Dom}(X)\} \,\rangle.$$ □

In implementation, coverage sets $\text{cov}(\cdot)$ are represented using integer hashes, and $\text{Sig}(\cdot)$ becomes a vector of integer hashes and sign values from $\{-1, 0, +1\}$. By Lemma 4.11, redundant paradoxes share identical signatures, so a paradox p belongs to an existing redundant paradox group if $\text{Sig}(p)$ is already a key in I.

Lemma 4.11. *Two Simpson's paradoxes p and p' are redundant if and only if $\text{Sig}(p) = \text{Sig}(p')$.* □

These optimizations improve over the brute-force enumeration by avoiding repeated work while producing concise representations. We note below that finding all redundant paradox groups is #P-hard (from #SAT reduction [46]), and provide a worst-case complexity analysis in the artifact supplement.

Theorem 4.12 (#P-Hardness). *Finding all redundant paradox groups in a multidimensional table is #P-hard.* □

4.3 Threshold Selection

We provide brief guidance on choosing the pruning threshold and interpreting the output. For the coverage threshold θ, if a user requires each paradox to involve at least $x\%$ of the records, we recommend setting $\theta = x/(2d)$, where d is the maximum domain cardinality of categorical attributes, since each paradox may involve

up to $2d$ populations. When runtime is a concern, θ can be increased gradually until the computation meets the desired time budget.

To interpret the concise representation, the upper bounds of $\mathcal{E}_1$ and $\mathcal{E}_2$ capture attribute values shared by all records in the corresponding populations; the separator set $\mathcal{X}$ highlights attributes that induce equivalent partitions (i.e., drill-down paths); and the label set $\mathcal{Y}$ identifies outcome attributes that are strongly correlated.

5 EXPERIMENTAL RESULTS

we evaluate our framework for detecting redundant Simpson's paradoxes on both real-world and synthetic datasets, described in Section 5.1. Our study is guided by three research questions (RQs): **RQ1** examines whether redundant Simpson's paradoxes are rare in practice (Section 5.2); **RQ2** examines the scalability of our framework (Section 5.3); and **RQ3** examines the structural robustness of the discovered redundant Simpson's paradoxes (Section 5.4).

Overall, our results show that redundant Simpson's paradoxes prevail in real-world datasets, our method scales effectively, and the paradoxes are structurally robust under data perturbation.

5.1 Experimental Setup

Experiments were conducted on the Duke Computer Science Department's computing cluster, using nodes equipped with Intel Xeon Gold 5317 processors (3.0 GHz, 12 cores) and 64 GB of RAM.

We evaluate our methods on both real-world categorical data and synthetic datasets generated with controlled parameters. The real-world datasets allow us to measure the prevalence of redundant paradoxes, while the synthetic datasets enable systematic assessment of efficiency and scalability under controlled conditions.

5.1.1 Real-World Datasets. We select four representative real-world datasets from diverse domains, covering a range of dimensionalities (8 to 35 attributes) and dataset sizes (8K to 3M records):

- **Adult:** A census income dataset with 48,842 records, 8 attributes (e.g., education, occupation), and a binary label indicating whether annual income exceeds \$50K [4].
- **Mushroom:** A dataset of 8,124 records describing mushrooms using 22 categorical attributes (e.g., cap shape, habitat), with edibility as the binary label [1].
- **Loan:** A large financial dataset[2] containing >3 million loan applications, with 12 categorical attributes (e.g., loan purpose, home ownership) and a label of loan approval.
- **CDC Diabetes Health Indicators:** A healthcare dataset[3] of 253,681 individuals, with 35 categorical attributes (e.g., demographics) and diabetes status as the binary label.

To evaluate our framework on multi-class and multi-label settings (cf. Section 2.2), we include two additional datasets:

- **CMC:** A social science dataset[4] of 1,473 records with 9 categorical attributes and a 3-class label (no use, long-term, short-term contraceptive method). We produce 3 binary copies corresponding to 3 binarization of class labels.

[2]https://www.kaggle.com/datasets/ikpeleambrose/irish-loan-data
[3]https://www.kaggle.com/datasets/alexteboul/diabetes-health-indicators-dataset/data
[4]https://archive.ics.uci.edu/dataset/30/contraceptive+method+choice

Table 4: Simpson's paradoxes in real-world datasets. The top rows report the number of paradoxes and groups, with a breakdown by equivalence type. The bottom rows summarize the most frequent separator attribute (% of all paradoxes), the constrained attribute (% of redundant groups), and the top correlated separator pair (% of separator equivalences).

Dataset	Adult	Mushroom	Loan	Diabetes	CMC (no use)	CMC (long)	CMC (short)	Chicago
#paradoxes	3,880	6,878	18,330	1,464,250	1,806	1,571	1,873	4,893
#groups	3,460	4,931	16,293	1,065,189	1,343	1,226	1,407	3,980
#standalone	3,094	3,590	14,354	809,388	940	921	991	3,249
#sib. child eq.	366	1,220	1,939	255,690	401	305	416	659
#separator eq.	0	146	0	340	2	0	0	0
#statistic eq.	0	0	0	0	0	0	0	126
Top sep.	sex 96%	g-size 57%	term 62%	HighBP 23%	media 33%	media 32%	work 37%	DAY 59%
Top constr. attr.	country 33%	g-space 43%	interest 20%	CholChk 49%	w_rel 49%	w_rel 43%	w_rel 52%	ALIGN 48%
Top corr. sep. pair	—	(g-size, s-shape) 66%	—	(HighBP, DiffWk) 16%	(w_rel, media) 50%	—	—	—

- **Chicago Traffic Crashes**[5]**:** Contains 1,039,019 crash records with 10 categorical attributes (e.g., weather) and 3 binary labels (crash type, injury, and damage).

5.1.2 Synthetic Datasets. To evaluate performance in a controlled setting, we employ a synthetic data generator that accepts the three key parameters: (i) n, number of categorical attributes; (ii) m, number of label attributes; and (iii) d, number of values per attribute.

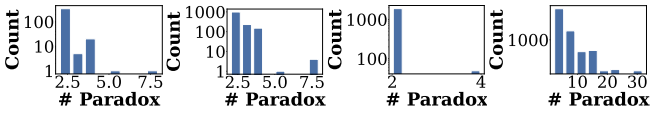
The generation process proceeds in two steps. First, it constructs individual instances of Simpson's paradox. For a given AC (s_1, s_2, X, Y), the generator enforces a consistent trend across all sub-populations and solves an optimization problem to distribute labels that reverses aggregate association. Second, the framework explicitly injects redundancy by modifying the generated records: sibling child equivalence is induced by aligning sibling pairs to have identical coverage, separator equivalence is enforced by mapping domains of two separator attributes via a bijection, and statistic equivalence is created by introducing a correlated label. We provide the detailed procedures in the artifact supplements.

