FernUniversität in Hagen
Faculty of Mathematics and Computer Science

Artificial Intelligence Group

An empirical correlation analysis of ranking semantics for abstract argumentation frameworks

Bachelor's Thesis

in partial fulfillment of the requirements for
the degree of Bachelor of Science (B.Sc.)
in Informatik

submitted by
Marcell Jawhari

First examiner:	Univ.-Prof. Dr. Matthias Thimm Artificial Intelligence Group
Advisor:	Dr. Kai Sauerwald Artificial Intelligence Group

Statement

I declare that I have written the bachelor´s thesis independently and without unauthorized use of third parties. I have only used the indicated resources and I have clearly marked the passages taken verbatim or in the sense of these resources as such. The assurance of independent work also applies to any drawings, sketches or graphical representations. The work has not previously been submitted in the same or similar form to the same or another examination authority and has not been published. By submitting the electronic version of the final version of the bachelor´s thesis, I acknowledge that it will be checked by a plagiarism detection service to check for plagiarism and that it will be stored exclusively for examination purposes.

I explicitly agree to have this thesis published on the webpage of the artificial intelligence group and endorse its public availability.

Software created for this work has been made available as open source; a corresponding link to the sources is included in this work. The same applies to any research data.

Streifing, 12.08.2025 .....

(Place, Date) (Signature)

Zusammenfassung

Traditionelle extensionsbasierte Semantiken in der abstrakten Argumentation bieten nur eine binäre Klassifizierung von Argumenten. Rangsemantiken stellen eine nuanciertere Alternative dar, indem sie jedem Argument einen Stärkewert zuweisen, was für viele Anwendungen entscheidend ist.

Diese Arbeit präsentiert eine empirische Korrelationsanalyse prominenter Rangsemantiken, darunter die diskussionsbasierte (Dbs), kategorisierungsbasierte (Cat), serialisierbarkeitsbasierte (Ser) und probabilistische Ansätze. Die von diesen Semantiken erzeugten Ranglisten wurden auf Benchmark-Datensätzen der International Competition on Computational Models of Argumentation (ICCMA) mittels Kendall-Tau und Spearman-Rho verglichen.

Die Ergebnisse zeigen eine geringe Gesamtkorrelation, was auf eine erhebliche konzeptionelle Vielfalt hindeutet. Der Grad der Übereinstimmung hängt entscheidend von den strukturellen Eigenschaften des Frameworks ab, insbesondere vom Vorhandensein von Zyklen und der Angriffsdichte. Dies belegt, dass es keine universell überlegene Semantik gibt. Die Auswahl muss sich am Kontext der Anwendung orientieren, und diese Arbeit liefert eine empirische Evidenz als Entscheidungshilfe.

Abstract

Traditional extension-based semantics in abstract argumentation offer only a binary classification of arguments. Ranking-based semantics provide a more nuanced alternative by assigning a strength score to each argument, which is crucial for many applications.

This thesis presents an empirical correlation analysis of prominent ranking semantics, including Discussion-based (Dbs), Categoriser-based (Cat), Serialisability-based (Ser), and Probabilistic approaches. The rankings produced by these semantics were compared on benchmark datasets from the International Competition on Computational Models of Argumentation (ICCMA) using Kendall's Tau and Spearman's Rho.

The findings show a low overall correlation, indicating significant conceptual diversity. The level of agreement is critically dependent on the framework's structural properties, particularly the presence of cycles and attack density. In other words, which method you choose depends on the specific type of argumentation framework you're working with. The selection must be informed by the application's context, and this work gives evidence to help choose.

Contents

1 Introduction

Abstract argumentation frameworks, introduced by Dung in 1995, serve as a foundational formalism for representing argumentative scenarios [Dun95]. These frameworks model arguments as nodes in a directed graph, with directed edges representing attacks between arguments. This abstraction provides a versatile tool for reasoning with conflicting information and has been applied in fields such as decision-making, legal reasoning, and multi-agent systems [Dun95, Del17]. One critical aspect of analyzing abstract argumentation frameworks is the evaluation of arguments based on their acceptability or plausibility.

In a debate, the goal is often to identify sets of arguments that can be collectively deemed acceptable, balancing coherence and defense against opposition. This pursuit leads to the concept of the preferred extension, which identifies maximal admissible sets of arguments. An admissible set is one that is conflict-free and where every argument in the set is defended against attacks from arguments outside the set.

Building on the foundation described above, the stable extension semantics extends the notion of conflict-freeness by requiring that every argument outside the extension is attacked by at least one argument within it. This ensures that stable extensions not only defend themselves but also actively counter opposing arguments. Together, these semantics provide a systematic method for determining sets of arguments that can be considered acceptable within the framework. However, they ultimately offer a binary analysis, classifying arguments as either acceptable or not, without accounting for varying degrees of plausibility or strength.

While the extension-based semantics have proven effective in many applications, their binary nature limits their ability to provide detailed insights into the relative strength or plausibility of arguments. To address this limitation, ranking semantics have been developed [ABN13, PLZL14, TCR18, BT22]. Unlike traditional extension-based semantics, ranking semantics assign a plausibility rank to each argument, enabling a more nuanced analysis of argumentation frameworks. This approach is particularly valuable in scenarios where a fine-grained assessment of arguments is necessary, such as prioritizing competing claims or evaluating argument strength within complex systems.

This bachelor thesis focuses on conducting an empirical correlation analysis of selected ranking semantics. The research involves implementing selected ranking semantics and comparing their outputs using quantitative measures, such as Kendall's Tau coefficient [Ken38] and Spearman's Rho coefficient [Spe04], on datasets derived from the International Competition on Computational Models of Argumentation (ICCMA) [1]. The study aims to provide valuable insights into the comparative behavior of ranking semantics, supporting more informed decisions about their application in argumentation-based reasoning tasks.

Our results show that correlations between ranking semantics vary significantly depending on the structural properties of the argumentation framework. In acyclic frame-

[1] ICCMA Website: `http://argumentationcompetition.org`

works, most semantics produce highly similar rankings, while in cyclic and dense frameworks the agreement drops substantially. Among the tested approaches, the probabilistic and discussion-based semantics often align most closely, whereas serialisability-based semantics diverge more strongly in dense settings.

The remainder of this thesis is structured as follows. Section 2 introduces the theoretical background and formal definitions. Section 3 presents the datasets, experimental setup, and evaluation metrics. Section 4 describes the implementation details for each ranking semantics. Section 6 reports the empirical findings, and Section 7 discusses their implications. Finally, Section 8 summarizes the findings and outlines directions for future research.

2 Preliminaries

In many fields, such as legal reasoning, artificial intelligence, and decision-making, the ability to evaluate conflicting arguments systematically is crucial. However, real-world argumentative scenarios often involve complex interactions and dependencies between arguments, making informal reasoning insufficient. This need for a formal and systematic approach to analyzing argumentation led to the development of Dung's abstract argumentation framework [Dun95], which has since become a cornerstone in the study of formal argumentation.

2.1 Dung's abstract argumentation framework

Dung's work on argumentation frameworks has significantly shaped the study of formal argumentation, providing a foundational approach to understanding and analyzing the acceptability of arguments. Published in 1995, Dung's theory introduces a highly abstract yet versatile framework that models arguments and their interactions through directed graphs [Dun95]. The central objective of this framework is to formalize the notion of argument acceptability, a concept critical for resolving conflicts and supporting reasoning in fields such as artificial intelligence, law, and decision-making. By focusing on the relationships between arguments, rather than their internal structure, Dung's approach enables a general analysis applicable across diverse contexts.