5.2 Q1: Are Redundant Paradoxes Rare?

Our analysis shows that redundant Simpson's paradoxes are common in real-world datasets. As summarized in Table 4, a substantial fraction of discovered paradoxes are redundant: 20.3% in Adult, 47.8% in Mushroom, 21.7% in Loan, and 44.7% in Diabetes. On CMC, we binarize each of the three contraceptive-method classes independently and observe 22–26% redundancy per class. Among the three equivalence types, *sibling child equivalence* is the most prevalent across all datasets. *Separator equivalence*, which requires one-to-one correspondence between separator attributes, appears only in higher-dimensional datasets—Mushroom (10.7%) and Diabetes (0.1%). On Chicago Traffic Crashes, we observe 126 groups with *statistic equivalence*. The remaining single-label datasets show zero statistic equivalence, as expected.

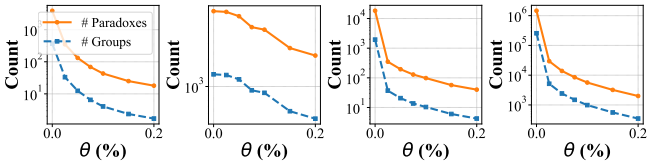
Beyond prevalence, redundant paradoxes exhibit clear patterns (Table 4, bottom rows). First, certain attributes frequently act as

[5]https://data.cityofchicago.org/Transportation/Traffic-Crashes-Crashes/85ca-t3if



(a) Adult (b) Mushroom (c) Loan (d) Diabetes

Figure 2: Distribution of the number of Simpson's paradoxes per redundant group in four primary datasets.



(a) Adult (b) Mushroom (c) Loan (d) Diabetes

Figure 3: Number of Simpson's paradoxes and redundant groups vs. pruning threshold θ on four primary datasets.

Table 5: Pair-separator paradoxes in real-world datasets.

Dataset	Adult	Mushroom	Loan	Diabetes	CMC (no use)	CMC (long)	CMC (short)	Chicago
#paradoxes	1	69	375	278,831	8	14	14	15
#groups	1	57	372	247,193	7	12	11	12
#standalone	1	45	369	222,066	6	10	8	10
#sib. child eq.	0	8	3	25,127	1	2	3	2
#separator eq.	0	4	0	0	0	0	0	0
Top sep. pair	(race, sex) 100%	(g-space, s-shape) 68%	(term, interest) 48%	(HighBP, Sex) 6.1%	(work, media) 57%	(work, media) 67%	(w_age, media) 36%	(DAY, LIGHT) 53%

separators; for example, `sex` appears in 96% of Adult paradoxes, and `term` appears in 62% of Loan paradoxes. Second, we identify *constrained* attributes under sibling child equivalence, where $\text{cov}(s\langle X \to * \rangle) = \text{cov}(s\langle X \to v \rangle)$. In Diabetes, `CholCheck` appears in 49% of redundant groups, indicating strong correlation with other health indicators. Third, in datasets exhibiting separator equivalence (Mushroom and Diabetes), correlated separator pairs induce identical partitions; for example, `(g-size, s-shape)` accounts for 66% of separator equivalences in Mushroom.

Figure 2 shows the distribution of redundant group sizes on the four primary datasets. Most groups contain only 2–3 paradoxes, but in higher-dimensional datasets such as Diabetes, groups can be much larger—up to 32 paradoxes—due to combined effect of sibling child and separator equivalences.

A natural question is whether these redundancy patterns persist under pruning. Figure 3 shows that as θ increases from 0% to 0.2%, both the number of Simpson's paradoxes and redundant groups decrease steadily, with a sharp initial drop since many involve low-coverage populations removed by even small thresholds. Notably, the top separator attributes, constrained attributes, and correlated separator pairs remain stable across θ.

We further evaluate multi-attribute separators (Definition 2.3) by performing pair-separator ($|\mathbf{X}| = 2$) discovery on all datasets. Per Table 5, paradoxes are rarer due to the finer population partitioning.

We further study redundancy in synthetic datasets. With a fixed number of records (Section 5.1.2), the number of non-redundant

Table 6: Simpson's paradoxes in organically generated synthetic data under varying noise levels.

Noise (%)	0	1	2	3	5	7	10
#paradoxes	506	32,904	31,584	35,795	53,924	61,212	76,480
#groups	41	31,803	31,504	35,649	53,603	60,637	76,089
#sib. child eq.	41	486	19	45	164	344	214
#separator eq.	41	280	61	85	115	141	156
#statistic eq.	0	0	0	3	3	14	11

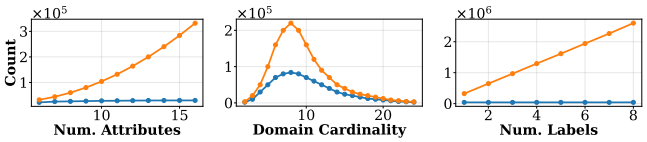


Figure 4: Effect of dataset parameters on the total number of Simpson's paradoxes (orange) and redundant paradox groups (blue) in synthetic data.

paradoxes remains stable, so growth in total paradoxes reflects increased redundancy. As shown in Figure 4, the total number of paradoxes grows exponentially with the number of categorical attributes n, increases then decreases with domain cardinality d, and scales linearly with the number of label attributes m, as each additional label generates statistic-equivalent paradoxes.

To test whether Simpson's paradoxes and redundancies arise naturally, we generate data ($n = 8, d = 4, m = 4$) via ancestral sampling from a Bayesian network [20, 29] with planted inter-attribute correlations (e.g., functional dependencies). We vary the noise level—the only parameter—to progressively weaken these correlations. Table 6 reports the results. At 0% noise, strong correlations yield few distinct populations and thus few paradoxes (506), all of which are redundant (41 groups). As noise increases, correlations weaken, more populations are materialized, and more paradoxes emerge (up to 76,480 at 10% noise), along with more redundancy (214 sibling child, 156 separator, and 11 statistic equivalence groups).

5.3 Q2: Scalability

We evaluate the computational efficiency of our method on real-world and synthetic datasets, yielding three main findings. (i) Our base algorithm (Algorithms 2 and 6) achieves an average 6.72× speedup over brute-force baselines (Algorithms 1 and 3) on real datasets, enabled by DFS-based materialization and redundancy-aware discovery. (ii) Population pruning, which removes populations covering fewer than 0.1% of records, provides an additional 50% reduction in run time and peak memory. (iii) On synthetic data, run time and memory scale predictably with the total number of Simpson's paradoxes, as characterized in Section 5.2 (Figure 4).

5.3.1 Scalability on Real-World Datasets. Figure 5(left) compares three methods: the brute-force approach of Xu et al. [53] (Algorithms 1 and 3), our base algorithm without pruning, and our base algorithm with 0.1% pruning. We use Xu et al. [53] as the baseline, as other methods (e.g., [2, 14, 39]) only detect Simpson's

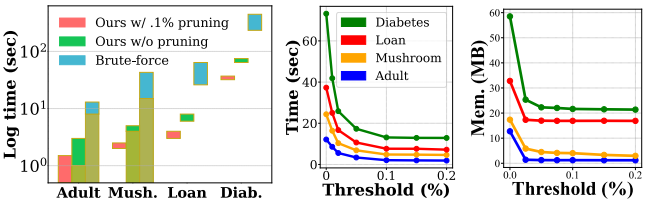


Figure 5: (Left) Run time comparison across three methods (brute-force, base, base+pruning) on four primary datasets. Each bar denotes total runtime, where the yellow shaded region corresponds to materialization time. (Middle) Run time vs. pruning threshold θ on four primary datasets. (Right) Peak memory vs. θ on four primary datasets.

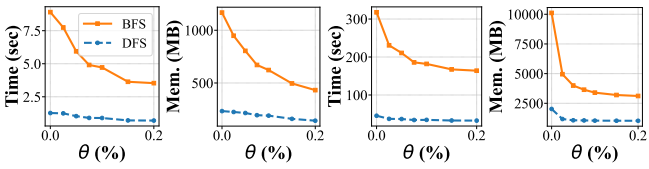


Figure 6: DFS vs. BFS materialization comparison w.r.t. θ. (a, b) Mushroom dataset. (c, d) Diabetes dataset.

paradoxes at the root $(*, \ldots, *)$ of the population lattice (cf. Section 2.1, Figure 1), whereas our framework discovers them across all populations. Across the four primary datasets, the base algorithm achieves an average 6.72× speedup over brute force. This gain comes from two optimizations: DFS-based materialization (Algorithm 2), which provides a 4.94× speedup by avoiding explicit enumeration of intermediate populations, and redundancy-aware discovery (Algorithms 4–6), which achieves a 20.24× speedup by eliminating redundant evaluations. Applying 0.1% pruning further reduces runtime by ∼50%, making the approach practical even for high-dimensional datasets such as Diabetes with 35 attributes.

Figure 5(middle) shows the effect of varying the pruning threshold θ from 0% to 0.2%. Even a small threshold (0.05%) reduces runtime by an average of 41.2%, as many low-coverage populations are pruned early. Gains diminish beyond 0.1%, where most such populations have already been removed. This non-linear trend reflects a heavy-tailed distribution: most populations have small coverage, so a small threshold eliminates them in bulk. Figure 3 shows the same pattern for paradox and redundant group counts.

Figure 5(right) reports peak memory under the same θ settings. Memory follows a similar trend as runtime, since even small thresholds remove most low-coverage populations.

Figure 6 compares materialization time and peak memory of our DFS-based approach with a level-wise BFS variant on Mushroom and Diabetes. DFS consistently outperforms BFS in both time (5–7×) and memory (3–5×). DFS derives the upper bound of each coverage group directly from its lower bound, skipping intermediate populations whose statistics can be inferred (Property 2), and prunes recursion when coverage falls below θ (Section 4.1.3), whereas BFS must materialize all populations at each lattice level.

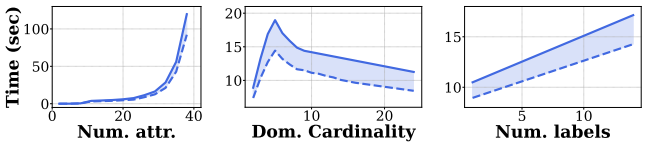


Figure 7: Run time vs. synthetic parameters. Solid lines denote total time; dotted lines show materialization time.

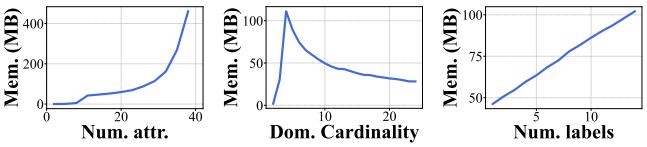


Figure 8: Memory scaling with synthetic dataset parameters.

5.3.2 Scalability on Synthetic Datasets. Figure 7 reports run time w.r.t. the number of categorical attributes (n), domain cardinality (d), and label attributes (m), with default values $n = 8$, $d = 8$, and $m = 4$. All three redundancy types are explicitly injected, and datasets with up to 30 million records are generated to stress-test scalability. Each plot reports both materialization and total time.

Run time trends closely follow the total number of Simpson's paradoxes in Figure 4, since materialization dominates computation and depends on the number of populations. As n increases (from 2 to 40 in our experiments), the population lattice grows exponentially, leading to exponential growth. For very large n (>100), the problem becomes computationally prohibitive (cf. Theorem 4.12); population pruning (cf. Section 4.1.3) and attribute selection (e.g., removing correlated attributes) can help mitigate this. Increasing d initially raises runtime but later reduces it, as larger domains decrease the number of paradoxes and materialized populations. Increasing m leads to linear scaling, since frequency statistics are computed for the same populations under additional labels.

By contrast, paradox discovery time remains nearly constant across all settings, because the number of redundant paradox groups is fixed by construction in the synthetic data (Figure 4).

Figure 8 reports peak memory under the same parameters. Overall, memory follows the same trends as run time.

5.4 Q3: Are Redundant Paradoxes Robust?

Beyond demonstrating the prevalence of redundant Simpson's paradoxes, we now investigate whether these paradoxical reversals and redundancies reflect *genuine structural properties* of the data or are merely random artifacts introduced by noise or errors in data collection. To this end, we adopt a perturbation-based framework to test the *robustness* of paradoxes and redundancies.