Definition 2.1.1 (Argumentation Framework (AF)) *An argumentation framework (AF) is formally defined as a pair $AF = (A, R)$, where:*

- *A is a finite set of arguments, and*
- *$R \subseteq A \times A$ is a binary relation representing attacks between arguments.*

In this framework, arguments are treated as abstract entities, and the relation R encodes the directed edges in a graph, where $(a, b) \in R$ signifies that argument a attacks

argument b. This abstraction allows the framework to generalize over diverse application domains, focusing solely on the structure of argumentation rather than the internal content of arguments. The power of this model lies in its simplicity and its ability to capture the essence of argumentative interactions through such minimal components.

Example 2.1.1.1 *Consider a debate about choosing a mode of transportation:*

- ***Argument*** a***:*** *"Taking the car is faster than public transport."*
- ***Argument*** b***:*** *"Using the car is bad for the environment."*

These arguments can be represented in an abstract argumentation framework, where:

- a *attacks* b*: The argument about speed challenges the environmental concerns by prioritizing time.*
- b *attacks* a*: The environmental concerns question whether speed should be the primary consideration.*

This forms a straightforward framework with two arguments attacking each other. It can be formalized as $AF = (A, R)$*, where:*

- $A = \{a, b\}$*, the set of arguments, and*
- $R = \{(a, b), (b, a)\}$*, the attack relation.*

This interaction is visualized in Figure 1, where nodes represent arguments and edges depict the attack relations between them.

Figure 1: Graph representation of the argumentation framework.

Definition 2.1.2 (Defense) *Given an argumentation framework* $AF = (A, R)$*, a set of arguments* $S \subseteq A$ *defends an argument* $a \in A$ *if and only if for every argument* $c \in A$ *such that* c *attacks* a*, there exists an argument* $b \in S$ *such that* b *attacks* c*.*

The concept of defense is crucial in evaluating argumentation frameworks, as it helps determine which arguments can withstand attacks through the support of other arguments. This forms the basis for many semantics that assess argument acceptability. As noted by Delobelle [Del17], the principle of defense can be intuitively captured by the adage: "my enemy's enemy is my friend", emphasizing the strategic nature of argumentative interactions.

Definition 2.1.3 (Attack Sequence) *Given an argumentation framework* $AF = (A, R)$*, an attack sequence for an argument* $a \in A$ *is a finite sequence of arguments* $(a_1, a_2, \ldots, a_n)$ *such that:*

- $a_1 = a$,
- *For all i where $2 \leq i \leq n$, $(a_i, a_{i-1}) \in R$.*

The length of an attack sequence is defined as the number of arguments in the sequence, i.e., n.

Paths in an argumentation framework help us analyze the influence of arguments over others, whether directly or indirectly. By examining the number of attack steps along a path, we can determine whether an argument ultimately acts as an attacker or a defender, reinforcing the dynamic interplay of argumentation.

Definition 2.1.4 (Attacker and Defender Classification) *Given an argumentation framework $AF = (A, R)$ and a path $(a_0, a_1, \ldots, a_n)$ from a_0 to a_n, we classify a_0 as follows:*

- *If n is 1, a_0 is a direct attacker of a_n.*
- *If n is 2, a_0 is a direct defender of a_n.*
- *If n is odd, a_0 is an indirect attacker of a_n.*
- *If n is even, a_0 is a indirect defender of a_n.*

The classification follows from the alternating nature of attack sequences in argumentation frameworks, where every successive attack reverses the argumentative stance.

Definition 2.1.5 (Rooted Path and Root Arguments) *A path $(a_0, a_1, \ldots, a_n)$ in an argumentation framework $AF = (A, R)$ is called a rooted path if a_0 is **not attacked** by any argument in A, i.e.,*

$$\forall x \in A, (x, a_0) \notin R.$$

The argument a_0 is referred to as a root argument of the path.
Furthermore:

- *If n is odd, a_0 is called an attack root, meaning it ultimately acts as an attacker of a_n.*
- *If n is even, a_0 is called a defense root, meaning it ultimately acts as a defender of a_n.*

To evaluate arguments systematically within Dung's abstract argumentation framework, it is necessary to determine which arguments can be considered collectively acceptable. This process involves identifying subsets of arguments that satisfy specific criteria related to conflict resolution and defense. Extension-based semantics were developed to formalize these notions, providing a structured way to classify arguments based on their interactions.

2.2 Extension-based semantics

Dung's work introduced several semantics for defining acceptable subsets of arguments, known as extension, within an argumentation framework:

Definition 2.2.1 (Conflict-freeness) *A set of argument $S \subseteq A$ is conflict-free if no argument in S attacks another argument in S.*

This property ensures coherence among the selected arguments.

Definition 2.2.2 (Admissibility) *A set $S \subseteq A$ is admissible if:*

- *S is conflict-free, and*
- *For every $a \in S$, if $b \in A$ attacks a, then there exists $c \in S$ such that c attacks b.*

An admissible set builds on conflict-freeness by requiring that each argument in S is defended by S against all attacks.

Example 2.2.2.1 (Admissibility vs. conflict-freeness) *Consider an argumentation framework $AF = (A, R)$, where:*

- *$A = \{a, b, c\}$ is the set of arguments, and*
- *$R = \{(a, c), (c, b)\}$ is the attack relation.*

This framework is illustrated in Figure 2.

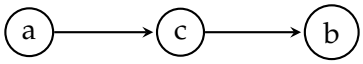

Figure 2: Argumentation framework for the Admissibility vs. conflict-freeness example.

The conflict-free sets of arguments in this framework are:

$$\emptyset, \{a\}, \{b\}, \{c\}, \{a, b\}.$$

Let us evaluate their admissibility:

- *$\{\}$: This set is conflict-free and trivially admissible.*
- *$\{a\}$: This set is conflict-free and admissible because a is not attacked by any argument.*
- *$\{b\}$: This set is conflict-free but not admissible because b is attacked by c and is not able to defend itself.*
- *$\{c\}$: This set is conflict-free but not admissible because c is attacked by a and is not able to defend itself.*

- $\{a, b\}$: *This set is conflict-free (since a and b do not attack each other) and admissible because while b is attacked by c, a defends b by attacking c.*

Thus, the admissible sets are:

$$\{\}, \{a\}, \{a, b\}.$$

While conflict-freeness ensures coherence among selected arguments and admissibility builds on this by requiring defense against external attacks, these properties serve as the foundation for defining acceptable sets of arguments within an argumentation framework. Dung formalized this idea further by introducing different types of extensions, each offering a unique perspective on argument acceptability. These include the **preferred extension**, which maximizes admissibility; the **stable extension**, which emphasizes external dominance; and the **grounded extension**, which provides a conservative baseline of acceptability.

Definition 2.2.3 (Preferred extension) *A set of arguments $S \subseteq A$ is a preferred extension if:*

- *S is admissible, and*
- *S is maximal with respect to set inclusion, meaning there is no admissible set T such that $S \subset T$.*

Preferred extensions are significant because they represent the broadest perspective of defensible arguments within an argumentation framework. By including all admissible arguments that can coexist without conflict, preferred extensions aim to capture comprehensive sets of arguments that are collectively acceptable. This makes them particularly useful in scenarios where maximal defensibility is desirable, such as in legal reasoning or collaborative decision-making.

Definition 2.2.4 (Stable extension) *A set of arguments $S \subseteq A$ is a stable extension if:*

- *S is conflict-free, and*
- *Every argument not in S is attacked by at least one argument in S.*

Stable extensions emphasize the idea of dominance, ensuring that all arguments outside the extension are countered. This property makes stable extensions particularly appealing in adversarial settings, such as debates or negotiations, where the goal is to eliminate opposition.

However, stable extensions may not always exist in an argumentation framework [Dun95], which is a limitation that motivates the exploration of alternative semantics. The following example illustrates a scenario where no stable extension can be found.

Example 2.2.4.1 (Framework without stable extensions) *Consider a simple cyclic argumentation framework $AF = (A, R)$, where $A = \{a, b, c\}$ and $R = \{(a, b), (b, c), (c, a)\}$, as illustrated in Figure 3.*

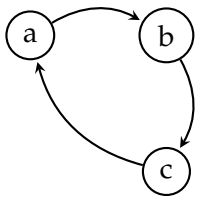

Figure 3: A simple 3-cycle argumentation framework with no stable extension.

In this framework, no stable extension exists. To be stable, a set must be conflict-free and must attack every argument outside the set. The conflict-free sets are $\{\}$, $\{a\}$, $\{b\}$, and $\{c\}$.

- *The empty set $\{\}$ does not attack a, b, or c.*
- *The set $\{a\}$ does not attack b.*
- *The set $\{b\}$ does not attack c.*
- *The set $\{c\}$ does not attack a.*

Since none of the conflict-free sets attack all arguments outside of them, there is no stable extension for this framework.

Definition 2.2.5 (Characteristic function) *Let $AF = (A, R)$ be an argumentation framework. The characteristic function $F_{AF} : 2^A \to 2^A$ is defined as:*

$$F_{AF}(S) = \{a \in A \mid S \text{ defends } a\}$$

Definition 2.2.6 (Grounded extension [Dun95]) *The grounded extension of an argumentation framework AF, denoted GE_{AF}, is the least fixed point of its characteristic function F_{AF}.*

The grounded extension always exists and is unique for any argumentation framework [Dun95].

The characteristic function F_{AF} plays a central role in abstract argumentation by using the concept of defense. It evaluates the interactions between arguments and identifies which arguments can withstand attacks when supported by a given set S. This function serves as the foundation for defining various extensions, including the grounded extension. The grounded extension, as the smallest fixed point of F_{AF}, provides a conservative and uncontroversial set of acceptable arguments. By iteratively applying F_{AF}, it ensures that only arguments that are either unchallenged or successfully defended are included. This makes the grounded extension particularly valuable in applications requiring a cautious or minimal approach to argumentation, such as legal reasoning or safety-critical systems. Its guaranteed existence and uniqueness further enhance its utility as a baseline for reasoning within argumentation frameworks.

2.3 Ranking semantics

While extension-based semantics, such as preferred, stable, and grounded extensions, provide structured ways to evaluate abstract argumentation frameworks, they are inherently binary in nature. These approaches classify arguments as either acceptable or unacceptable, offering no further differentiation within the set of acceptable arguments or between those considered unacceptable. This limitation is particularly problematic in complex argumentation scenarios, where arguments may vary significantly in their plausibility, strength, or degree of acceptability [Dun95]. For example, two arguments might both belong to a preferred extension, but one may be much stronger than the other based on additional contextual factors.

Ranking semantics were developed to address these shortcomings by providing a more nuanced evaluation of arguments. Instead of categorising arguments into binary outcomes, ranking semantics assign a relative degree of plausibility or strength to each argument [Del17]. This allows for finer-grained analyses, enabling comparisons between all arguments within an argumentation framework. As a result, ranking semantics are particularly useful in applications where detailed prioritization or weighting of arguments is necessary, such as decision support systems, multi-agent negotiations, and legal reasoning.

Definition 2.3.1 (Ranking-based semantics [ABN13]) *A **ranking-based semantics** S for an abstract argumentation framework $AF = (A, R)$ is a function*

$$S : (A, R) \mapsto \succeq^S_{AF}$$

that assigns a preorder $\succeq^S_{AF}$ over the set of arguments A, where for every $a, b \in A$, the following relations hold:

- *$a \succ^S_{AF} b$ if argument a is strictly more acceptable than argument b.*
- *$a \succeq^S_{AF} b$ if argument a is at least as acceptable as argument b.*
- *$a \simeq^S_{AF} b$ if arguments a and b are equally acceptable.*

These relations are all transitive, meaning that if $a \succeq^S_{AF} b$ and $b \succeq^S_{AF} c$, then $a \succeq^S_{AF} c$ for all $a, b, c \in A$.

The definition above establishes the fundamental structure of ranking-based semantics in abstract argumentation. Unlike extension-based semantics, which classify arguments into accepted or rejected sets, ranking-based semantics introduce a finer-grained evaluation by ordering arguments according to their relative acceptability.

With this foundational understanding in place, we now introduce our example argumentation framework AF_{ex}, illustrated in Figure 4, which will serve as the basis for illustrating the application of different ranking-based semantics.

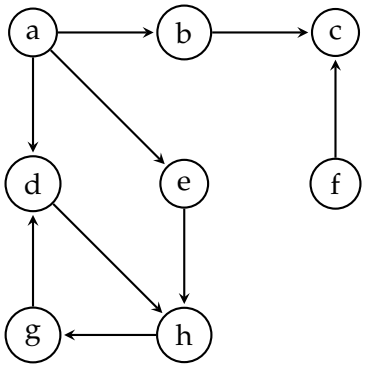

Figure 4: Example argumentation framework AF_{ex}

2.3.1 Discussion-based ranking semantics

The **Discussion-based ranking semantics** (Dbs) [ABN13] assigns a ranking to arguments based on the number of attacks and defense paths leading to them within an argumentation framework. It relies on the concepts of attack sequences (Definition 2.1.3) and discussion counts defined below.

Definition 2.3.2 (Discussion Count) *Let $AF = (A, R)$ be an argumentation framework, $a \in A$, and $i \in \mathbb{N} \setminus \{0\}$. We define the discussion count at step i for argument a as:*

$$Dis_i(a) = \begin{cases} -N & \text{if } i \text{ is even} \\ N & \text{if } i \text{ is odd} \end{cases}$$

where N is the number of attack sequences for a in AF of length i. The full discussion count for a is the vector $Dis(a) = \langle Dis_1(a), Dis_2(a), \dots \rangle$.

Definition 2.3.3 (Discussion-based ranking semantics) *The **Discussion-based ranking semantics (Dbs)** associates to any argumentation framework $AF = (A, R)$ a ranking $\succeq_{AF}^{Dbs}$ on A such that $\forall a, b \in A$:*

$$a \succeq_{AF}^{Dbs} b \text{ if and only if } Dis(b) \succeq_{lex} Dis(a)$$

where $\succeq_{lex}$ is the lexicographical order. That is, $a \succeq_{AF}^{Dbs} b$ iff either $Dis(a) = Dis(b)$ or there exists $i \geq 1$ such that $Dis_i(a) < Dis_i(b)$ and for all $j < i$, $Dis_j(a) = Dis_j(b)$.