At a high level, we measure *tolerance:* given an observed Simpson's paradox (or redundancy), we quantify how much the data can be perturbed before the paradoxical (or redundant) relationship disappears. Robust patterns should persist under small perturbations, whereas random artifacts should vanish quickly.

We evaluate two aspects: (1) the **robustness** of individual Simpson's paradoxes under label and record perturbations; and (2) the **persistence** of coverage redundancies under record perturbations.

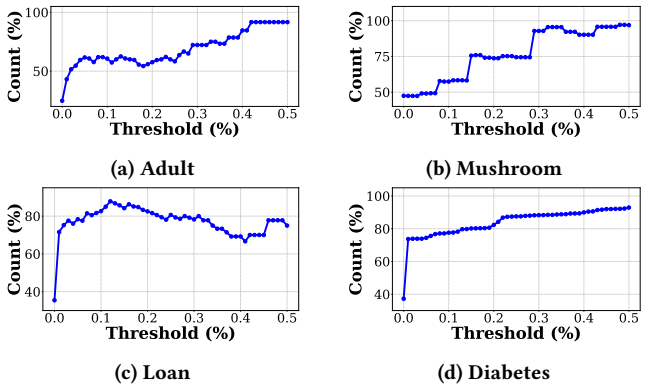


Figure 9: Fraction of structurally robust Simpson's paradoxes vs. pruning threshold across four primary datasets.

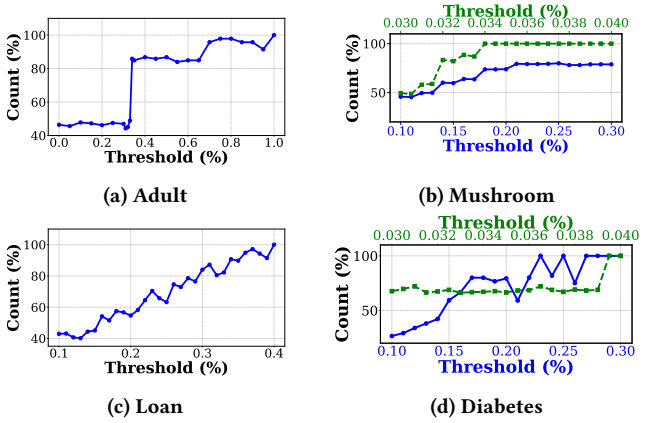


Figure 10: Fraction of robust redundant paradox groups vs. pruning thresholds. The blue curves show redundancy due to sibling child equivalence; the green curves show redundancy due to separator equivalence.

5.4.1 Robustness of Individual Simpson's Paradoxes. We first examine whether Simpson's paradoxes persist under perturbations. The frequency statistic $P(Y|s)$ is a weighted average of sub-population statistics $P(Y|s\langle X \nrightarrow v\rangle)$, where the weights are determined by the record distribution $|\operatorname{cov}(s\langle X \nrightarrow v\rangle)|/|\operatorname{cov}(s)|$. Simpson's paradoxes emerge from specific interactions between these two components.

For each $p = (s_1, s_2, X, Y)$, we apply two perturbations:

- **Label perturbation:** Randomly flip labels of 5% of the records in $\operatorname{cov}(s_1) \cup \operatorname{cov}(s_2)$ to alter the frequency statistics $P(Y|s\langle X \nrightarrow v\rangle)$. This tests whether paradoxical reversals depend critically on exact label assignments.
- **Coverage perturbation:** Randomly modify X for 5% of records to change the weights $|\operatorname{cov}(s\langle X \nrightarrow v\rangle)|/|\operatorname{cov}(s)|$ across sub-populations. We then reassign labels in each sub-population according to their original frequency statistics, isolating the effect of record distribution.

Each perturbation is repeated 10,000 times, and robustness is measured by the survival rate, defined as the percentage of trials in which the paradox persists. Figure 9 reports robustness under label perturbations. The fraction of *robust* paradoxes—those surviving at least 95% of trials—increases with higher pruning thresholds. This trend indicates that paradoxes supported by larger populations are less sensitive to individual label flips, since their frequency statistics $P(Y \mid s)$ are computed over more records and thus have lower variance. Across all four primary datasets, a pruning threshold of 0.3% is sufficient to ensure that more than 70% of paradoxes are structurally robust. Slight decreases at higher thresholds are expected, as the total number of paradoxes declines when additional populations are pruned due to insufficient coverage.

5.4.2 Robustness of Redundant Groups. Sibling child and separator equivalence rely on populations with identical coverage, which can be organized into convex coverage groups. Redundancy is therefore robust if coverage identicality persists under perturbations. We test robustness with two strategies:

- **Sibling child equivalence** (Lemma 3.1): For sibling-child-equivalent paradoxes $p = (s_1, s_2, X, Y)$ and $p' = (s'_1, s'_2, X, Y)$, we randomly alter one attribute value in 5% of the records in $\operatorname{cov}(s_1) \cup \operatorname{cov}(s_2)$. This tests whether $\operatorname{cov}(s_1) = \operatorname{cov}(s'_1)$ and $\operatorname{cov}(s_2) = \operatorname{cov}(s'_2)$ remain intact.
- **Separator equivalence** (Lemma 3.3): For separator-equivalent paradoxes $p = (s_1, s_2, X, Y)$ and $p' = (s_1, s_2, X', Y)$, we perturb 5% of records in $\operatorname{cov}(s_1) \cup \operatorname{cov}(s_2)$ on attributes other than X and X'. The one-to-one mapping $f : X \mapsto X'$ is preserved, and we test whether equivalence $\operatorname{cov}(s\langle X \nrightarrow v\rangle) = \operatorname{cov}(s\langle X' \nrightarrow f(v)\rangle)$ persists.

As before, each perturbation is repeated 10,000 times and robustness is measured by the survival rate. Figure 10 shows that the fraction of robust redundant groups increases with higher pruning thresholds, as sibling- and separator-equivalent paradoxes supported by larger populations are more tolerant to perturbations. This robustness is structural: coverage equivalence persists unless perturbations alter record values enough to break $\operatorname{cov}(s) = \operatorname{cov}(s')$. In Mushroom and Diabetes, robustness of separator equivalence (green curves) remains stable across thresholds due to the stricter one-to-one correspondence required between separator attributes.

5.4.3 Sensitivity Analysis. Figure 11 reports robustness on Diabetes under varying perturbation rates $\varepsilon \in \{1\%, 5\%, 10\%\}$ and survival thresholds $\tau \in \{90\%, 95\%, 99\%\}$. For individual Simpson's paradoxes (Figure 11(a)), increasing ε reduces the fraction of robust paradoxes, as perturbations to $\operatorname{cov}(s_1) \cup \operatorname{cov}(s_2)$ are more likely to alter $P(Y \mid s)$ and flip the paradoxical sign. Similarly, increasing τ (Figure 11(c)) enforces a stricter survival criterion, further reducing robustness. In both cases, the increasing trend with θ persists (Figure 11(e)).

For sibling child equivalence groups (Figure 11(b, d, f)), the same patterns hold: larger ε increases the chance of breaking coverage identicality, and larger τ imposes stricter survival. Nevertheless, the increasing trend with θ remains consistent, indicating that our findings are relatively insensitive to the choice of ε and τ.

5.5 Case Study: Redundancy in Diabetes Risk

We illustrate the practical value of redundancy detection and our framework using a case from the Diabetes dataset.

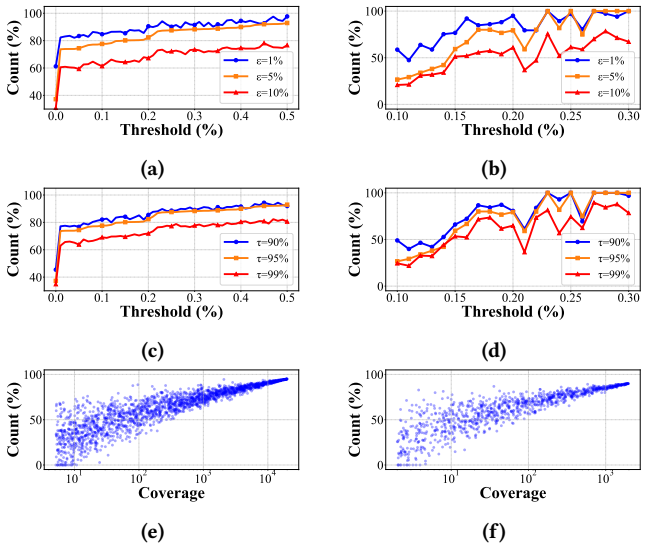


Figure 11: Sensitivity of robustness to perturbation rate ε and survival threshold τ on Diabetes. Left column: individual Simpson's paradoxes; right column: sibling child equivalence groups. (a, b) Fraction of robust paradoxes/groups under varying ε with $\tau = 95\%$. (c, d) Under varying τ with $\varepsilon = 5\%$. (e, f) Survival fraction vs. population coverage at $\varepsilon = 5\%$.

An Example Paradox. Consider individuals with income \$20,000–\$25,000 and heavy alcohol consumption. In the aggregate, younger individuals (s_2, Age=8) exhibit lower diabetes prevalence than older individuals (s_1, Age=13), i.e., $P(\text{Diabetes} \mid s_2) < P(\text{Diabetes} \mid s_1)$. However, when partitioned by DiffWalk (difficulty walking; 0=no, 1=yes), this trend reverses in both subgroups: $P(\text{Diabetes} \mid s_2\langle \text{DiffWalk} \curvearrowright v\rangle) > P(\text{Diabetes} \mid s_1\langle \text{DiffWalk} \curvearrowright v\rangle)$, $v \in \{0, 1\}$.

Redundancy. This paradox is redundant with 29 others due to:

(1) **Sibling child equivalence** (Lemma 3.1). The siblings (HighBP = 1, HighChol = 1, ..., Age = 13) and (HighBP = 1, HighChol = 1, ..., Age = 8) have the same coverage as the siblings (HighBP = $*$, HighChol = $*$, ..., Age = 13) and (HighBP = $*$, HighChol = $*$, ..., Age = 8). The sibling pairs from $\mathcal{E}_1 \times \mathcal{E}_2$ generate 10 sibling-equivalent paradoxes.
(2) **Separator equivalence** (Lemma 3.3). DiffWalk, Smoker, and HeartDiseaseorAttack induce identical partitions.

Practical Implications. This redundancy exposes strong attribute correlations in this high-risk cohort. HighBP and HighChol are strongly associated among heavy alcohol consumers, causing sibling child equivalence, while DiffWalk, Smoker, and HeartDiseaseorAttack are correlated cardiovascular indicators, causing separator equivalence. Our framework condenses these 30 paradoxes into a single interpretable pattern: age and cardiovascular health jointly shape diabetes risk among heavy alcohol consumers.

Summary. Our experiments show that redundant Simpson's paradoxes are common in real-world datasets, our framework scales efficiently, and the discovered paradoxes and redundancies remain stable under data perturbations. Our case study demonstrates that the concise representation condenses redundant paradoxes into interpretable patterns that expose underlying attribute correlations.

6 RELATED WORK

To the best of our knowledge, we are the first to study redundancy among Simpson's paradoxes. Our work relates to paradox finding in high-dimensional data and summarizing data populations.

Detecting Simpson's Paradox. Since the introduction of Association Reversal and Amalgamation Paradox [15, 40, 57], Simpson's paradox has been widely studied. Early methods relied on statistical modeling, such as Bayesian networks [14] and surprisingness ranking [13]. Later approaches used statistical tests [2], correlation-based methods for continuous and categorical data [39, 52], and broader frameworks such as bias detection and query rewriting [35–37] and confounder discovery [27]. Recent work includes neural disaggregation [49], federated counterfactual learning [21], and regression-tree-based methods [32].

Most methods focus on global analysis and overlook paradoxes in localized sub-populations. The closest work is Xu et al. [53], whose brute-force approach (Alg. 1 and 3) enumerates coverage-equivalent populations and redundant paradoxes. Our DFS-based materialization and redundancy-aware discovery avoid this redundancy, achieving an average 6.72× speedup (Section 5.3.2).

Summarizing Data Populations. Our approach groups populations with identical coverage, which is closely related to data cube research [11, 16, 31, 51]. Prior work studied efficient materialization [3, 33, 55] and cube condensation [41, 50]. Ours is closely related to quotient cubes [23, 24], which partition the lattice into equivalence classes; our coverage groups correspond to such cells, which exhibit convexity (Property 1) and support reconstruction (Property 2). Our pruning is also related to iceberg cubes [5, 18].

Unlike prior work that focuses on storage and query optimization, we leverage population organization to efficiently identify and summarize redundant Simpson's paradoxes.