Example 2.3.3.1 (Dbs calculation for AF_{ex}) *Let us calculate the ranking for the argumentation framework AF_{ex} (illustrated in Figure 4). We compute the Discussion count vector $Dis(x) = \langle Dis_1(x), Dis_2(x), \dots \rangle$ for each argument x. The presence of the cycle means the vectors for d, g, h will be infinite, so we compute the first few steps for comparison.*

- $Dis(a) = \langle 0, 0, 0, \dots \rangle$

- $Dis(b) = \langle 1, 0, 0, \dots \rangle$
- $Dis(c) = \langle 2, -1, 0, \dots \rangle$
- $Dis(d) = \langle 2, -1, 2, -3, 1, \dots \rangle$
- $Dis(e) = \langle 1, 0, 0, \dots \rangle$
- $Dis(f) = \langle 0, 0, 0, \dots \rangle$
- $Dis(g) = \langle 1, -2, 3, -1, 2, \dots \rangle$
- $Dis(h) = \langle 2, -3, 1, -2, 3, \dots \rangle$

We compare these vectors using lexicographical order, where smaller values at the first differing index indicate a higher (better) rank.

- ***Rank 1:*** *Arguments a and f have the vector $\langle 0, 0, \dots \rangle$, which is the smallest. Thus, $a \simeq_{AF}^{Dbs} f$.*
- ***Rank 2:*** *Next, we compare arguments whose vectors start with 1: g and (b, e).*
  - *$Dis(g) = \langle 1, -2, \dots \rangle$*
  - *$Dis(b, e) = \langle 1, 0, \dots \rangle$*
  - *At the second index, $-2 < 0$, so the vector for g is smaller. This means g is ranked higher.*
- ***Rank 3:*** *The arguments (b, e) with vector $\langle 1, 0, \dots \rangle$ are next.*
- ***Rank 4:*** *Now we compare the arguments whose vectors start with 2: h, c, and d.*
  - *$Dis(h) = \langle 2, -3, \dots \rangle$*
  - *$Dis(c) = \langle 2, -1, \dots \rangle$*
  - *$Dis(d) = \langle 2, -1, \dots \rangle$*
  - *At the second index, -3 is the smallest value, so h is ranked highest within this group.*
- ***Rank 5:*** *We compare the remaining arguments, c and d.*
  - *$Dis(c) = \langle 2, -1, 0, \dots \rangle$*
  - *$Dis(d) = \langle 2, -1, 2, \dots \rangle$*
  - *Their vectors match until the third index, where $0 < 2$. The vector for c is smaller, so c is ranked higher.*
- ***Rank 6:*** *The last remaining argument is d.*

The final ranking, derived from this step-by-step comparison, is:

$$a \simeq_{AF}^{Dbs} f \succ_{AF}^{Dbs} g \succ_{AF}^{Dbs} b \simeq_{AF}^{Dbs} e \succ_{AF}^{Dbs} h \succ_{AF}^{Dbs} c \succ_{AF}^{Dbs} d$$

2.3.2 Categoriser-based ranking semantics

The **Categoriser-based semantics** (Cat) [PLZL14] evaluates the acceptability of arguments based on their relationships within the argumentation graph. It assigns a strength value to each argument, considering both the support it receives from other arguments and the attacks it faces. This is achieved by defining a system of equations for the strength values.

Definition 2.3.4 (Categoriser function) *Given an argumentation framework $AF = (A, R)$, the categoriser function $Cat : A \to [0, 1]$ assigns a value to each argument $a \in A$ defined by the following system of equations:*

$$Cat(a) = \begin{cases} 1 & \text{if } R^-(a) = \emptyset \\ \frac{1}{1+\sum_{b \in R^-(a)} Cat(b)} & \text{otherwise} \end{cases}$$

where $R^-(a)$ denotes the set of arguments that attack a.

The function assigns a value of 1 to an argument with no attackers, representing maximal strength. If an argument is attacked, its strength is calculated as the inverse of 1 plus the sum of the strength values of its attackers. This definition captures the principle that an argument is strengthened by having weak attackers and weakened by having strong attackers. It's important to note that the Categoriser function is **well-defined for all argumentation frameworks**, meaning a unique solution for the strength values exists. This is guaranteed as the function is a **contraction mapping**, ensuring convergence to a unique fixed point [PLZL14].

Definition 2.3.5 (Categoriser-based ranking semantics) *The Categoriser-based ranking semantics associates to any argumentation framework $AF = (A, R)$ a ranking $\succeq_{AF}^{Cat}$ on A such that for all $a, b \in A$:*

$$a \succeq_{AF}^{Cat} b \text{ if and only if } Cat(a) \geq Cat(b).$$

Example 2.3.5.1 (Cat calculation for AF_{ex}) *Consider the argumentation framework AF_{ex} from Figure 4. The Cat semantics calculates the strength of each argument as follows:*

- $Cat(a) = 1$ *(no attackers)*
- $Cat(b) = \frac{1}{1+Cat(a)} = \frac{1}{1+1} = 0.5$
- $Cat(c) = \frac{1}{1+Cat(b)+Cat(f)}$
- $Cat(d) = \frac{1}{1+Cat(a)+Cat(g)}$
- $Cat(e) = \frac{1}{1+Cat(a)} = 0.5$
- $Cat(f) = 1$

- $Cat(g) = \frac{1}{1+Cat(h)}$
- $Cat(h) = \frac{1}{1+Cat(d)+Cat(e)}$

To solve for $Cat(c)$, $Cat(d)$, $Cat(g)$ and $Cat(h)$, we must solve a system of equations. By substituting the known values and simplifying, we get:

$$Cat(d) = \frac{1}{2 + Cat(g)}$$
$$Cat(g) = \frac{1}{1 + Cat(h)}$$
$$Cat(h) = \frac{1}{1.5 + Cat(d)}$$

Solving this system yields the approximate values: $Cat(d) \approx 0.38$, $Cat(g) \approx 0.65$, and $Cat(h) \approx 0.53$. And $Cat(c) = \frac{1}{1+0.5+1} = 0.4$. The final ranking is:

$$a \simeq_{AF}^{Cat} f \succ_{AF}^{Cat} g \succ_{AF}^{Cat} h \succ_{AF}^{Cat} b \simeq_{AF}^{Cat} e \succ_{AF}^{Cat} c \succ_{AF}^{Cat} d$$

2.3.3 Probabilistic ranking semantics

The **Probabilistic ranking semantics** (Pro) [TCR18] relies on probabilistic argumentation frameworks where each argument is assigned a probability of being present or absent. The overall acceptability of an argument is then determined by evaluating its likelihood of being accepted across various possible subgraphs of the framework, each weighted by its probability of occurrence.

Definition 2.3.6 (Probabilistic argumentation framework) *A **probabilistic argumentation framework (PAF)** is a triple $PAF = (A, R, P)$ where:*

- *(A, R) is an abstract argumentation framework, and*
- *$P : A \to [0, 1]$ is a function that assigns a probability to each argument $a \in A$, representing the probability that a is present in the argumentation framework.*

For the purpose of ranking, we assume a uniform probability p is assigned to all arguments. Assuming the presence of each argument is probabilistically independent, this induces a probability distribution over all possible subsets of arguments (subgraphs).