7 CONCLUSIONS

We studied redundancy in Simpson's paradox and showed that many paradoxes in multidimensional data are redundant due to identical coverage or equivalent partitioning and label attributes. We formalized three types of redundancy, proved that they form an equivalence relation, and proposed a concise representation based on the convexity properties of the population lattice. Building on this, we developed efficient algorithms that combine depth-first materialization, pruning, and redundancy-aware discovery to find all non-redundant paradoxes. Experiments on real and synthetic datasets demonstrate that redundancy is prevalent, that our methods scale efficiently, and the identified paradoxes are robust.

ACKNOWLEDGMENTS

This work was supported in part by NSF grants MSPA-2434666 and IIS-2402823, and NIH grant R01HG012555. All opinions, findings, conclusions, and recommendations in this paper are those of the authors and do not necessarily reflect the views of the funding agencies.

REFERENCES

[1] 1987. Mushroom. UCI Machine Learning Repository. https://doi.org/10.24432/C5959T Donated on 4/26/1987. Accessed: 2026-05-18.

[2] Nazanin Alipourfard, Peter G. Fennell, and Kristina Lerman. 2018. Can you trust the trend? Discovering Simpson's paradoxes in social data. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining (WSDM '18)*. ACM, Marina Del Rey, CA, USA, 19–27. https://doi.org/10.1145/3159652.3159684

[3] Sachin Basil John, Christoph Koch, and Peter Lindner. 2025. High-dimensional Data Cubes. *ACM Trans. Database Syst.* 50, 3, Article 11 (June 2025), 43 pages. https://doi.org/10.1145/3716373

[4] Barry Becker and Ronny Kohavi. 1996. Adult. UCI Machine Learning Repository. https://doi.org/10.24432/C5XW20 Accessed: 2026-05-18.

[5] Kevin Beyer and Raghu Ramakrishnan. 1999. Bottom-Up Computation of Sparse and Iceberg CUBEs. In *Proceedings of the 1999 ACM SIGMOD International Conference on Management of Data*. ACM, Philadelphia, Pennsylvania, USA, 359–370. https://doi.org/10.1145/304182.304214

[6] Peter J. Bickel, Eugene A. Hammel, and J. William O'Connell. 1975. Sex bias in graduate admissions: Data from Berkeley. *Science* 187, 4175 (1975), 398–404. https://doi.org/10.1126/science.187.4175.398

[7] Francesco Bonchi, Francesco Gullo, Bud Mishra, and Daniele Ramazzotti. 2018. Probabilistic Causal Analysis of Social Influence. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management (CIKM '18)*. ACM, Torino, Italy, 1003–1012. https://doi.org/10.1145/3269206.3271756

[8] Michael Brimacombe. 2014. Genomic aggregation effects and Simpson's paradox. *Open Access Medical Statistics* 4 (2014), 1–6. https://doi.org/10.2147/OAMS.S52288

[9] Christopher R. Charig, David R. Webb, Stephen R. Payne, and John E. A. Wickham. 1986. Comparison of treatment of renal calculi by open surgery, percutaneous nephrolithotomy, and extracorporeal shockwave lithotripsy. *British Medical Journal (Clinical Research Ed.)* 292, 6524 (1986), 879–882. https://doi.org/10.1136/bmj.292.6524.879

[10] Changqing Chen, Jianlin Feng, and Longgang Xiang. 2003. Computation of Sparse Data Cubes with Constraints. In *Data Warehousing and Knowledge Discovery, 5th International Conference, DaWaK 2003*. Springer, Prague, Czech Republic, 14–23. https://doi.org/10.1007/978-3-540-45228-7_3

[11] Yu Chen, Jinguo You, Benyuan Zou, Guoyu Gan, Ting Zhang, and Lianyin Jia. 2020. Exploring Structural Characteristics of Lattices in Real World. *Complexity* 2020, 1 (2020), 1250106. https://doi.org/10.1155/2020/1250106

[12] Yuhao Deng, Yu Wang, Lei Cao, Lianpeng Qiao, Yuping Wang, Jingzhe Xu, Yizhou Yan, and Samuel Madden. 2024. Outlier Summarization via Human Interpretable Rules. *Proceedings of the VLDB Endowment* 17, 7 (2024), 1591–1604. https://doi.org/10.14778/3654621.3654627

[13] Carem C. Fabris and Alex A. Freitas. 2006. Discovering Surprising Instances of Simpson's Paradox in Hierarchical Multidimensional Data. *International Journal of Data Warehousing and Mining (IJDWM)* 2, 1 (2006), 27–49. https://doi.org/10.4018/jdwm.2006010102

[14] Alex A. Freitas, Ken McGarry, and Elon S. Correa. 2007. Integrating Bayesian Networks and Simpson's Paradox in Data Mining. In *Causality and Probability in the Sciences*, Federica Russo and Jon Williamson (Eds.). College Publications, London, UK, 43–62.

[15] Irving John Good and Yashaswini Mittal. 1987. The amalgamation and geometry of two-by-two contingency tables. *The Annals of Statistics* 15, 2 (1987), 694–711. https://doi.org/10.1214/aos/1176350369

[16] Jim Gray, Adam Bosworth, Andrew Layman, and Hamid Pirahesh. 1996. Data Cube: A Relational Aggregation Operator Generalizing Group-By, Cross-Tab, and Sub-Total. In *Proceedings of the Twelfth International Conference on Data Engineering (ICDE '96)*. IEEE Computer Society, New Orleans, Louisiana, USA, 152–159. https://doi.org/10.1109/ICDE.1996.492099

[17] Yue Guo, Carsten Binnig, and Tim Kraska. 2017. What You See is Not What You Get! Detecting Simpson's Paradoxes During Data Exploration. In *Proceedings of the 2nd Workshop on Human-In-the-Loop Data Analytics (HILDA@SIGMOD)*. ACM, Chicago, IL, USA, 2:1–2:5. https://doi.org/10.1145/3077257.3077266

[18] Jiawei Han, Jian Pei, Guozhu Dong, and Ke Wang. 2001. Efficient computation of iceberg cubes with complex measures. In *Proceedings of the 2001 ACM SIGMOD International Conference on Management of Data*. ACM, Santa Barbara, CA, USA, 1–12. https://doi.org/10.1145/375663.375664

[19] Venky Harinarayan, Anand Rajaraman, and Jeffrey D. Ullman. 1996. Implementing Data Cubes Efficiently. *ACM SIGMOD Record* 25, 2 (1996), 205–216. https://doi.org/10.1145/235968.233333

[20] David Heckerman. 2008. A Tutorial on Learning with Bayesian Networks. In *Innovations in Bayesian Networks: Theory and Applications*, Dawn E. Holmes and Lakhmi C. Jain (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 33–82. https://doi.org/10.1007/978-3-540-85066-3_3

[21] Zhonghua Jiang, Jimin Xu, Shengyu Zhang, Tao Shen, Jiwei Li, Kun Kuang, Haibin Cai, and Fei Wu. 2025. FedCFA: Alleviating Simpson's Paradox in Model Aggregation with Counterfactual Federated Learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 39. AAAI Press, Philadelphia, Pennsylvania, USA, 17662–17670. https://doi.org/10.1609/aaai.v39i17.33942

[22] Sanjay Krishnan, Jiannan Wang, Eugene Wu, Michael J. Franklin, and Ken Goldberg. 2016. ActiveClean: Interactive Data Cleaning For Statistical Modeling. *Proceedings of the VLDB Endowment* 9, 12 (2016), 948–959. https://doi.org/10.14778/2994509.2994514

[23] Laks V. S. Lakshmanan, Jian Pei, and Jiawei Han. 2002. Quotient Cube: How to Summarize the Semantics of a Data Cube. In *VLDB 2002, Proceedings of the 28th International Conference on Very Large Data Bases*. Morgan Kaufmann, Hong Kong, China, 778–789. https://doi.org/10.1016/B978-155860869-6/50074-3

[24] Laks V. S. Lakshmanan, Jian Pei, and Yan Zhao. 2003. QC-Trees: An Efficient Summary Structure for Semantic OLAP. In *Proceedings of the 2003 ACM SIGMOD International Conference on Management of Data*. ACM, San Diego, California, USA, 64–75. https://doi.org/10.1145/872757.872768

[25] Hans-Joachim Lenz and Arie Shoshani. 1997. Summarizability in OLAP and Statistical Data Bases. In *Proceedings of the Ninth International Conference on Scientific and Statistical Database Management (SSDBM '97)*. IEEE Computer Society, Olympia, WA, USA, 132–143. https://doi.org/10.1109/SSDM.1997.621175

[26] Yin Lin, Brit Youngmann, Yuval Moskovitch, H. V. Jagadish, and Tova Milo. 2021. On Detecting Cherry-picked Generalizations. *Proceedings of the VLDB Endowment* 15, 1 (2021), 59–71. https://doi.org/10.14778/3485450.3485457

[27] Guimei Liu, Mengling Feng, Yue Wang, Limsoon Wong, See-Kiong Ng, Tzia Liang Mah, and Edmund Jon Deoon Lee. 2011. Towards exploratory hypothesis testing and analysis. In *2011 IEEE 27th International Conference on Data Engineering (ICDE)*. IEEE, Hannover, Germany, 745–756. https://doi.org/10.1109/ICDE.2011.5767907

[28] Y. Zee Ma. 2015. Simpson's paradox in GDP and per capita GDP growths. *Empirical Economics* 49, 4 (2015), 1301–1315. https://doi.org/10.1007/s00181-015-0921-3

[29] Judea Pearl. 2014. Comment: Understanding Simpson's Paradox. *The American Statistician* 68, 1 (2014), 8–13. https://doi.org/10.1080/00031305.2014.876829

[30] Karl Pearson, Alice Lee, and Leslie Bramley-Moore. 1899. VI. Mathematical contributions to the theory of evolution. —VI. Genetic (reproductive) selection: Inheritance of fertility in man, and of fecundity in thoroughbred racehorses. *Philosophical Transactions of the Royal Society of London, Series A: Containing Papers of a Mathematical or Physical Character* 192 (12 1899), 257–330. https://doi.org/10.1098/rsta.1899.0006

[31] Viet Phan-Luong. 2016. A Data Cube Representation for Efficient Querying and Updating. In *2016 International Conference on Computational Science and Computational Intelligence (CSCI)*. IEEE, Las Vegas, Nevada, USA, 415–420. https://doi.org/10.1109/CSCI.2016.0085

[32] Eduarda Portela, Rita P. Ribeiro, and João Gama. 2019. The search of conditional outliers. *Intelligent Data Analysis* 23, 1 (2019), 23–39. https://doi.org/10.3233/IDA-173619

[33] Kenneth A. Ross and Divesh Srivastava. 1997. Fast computation of sparse datacubes. In *Proceedings of the 23rd International Conference on Very Large Data Bases (VLDB '97)*. Morgan Kaufmann, Athens, Greece, 116–125.

[34] Ricardo Salazar, Felix Neutatz, and Ziawasch Abedjan. 2021. Automated Feature Engineering for Algorithmic Fairness. *Proceedings of the VLDB Endowment* 14, 9 (2021), 1694–1702. https://doi.org/10.14778/3461535.3463474

[35] Babak Salimi, Corey Cole, Peter Li, Johannes Gehrke, and Dan Suciu. 2018. HypDB: A Demonstration of Detecting, Explaining and Resolving Bias in OLAP Queries. *Proceedings of the VLDB Endowment* 11, 12 (2018), 2062–2065. https://doi.org/10.14778/3229863.3236260

[36] Babak Salimi, Johannes Gehrke, and Dan Suciu. 2018. Bias in OLAP Queries: Detection, Explanation, and Removal. In *Proceedings of the 2018 International Conference on Management of Data (SIGMOD '18)*. ACM, Houston, TX, USA, 1021–1035. https://doi.org/10.1145/3183713.3196914

[37] Babak Salimi, Bill Howe, and Dan Suciu. 2020. Database Repair Meets Algorithmic Fairness. *ACM SIGMOD Record* 49, 1 (2020), 34–41. https://doi.org/10.1145/3422648.3422657

[38] Myra L. Samuels. 1993. Simpson's paradox and related phenomena. *J. Amer. Statist. Assoc.* 88, 421 (1993), 81–88. https://doi.org/10.1080/01621459.1993.10594297

[39] Rahul Sharma, Huseyn Garayev, Minakshi Kaushik, Sijo Arakkal Peious, Prayag Tiwari, and Dirk Draheim. 2022. Detecting Simpson's Paradox: A Machine Learning Perspective. In *Database and Expert Systems Applications, 33rd International Conference, DEXA 2022*. Springer, Vienna, Austria, 323–335. https://doi.org/10.1007/978-3-031-12423-5_25

[40] E. H. Simpson. 1951. The Interpretation of Interaction in Contingency Tables. *Journal of the Royal Statistical Society: Series B (Methodological)* 13, 2 (1951), 238–241. https://doi.org/10.1111/j.2517-6161.1951.tb00088.x

[41] Yannis Sismanis, Antonios Deligiannakis, Nick Roussopoulos, and Yannis Kotidis. 2002. Dwarf: Shrinking the PetaCube. In *Proceedings of the 2002 ACM SIGMOD International Conference on Management of Data*. ACM, Madison, Wisconsin, USA, 464–475. https://doi.org/10.1145/564691.564745

[42] Peter Spirtes, Clark Glymour, and Richard Scheines. 2000. *Causation, Prediction, and Search* (2nd ed.). MIT Press, Cambridge, MA, USA.