Definition 2.3.7 (Subgraph and its probability) *Given a set of arguments $X \subseteq A$, the **induced subgraph** is $AF_X = (X, R \cap (X \times X))$. The probability of any specific subgraph AF_X occurring is given by:*

$$P(X) = \prod_{a \in X} P(a) \cdot \prod_{a \notin X} (1 - P(a))$$

When using a uniform probability p, this simplifies to:

$$P(X) = p^{|X|} \cdot (1 - p)^{|A \setminus X|}$$

Example 2.3.7.1 *Consider a simple framework $AF = (\{a, b\}, \{(b, a)\})$ and let $p = 0.5$. There are four possible subgraphs:*

1. *$AF_\emptyset$ (empty set): $P(\emptyset) = (0.5)^0 \cdot (0.5)^2 = 0.25$. The grounded extension is $\emptyset$.*
2. *$AF_{\{a\}}$ (a is present): $P(\{a\}) = (0.5)^1 \cdot (0.5)^1 = 0.25$. The grounded extension is $\{a\}$.*
3. *$AF_{\{b\}}$ (b is present): $P(\{b\}) = (0.5)^1 \cdot (0.5)^1 = 0.25$. The grounded extension is $\{b\}$.*
4. *$AF_{\{a,b\}}$ (both present): $P(\{a, b\}) = (0.5)^2 \cdot (0.5)^0 = 0.25$. The grounded extension is $\{b\}$.*

The probability of acceptance for an argument is then calculated by summing the probabilities of all subgraphs in which that argument is accepted.

Definition 2.3.8 (Probability of acceptance) *Let σ be a classical semantics and $\circ$ be an inference mode. The probability of acceptance for an argument a in a PAF is the sum of the probabilities $P(X)$ for all subgraphs AF_X where a is accepted:*

$$P^{PAF}_{\circ,\sigma}(a) = \sum_{a \in X \subseteq A, AF_X \models^\circ_\sigma a} P(X)$$

Continuing our example, the acceptance probabilities for a and b are:

- **Argument** a: Is accepted only in $AF_{\{a\}}$, so its acceptance probability is 0.25.
- **Argument** b: Is accepted in $AF_{\{b\}}$ and $AF_{\{a,b\}}$, so its acceptance probability is $0.25 + 0.25 = 0.50$.

The probabilistic ranking semantics formalizes this idea.

Definition 2.3.9 (Probabilistic ranking semantics) *Let $AF = (A, R)$ be an argumentation framework, $p \in [0, 1]$ be a uniform probability assigned to all arguments, $\sigma \in \{co,gr,pr\}$ be a classical semantics (complete, grounded or preferred), and $\circ \in \{s,c\}$ be an inference mode (skeptical or credulous). The probabilistic ranking semantics $G^{\sigma,\circ,p}$ is defined for every argument $a \in A$ as:*

$$G^{\sigma,\circ,p}_{AF}(a) = P^{AF[p]}_{\circ,\sigma}(a)$$

where $P^{AF[p]}_{\circ,\sigma}(a)$ denotes the probability of acceptance of argument a in the probabilistic framework $AF[p]$. The ranking $\geq^{Pro}_{AF}$ is then derived such that for any two arguments $a, b \in A$:

$$a \geq^{Pro}_{AF} b \text{ if and only if } G^{\sigma,\circ,p}_{AF}(a) \geq G^{\sigma,\circ,p}_{AF}(b)$$

A higher score (probability of acceptance) indicates a greater level of acceptability, suggesting that the argument is robustly accepted across many varying topological scenarios. Conversely, a low score implies the argument is frequently rejected, even when some of its attackers might be absent. For the purpose of ranking, a common choice for p is 0.5, and often grounded credulous semantics (gr, c) is used for calculation, though the framework is flexible.

Example 2.3.9.1 (Pro calculation for AF_{ex}) *Consider the argumentation framework AF_{ex} from Figure 4. To calculate the probabilistic ranking semantics, let's assume a uniform probability $p = 0.5$ for the presence of each argument and use the grounded credulous (gr, c) semantics for acceptance within subgraphs. With 8 arguments in AF_{ex}, there are $2^8 = 256$ possible subgraphs. The probability of any specific subgraph occurring (given that each argument's presence is independent and has a probability of 0.5) is $(0.5)^8 = 1/256 = 0.00390625$.*

The calculation involves the following conceptual steps:

1. *Enumerate all 2^8 subgraphs of AF_{ex}. (The detailed enumeration and grounded extension for each non-empty subgraph can be found in Appendix 8.)*
2. *For each subgraph, determine the set of accepted arguments under the grounded semantics.*
3. *For each argument $x \in \{a, b, c, d, e, f, g, h\}$, count the number of subgraphs in which x is accepted. Let this be $count(x)$.*
4. *The probability score for each argument x is then $G^{gr,c,0.5}_{AF_{ex}}(x) = count(x) \times (0.5)^8$.*

Based on the analysis of all 256 subgraphs, the acceptance counts for each argument are as follows:

- *Argument a: accepted in 128 subgraphs.*
- *Argument b: accepted in 64 subgraphs.*
- *Argument c: accepted in 48 subgraphs.*
- *Argument d: accepted in 32 subgraphs.*
- *Argument e: accepted in 64 subgraphs.*
- *Argument f: accepted in 128 subgraphs.*
- *Argument g: accepted in 80 subgraphs.*
- *Argument h: accepted in 80 subgraphs.*

Using these counts, the probability scores $G^{gr,c,0.5}_{AF_{ex}}(x)$ are:

- $G(a) = 128 \times (1/256) = 0.5$
- $G(b) = 64 \times (1/256) = 0.25$
- $G(c) = 48 \times (1/256) = 0.1875$
- $G(d) = 32 \times (1/256) = 0.125$
- $G(e) = 64 \times (1/256) = 0.25$
- $G(f) = 128 \times (1/256) = 0.5$

- $G(g) = 80 \times (1/256) = 0.3125$
- $G(h) = 80 \times (1/256) = 0.3125$

Ranking arguments based on these scores (higher score means more acceptable), we get the following ranking for AF_{ex} under the Probabilistic semantics with $p = 0.5$ and grounded credulous acceptance:

$$a \simeq^{Pro}_{AF_{ex}} f \succ^{Pro}_{AF_{ex}} g \simeq^{Pro}_{AF_{ex}} h \succ^{Pro}_{AF_{ex}} b \simeq^{Pro}_{AF_{ex}} e \succ^{Pro}_{AF_{ex}} c \succ^{Pro}_{AF_{ex}} d$$

For any given classical semantics σ (eg. admissible, complete, grounded, ideal, preferred and stable), we derive a corresponding probabilistic ranking semantic, which we denote as Pro_σ. For example **probabilistic-grounded (Pro_g)** refers to the ranking derived from using the grounded semantics ($\sigma = gr$) within the probabilistic framework. This approach enables a detailed empirical comparison not only between the foundational ranking methods (Cat, Dbs, Ser) but also among the different flavors of the probabilistic approach itself.

2.3.4 Serialisability-based ranking semantics

The **Serialisability-based ranking semantics (Ser)** [BT22] ranks arguments based on the "effort" required to include them in an admissible set. This effort is measured by the minimum number of steps in a constructive process where admissible sets are built by iteratively adding "initial sets" from a progressively simplified argumentation framework. The fewer steps an argument requires to be included as part of such a final accepted group, the higher its rank.

The formalisation of this ranking approach relies on the following key concepts:

Definition 2.3.10 (Initial Set [XC18]) *For an argumentation framework $AF = (A, R)$, a set $S \subseteq A$ with $S \neq \emptyset$ is called an **initial set** if S is admissible and there is no admissible set $S' \subset S$ with $S' \neq \emptyset$. Let $IS(AF)$ denote the set of all initial sets of AF.*

Initial sets are described as the "smallest units within an admissible set, which still maintain admissibility". They represent foundational, self-defending groups of arguments. An argument that is part of an initial set of the original framework can be considered to resolve its acceptance conflicts largely "by itself" within that minimal group.

Definition 2.3.11 (Reduct [BBU20]) *For an argumentation framework $AF = (A, R)$ and a set $S \subseteq A$, the **reduct** of S with respect to AF is $AF^S = AF|_{A\setminus(S\cup S^+)}$*

The reduct AF^S represents the remaining argumentation scenario after the arguments in set S (and those they attack, S^+) are considered accepted and thus removed from the framework. It is in this simplified framework that subsequent initial sets are identified.

Definition 2.3.12 (Serialisation Sequence [BT22]) *A **serialisation sequence** for an argumentation framework $AF = (A, R)$ is a sequence $\mathcal{F} = (S_1, ..., S_n$ with $S_1 \in IS(AF)$ and for each $2 \leq i \leq n$ we have $S_i \in IS(AF^{S_1 \cup ... \cup S_{i-1}})$.*

A serialisation sequence therefore outlines a constructive method for forming a larger admissible set $E = S_1 \cup ... \cup S_n$. The existence of such a sequence can characterize the admissibility of the set E.

The rank is determined by the serialisation index:

Definition 2.3.13 (Serialisation Index [BT22]) *For an argumentation framework $AF = (A, R)$ and an argument $a \in A$, the serialisation index $ser_{AF}(a)$ is defined as:*

$$ser_{AF}(a) = \min\{n \mid (S_1, \ldots, S_n) \text{ is a serialisation sequence and } a \in S_n\}$$

with $\min \emptyset = \infty$. An argument a having $ser_{AF}(a) = n$ means that n is the shortest sequence length where a is part of the n-th (last added) initial set. A lower index means a better rank. Arguments that cannot be part of such an S_n (i.e., are not credulously accepted via this constructive process) get an index of ∞.

Definition 2.3.14 (Serialisability-based ranking semantics [BT22]) *The **Serialisability-based ranking semantics** (Ser) associates to any argumentation framework $AF = (A, R)$ a ranking $\succeq^{Ser}_{AF}$ on A such that for all $a, b \in A$:*

$$a \succeq^{Ser}_{AF} b \text{ if and only if } ser_{AF}(a) \leq ser_{AF}(b).$$

Example 2.3.14.1 (Ser calculation for AF_{ex}) *Consider AF_{ex} from Figure 4.*

1. ***Arguments with** $ser_{AF_{ex}}(x) = 1$**:** These are arguments that are part of an initial set of AF_{ex} itself.*
   - *$\{a\}$ is an initial set (unattacked, admissible, minimal non-empty). So, $ser_{AF_{ex}}(a) = 1$.*
   - *$\{f\}$ is an initial set (unattacked, admissible, minimal non-empty). So, $ser_{AF_{ex}}(f) = 1$.*

   No other single argument forms an initial set.

2. ***Arguments with** $ser_{AF_{ex}}(x) = 2$**:** These arguments x must be in an initial set S_2 of a reduct AF^{S_1}, where S_1 is an initial set from step 1.*
   - *Let $S_1 = \{a\}$. The reduct is $AF^{\{a\}} = AF|_{A \setminus (\{a\} \cup \{b,d,e\})} = AF|_{\{c,f,g,h\}}$. The attacks within this reduct are (f, c) and (h, g).*
   - *In $AF^{\{a\}}$, h is unattacked. Thus, $\{h\}$ is an initial set of $AF^{\{a\}}$.*
   - *Therefore, via the serialisation sequence $(\{a\}, \{h\})$, we find $h \in S_2$. So, $ser_{AF_{ex}}(h) = 2$.*
   - *(Note: $\{f\}$ is also an initial set in $AF^{\{a\}}$, but $ser(f) = 1$ is already minimal.)*

3. ***Arguments with*** $ser_{AF_{ex}}(x) = \infty$***:*** *We evaluate why other arguments cannot achieve a finite, low serialisation index.*
    - *Arguments* b, d, e*: These are directly attacked by* a*. Since* $ser(a) = 1$*, if* $S_1 = \{a\}$ *is the first step in a serialisation,* b, d, e *become part of* S_1^+ *(attacked by* S_1*) and are removed in the reduct* $AF^{\{a\}}$*. They cannot then appear in* S_2 *or any subsequent* S_n *of such a sequence. No other choice for* S_1 *(like* $\{f\}$*) makes them part of an* S_n*. Thus,* $ser_{AF_{ex}}(b) = \infty$*,* $ser_{AF_{ex}}(d) = \infty$*,* $ser_{AF_{ex}}(e) = \infty$*.*
    - *Argument* c*: Attacked by* f *(where* $ser(f) = 1$*) and by* b *(where* $ser(b) = \infty$*). If* $S_1 = \{f\}$*, then* $c \in S_1^+$ *and is removed. Thus,* $ser_{AF_{ex}}(c) = \infty$*.*
    - *Argument* g*: Attacked by* h*. We found* $ser(h) = 2$ *via sequence* $S_1 = \{a\}, S_2 = \{h\}$*. In this path, after* S_1 *and* S_2 *are accepted,* g *is in* S_2^+ *(attacked by* S_2*) and thus removed in the next reduct* $AF^{\{a,h\}}$*. It cannot appear in an* S_3 *of this sequence. Exploring other sequences does not yield a finite* n *where* $g \in S_n$*. Thus,* $ser_{AF_{ex}}(g) = \infty$*.*

The serialisation indices are:

- $ser_{AF_{ex}}(a) = 1$
- $ser_{AF_{ex}}(f) = 1$
- $ser_{AF_{ex}}(h) = 2$
- $ser_{AF_{ex}}(b) = \infty$
- $ser_{AF_{ex}}(c) = \infty$
- $ser_{AF_{ex}}(d) = \infty$
- $ser_{AF_{ex}}(e) = \infty$
- $ser_{AF_{ex}}(g) = \infty$

The final ranking (lower index is better) is:

$$a \simeq^{Ser}_{AF_{ex}} f \succ^{Ser}_{AF_{ex}} h \succ^{Ser}_{AF_{ex}} b \simeq^{Ser}_{AF_{ex}} c \simeq^{Ser}_{AF_{ex}} d \simeq^{Ser}_{AF_{ex}} e \simeq^{Ser}_{AF_{ex}} g$$

3 Methodology

This chapter details the systematic methodology employed to conduct the empirical correlation analysis of ranking-based semantics for abstract argumentation frameworks. The primary objective is to create a clear, reproducible, and robust experimental design that allows for a quantitative comparison of the selected semantics. This involves the implementation of the semantics, the selection of appropriate datasets, the definition of the experimental procedure, and the choice of statistical metrics for measuring correlation.

3.1 Research questions

The methodology is specifically designed to address the following research questions that arise from the thesis objectives:

1. **RQ1: Quantitative Correlation:** To what degree do the argument rankings produced by Discussion-based (Dbs), Categoriser-based (Cat), Serialisability-based (Ser), and the various instantiations of probabilistic semantics (Pro_a, Pro_c,...) correlate with one another when applied to a diverse set of standard benchmark argumentation frameworks?
2. **RQ2: Influence of Structural Properties:** Are there identifiable structural properties of argumentation frameworks (e.g., the presence of cycles, density of attacks) that systematically influence the level of agreement or disagreement between specific pairs of semantics?
3. **RQ3: Conceptual Interpretation:** What can the empirical correlation scores reveal about the underlying conceptual similarities and divergences of these semantics?

3.2 Datasets

To ensure the results are both generalizable and computationally feasible, this study utilizes two distinct collections of argumentation frameworks: a set of established, large-scale benchmarks and a custom-generated set of smaller frameworks.