[43] Jan Sprenger and Naftali Weinberger. 2021. Simpson's Paradox. In *The Stanford Encyclopedia of Philosophy* (summer 2021 ed.), Edward N. Zalta (Ed.). Metaphysics Research Lab, Stanford University, Stanford, CA, USA. https://plato.stanford.edu/archives/sum2021/entries/paradox-simpson/ Accessed: 2026-05-18.

[44] Guanting Tang, James Bailey, Jian Pei, and Guozhu Dong. 2013. Mining Multidimensional Contextual Outliers from Categorical Relational Data. In *Proceedings of the 25th International Conference on Scientific and Statistical Database Management*. ACM, Baltimore, Maryland, USA, Article 43, 4 pages. https://doi.org/10.1145/2484838.2484883

[45] Yu-Kang Tu, David Gunnell, and Mark S. Gilthorpe. 2008. Simpson's Paradox, Lord's Paradox, and Suppression Effects are the Same Phenomenon – the Reversal Paradox. *Emerging Themes in Epidemiology* 5, 1 (2008), 2:1–2:9. https://doi.org/10.1186/1742-7622-5-2

[46] Leslie G. Valiant. 1979. The complexity of enumeration and reliability problems. *SIAM J. Comput.* 8, 3 (1979), 410–421. https://doi.org/10.1137/0208032

[47] Jeffrey Scott Vitter, Min Wang, and Bala Iyer. 1998. Data cube approximation and histograms via wavelets. In *Proceedings of the Seventh International Conference on Information and Knowledge Management (CIKM)*. ACM, Bethesda, MD, USA, 96–104. https://doi.org/10.1145/288627.288645

[48] Clifford H. Wagner. 1982. Simpson's paradox in real life. *The American Statistician* 36, 1 (1982), 46–48. https://doi.org/10.1080/00031305.1982.10482778

[49] Jingwei Wang, Jianshan He, Weidi Xu, Ruopeng Li, and Wei Chu. 2023. Learning to Discover Various Simpson's Paradoxes. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '23)*. ACM, Long Beach, CA, USA, 5092–5103. https://doi.org/10.1145/3580305.3599859

[50] Wei Wang, Hongjun Lu, Jianlin Feng, and Jeffrey Xu Yu. 2002. Condensed Cube: An Efficient Approach to Reducing Data Cube Size. In *Proceedings of the 18th International Conference on Data Engineering (ICDE '02)*. IEEE Computer Society, USA, 155–165.

[51] Xike Xie, Xingjun Hao, Torben Bach Pedersen, Peiquan Jin, and Jinchuan Chen. 2016. OLAP over Probabilistic Data Cubes I: Aggregating, Materializing, and Querying. In *2016 IEEE 32nd International Conference on Data Engineering (ICDE)*. IEEE, Helsinki, Finland, 799–810. https://doi.org/10.1109/ICDE.2016.7498291

[52] Chenguang Xu, Sarah M. Brown, and Christan Grant. 2018. Detecting Simpson's Paradox. In *Proceedings of the Thirty-First International Florida Artificial Intelligence Research Society Conference (FLAIRS 2018)*. AAAI Press, Melbourne, Florida, USA, 221–224.

[53] Jay Xu, Jian Pei, and Zicun Cong. 2022. Finding Multidimensional Simpson's Paradox. *ACM SIGKDD Explorations Newsletter* 24, 2 (2022), 48–60. https://doi.org/10.1145/3575637.3575645

[54] Hanieh Yaghootkar, Michael P. Bancks, Sam E. Jones, Aaron McDaid, Robin Beaumont, Louise Donnelly, Andrew R. Wood, Archie Campbell, Jessica Tyrrell, Lynne J. Hocking, Marcus A. Tuke, Katherine S. Ruth, Ewan R. Pearson, Anna Murray, Rachel M. Freathy, Patricia B. Munroe, Caroline Hayward, Colin Palmer, Michael N. Weedon, James S. Pankow, Timothy M. Frayling, and Zoltán Kutalik. 2017. Quantifying the extent to which index event biases influence large genetic association studies. *Human Molecular Genetics* 26, 5 (2017), 1018–1030. https://doi.org/10.1093/hmg/ddw433

[55] Jinguo You, Yuxuan Wang, Xingrui Huang, Zhenrui Yi, Wanting Fu, Kaiqi Liu, Pengchen Zhang, and Bin Yao. 2025. SOC: A Succinct Adaptive Semantic OLAP Caching. *Data Science and Engineering* 10, 4 (2025), 621–638. https://doi.org/10.1007/s41019-025-00290-1

[56] Brit Youngmann, Michael Cafarella, Amir Gilad, and Sudeepa Roy. 2024. Summarized Causal Explanations for Aggregate Views. *Proceedings of the ACM on Management of Data* 2, 1 (2024), 71:1–71:27. https://doi.org/10.1145/3639328

[57] G. Udny Yule. 1903. Notes on the theory of association of attributes in statistics. *Biometrika* 2, 2 (1903), 121–134. https://doi.org/10.1093/biomet/2.2.121

[58] Yihong Zhao, Prasad M. Deshpande, and Jeffrey F. Naughton. 1997. An array-based algorithm for simultaneous multidimensional aggregates. In *Proceedings of the 1997 ACM SIGMOD International Conference on Management of Data*. ACM, Tucson, AZ, USA, 159–170. https://doi.org/10.1145/253260.253288

[59] Pierre Zuyderhoff, Michel Denuit, and Julien Trufin. 2025. Simpson's Paradox for Kendall's Rank Coefficient. *Methodology and Computing in Applied Probability* 27, 2 (2025), 29. https://doi.org/10.1007/s11009-025-10161-x