The first collection consists of benchmark argumentation frameworks from the **International Competition on Computational Models of Argumentation (ICCMA)**. The ICCMA datasets are the de facto standard for empirical evaluation in argumentation, ensuring our results are comparable to other research in the field. For this thesis, the specific dataset used was the main track from the ICCMA 2023.

To facilitate a complete pairwise comparison across all semantics, including the computationally intensive ones, a second collection of smaller frameworks was generated using methods from the **TweetyProject**. This dataset was specifically designed to ensure that the "slow" semantics (such as Serialisability-based and probabilistic-preferred) could complete their calculations within the established timeout, allowing for a comprehensive analysis across all implemented methods on a shared set of frameworks. All frameworks from both collections were parsed directly from the standard `.af` file format.

3.3 Experimental procedure

The empirical analysis follows a precise, automated workflow for each argumentation framework (AF) in the selected dataset. The procedure is as follows:

1. **Input:** An argumentation framework $AF = (A, R)$ is loaded from the benchmark dataset.

2. **Ranking Computation:** For the given AF, the ranking of all arguments A is computed using each of the implemented semantics: (Dbs, Cat, Ser, Pro_a, Pro_c, Pro_g, Pro_i, Pro_p, Pro_s).

3. **Pairwise Comparison:** All unique pairs of semantics are formed for comparison.

4. **Correlation Calculation:** For each pair, the two corresponding argument rankings are compared using standard statistical correlation coefficients. The specific metrics used are defined in Section 3.4.

5. **Data Logging:** The results are systematically stored, recording the name of the AF, the pair of semantics being compared, and the resulting correlation coefficients.

6. **Iteration:** Steps 1-5 are repeated for every argumentation framework in the selected benchmark collection.

3.4 Comparison metrics

To quantitatively compare the outputs of the different ranking semantics, standard statistical measures of rank correlation were employed. The specific metrics used are Kendall's Tau coefficient and Spearman's Rho coefficient.

Definition 3.4.1 (Kendall's Tau coefficient (τ) [Ken38]) *Given a set of n items, the Kendall's Tau coefficient is defined as:*

$$\tau = \frac{N_c - N_d}{\frac{1}{2}n(n-1)}$$

where N_c is the number of concordant pairs and N_d is the number of discordant pairs.

This metric measures the ordinal association between two rankings. A pair of arguments is **concordant** if their relative ordering is the same in both rankings and **discordant** if their relative ordering is different. A value of +1 indicates perfect agreement, -1 indicates perfect disagreement, and 0 indicates independence.

Example 3.4.1.1 (Kendall's Tau Calculation) *Consider two semantics producing the following rankings for arguments {a, b, c, d}:*

- *Ranking 1: $a \succ b \succ c \succ d$*
- *Ranking 2: $a \succ c \succ b \succ d$*

There are $\frac{4\times3}{2} = 6$ unique pairs of arguments: (a,b), (a,c), (a,d), (b,c), (b,d), and (c,d). We check each for concordance:

- *(a,b): Concordant (a is ranked higher than b in both)*
- *(a,c): Concordant (a is ranked higher than c in both)*

- *(a,d): Concordant (a is ranked higher than d in both)*
- *(b,c):* ***Discordant*** *(b > c in Ranking 1, but c > b in Ranking 2)*
- *(b,d): Concordant (b is ranked higher than d in both)*
- *(c,d): Concordant (c is ranked higher than d in both)*

Here, $N_c = 5$ *and* $N_d = 1$*. The calculation is:*

$$\tau = \frac{5-1}{6} = \frac{4}{6} \approx 0.67$$

Definition 3.4.2 (Spearman's Rho coefficient (ρ) [Spe04]) *For a set of* n *items, the Spearman's Rho coefficient is defined as:*

$$\rho = 1 - \frac{6 \sum_{i=1}^{n} d_i^2}{n(n^2-1)}$$

where d_i *is the difference between the ranks of item* i *in the two rankings.*

This metric measures how similarly two rankings order the same set of items based on how high or low each item appears in both lists. If items that are ranked high in one list also tend to be ranked high in the other (and vice versa), the coefficient will be close to +1. If the rankings are in opposite order, the coefficient will be close to -1, and if there is no clear relationship, it will be around 0.

Example 3.4.2.1 (Spearman's Rho Calculation) *Using the same rankings from Example 3.4.1.1, we first assign numerical ranks and find the differences:*

Argument	***Rank 1***	***Rank 2***	***Difference (d_i)***	d_i^2
a	*1*	*1*	*0*	*0*
b	*2*	*3*	*-1*	*1*
c	*3*	*2*	*1*	*1*
d	*4*	*4*	*0*	*0*
Total				$\sum d_i^2 = 2$

With $n = 4$ *and* $\sum d_i^2 = 2$*, the calculation is:*

$$\rho = 1 - \frac{6 \times 2}{4(4^2-1)} = 1 - \frac{12}{4(15)} = 1 - \frac{12}{60} = 1 - 0.2 = 0.8$$

These coefficients were calculated for every pair of semantics (S_1, S_2) on each argumentation framework AF by comparing the rankings $\succeq_{AF}^{S_1}$ and $\succeq_{AF}^{S_2}$.

3.5 Analysis plan

The raw correlation data logged during the experimental procedure were aggregated and analyzed to answer the research questions.

1. **Overall correlation analysis:** For each of the six pairs of semantics, the mean, median, and standard deviation of both Kendall's τ and Spearman's ρ coefficients were calculated across all tested argumentation frameworks. The results were represented in a **correlation matrix** to provide a clear, high-level overview of the general agreement between the semantics.

2. **Sub-group analysis:** To address RQ2, the dataset was partitioned based on key structural properties (e.g., cyclic vs. acyclic). The average correlation scores were computed independently for these subgroups to determine if specific topologies influenced the agreement between semantics.

3. **Interpretation:** Finally, the quantitative findings were interpreted in the context of the theoretical descriptions from Section 2. Consistently high correlations between a pair of semantics suggested a deep conceptual similarity, while low or volatile correlations highlighted fundamental differences in their evaluation principles. This analysis formed the core of the discussion section.

4 Implementation details

The empirical analysis described in this thesis was carried out using a custom software solution implemented in **Python 3** and is available on GitHub[2]. The implementation was designed to be modular, efficient, and robust, leveraging well-established scientific computing libraries to handle graph-based data and numerical calculations. The project is structured around a central execution script (`run_semantics.py`) that orchestrates the parsing of argumentation frameworks, the calculation of rankings for multiple semantics, and the final aggregation of correlation results.

The core components of the implementation are: the argumentation framework parser, the individual semantics modules, and the experimental harness responsible for execution and data logging.

4.1 Framework parsing and representation

Argumentation frameworks are parsed from the standard `.af` file format using a parser module (`util/af_parser.py`). This module reads the file line by line, processing the `paf` header to initialize the set of arguments and subsequent lines to define the attack relations.

[2] `https://github.com/marcelljawhari/ArgRankLab/`

The library used for representing the argumentation frameworks is **NetworkX**. Each framework is loaded into a `networkx.DiGraph` object, which provides a good foundation for graph-based operations. In line with the parser's implementation, all argument identifiers are handled as strings to prevent potential uncertainty with integer-based indexing.

4.2 Semantics implementation

The ranking semantics selected for this study were implemented in their own respective modules, ensuring clear separation of concerns. This modular design allows each semantic to be handled independently, as detailed in the following subsections.

4.2.1 Discussion-based ranking semantics

The **Discussion-based ranking semantics (Dbs)** was implemented in `semantics/dbs.py`.

Algorithmic approach

A naive implementation that recursively enumerates all attack paths for each argument would be computationally infeasible. The number of paths in a graph, particularly one containing cycles, can grow exponentially with the path length, making such an approach impractical for all but the smallest frameworks.

To ensure scalability, our implementation uses a method based on linear algebra, which is analytically equivalent to the path-counting definition of Dbs.

To efficiently count all paths of a given length ending at an argument, we can count the paths starting from that argument in a graph where all attack directions are reversed. This corresponds to using the **transpose** of the adjacency matrix (M^T). The total number of attack paths of length k ending at an argument a_i is therefore the sum of the i-th row in the matrix $(M^T)^k$.

The calculation of discussion vector components proceeds iteratively. For acyclic frameworks, the powers of the adjacency matrix will eventually become the zero matrix, providing a natural termination point for the algorithm. For cyclic frameworks, however, the vectors are potentially infinite. To guarantee termination in all cases, the calculation is bounded by a maximum path length. A well-justified heuristic for this bound is the number of arguments, $|A|$. This limit guarantees that all simple paths (which have a maximum length of $|A| - 1$) and all paths that traverse cycles with a length up to $|A|$ are included in the calculation. While rankings can theoretically change at path lengths greater than $|A|$, this bound provides a comprehensive finite basis for comparison and ensures a deterministic result for every framework. The procedure is detailed in Algorithm 1.

Algorithm 1 Discussion-based Semantics (Dbs)

Require: An argumentation framework $AF = (A, R)$.
Ensure: A ranking $\geq_{AF}^{Dbs}$ over A.

if $A = \emptyset$ **then return** empty ranking
end if
max_len $\leftarrow |A|$
$M \leftarrow$ Adjacency matrix of AF
$M_T \leftarrow M^T$
for all argument $a \in A$ **do**
    $Dis(a) \leftarrow \langle\rangle$ ▷ Initialize empty vector
end for
$M_{\text{power}} \leftarrow M_T$
for $k \leftarrow 1$ **to** max_len **do**
    for all argument $a_i \in A$ **do**
        $N_k(a_i) \leftarrow \sum_{j=1}^{|A|} (M_{\text{power}})_{i,j}$ ▷ Sum of i-th row
        if k is odd **then**
            $score \leftarrow N_k(a_i)$
        else
            $score \leftarrow -N_k(a_i)$
        end if
        Append $score$ to $Dis(a_i)$
    end for
    if M_{power} is the zero matrix **then break**
    end if
    $M_{\text{power}} \leftarrow M_{\text{power}} \cdot M_T$
end for
Rank arguments A using lexicographical comparison on the computed vectors $Dis(a)$.

Implementation details

The implementation of Algorithm 1 leverages the `scipy.sparse` library to represent the adjacency matrix. This is crucial for scalability. For a framework with millions of arguments, a dense matrix would be prohibitively large in terms of memory, whereas a sparse matrix stores only the non-zero entries (the attacks). The CSR (Compressed Sparse Row) format is used specifically for its efficiency in the matrix multiplication that forms the core operation of the algorithm's main loop.

The final step involves a lexicographical sort of the computed discussion vectors. Arguments with identical vectors are considered equally acceptable and are grouped into sets in the final ranking.

4.2.2 Categoriser-based ranking semantics

he **Categoriser-based ranking semantics (**Cat**)** was implemented in the `semantics/cat.py` module.

Algorithmic approach

The categoriser function, as defined in the preliminaries, creates a system of interdependent, non-linear equations for all arguments in the framework. A naive approach, such as looping through arguments and repeatedly substituting values, would be inefficient. Instead, we use a vectorized iterative method that solves for all argument strengths simultaneously. This approach is guaranteed to converge to a unique solution because the categoriser function is a **contraction mapping** on the space of possible strength assignments. A contraction mapping is a function that, when applied repeatedly, brings points closer together, eventually converging to a single, unique fixed point. This property, established by the Banach fixed-point theorem, ensures that an iterative update process will yield a stable and unique vector of argument strengths. [Ban22]

The algorithm begins with an initial strength vector (e.g., all zeros) and iteratively refines it using the update rule derived from the definition. The process continues until the change between consecutive strength vectors falls below a predefined tolerance threshold, which means that we reached convergence. The procedure is detailed in Algorithm 2.

Algorithm 2 Categoriser-based Semantics (Cat)

Require: An argumentation framework $AF = (A, R)$, tolerance tol, max iterations max_iter.
Ensure: A ranking $\geq_{AF}^{Cat}$ over A.

if $A = \emptyset$ **then return** empty ranking
end if
$M \leftarrow$ Adjacency matrix of AF
$M_T \leftarrow M^T$ ▷ Transpose for efficient attacker calculations
$S \leftarrow$ zero vector of size $|A|$ ▷ Initialize strengths vector
for $k \leftarrow 1$ **to** max_iter **do**
    $S_{\text{old}} \leftarrow S$
    attacker_sums $\leftarrow M_T \cdot S_{\text{old}}$ ▷ Vectorized sum of attacker strengths
    $S \leftarrow 1/(1 + \text{attacker\_sums})$ ▷ Vectorized update rule
    if $\max(|S - S_{\text{old}}|) <$ tol **then** ▷ Check for convergence
        break
    end if
end for
Create ranking $\geq_{AF}^{Cat}$ by sorting arguments A in descending order of their strengths in S.

Implementation details

The implementation of Algorithm 2 is designed for efficiency by leveraging the `NumPy` and `SciPy` libraries. Instead of a Python loop over individual arguments, the strength vector for all arguments is updated in a single vectorized operation in each iteration.

The core of this efficiency lies in the use of a transposed adjacency matrix, represented as a `scipy.sparse.csr_array`. A sparse matrix-vector product (M_T `@` S_{old}) computes the sum of attacker strengths for every argument simultaneously. This is substantially more efficient than iterating through each argument and its attackers in pure Python, as it delegates the computationally intensive loops to optimized, low-level C code within the `NumPy/SciPy` backend.

Convergence is checked by calculating the difference between the new and old strength vectors, and if it is less than the tolerance threshold convergence is reached. Once the strengths have converged, the final ranking is constructed by sorting the arguments based on their strength scores, grouping arguments whose scores differ by less than the tolerance value.

4.2.3 Probabilistic ranking semantics

The implementation of probabilistic ranking semantics is divided into two distinct strategies, reflecting the underlying computational complexity of the different classical semantics. For semantics where the probability of acceptance is tractable to compute directly (**admissible** and **stable**), an analytical approach is used. For computationally hard semantics (**grounded, complete, preferred, and ideal**), the implementation uses a function that dynamically chooses between an exact calculation for small graphs and a **Monte Carlo simulation** for larger ones.

Analytical implementation

For the admissible and stable semantics, determining the probability of an argument's acceptance can be done without simulation. This is inspired by the work of Fazzinga et al. [FFP$^+$13], which shows these problems are manageable. Our implementation calculates the probability that a singleton set containing just the argument in question, $\{a\}$, satisfies the semantic's criteria.

Admissible semantics (`ProbAdmissible`) The score for an argument a is its probability of being an admissible set on its own. This probability is the product of three independent events: (1) argument a must exist, (2) the set $\{a\}$ must be conflict-free, and (3) $\{a\}$ must be defended from all its potential attackers. The probability of defense against an attacker b is the probability that b does not exist, or that b exists and is attacked by a. The full procedure is detailed in Algorithm 3.

Algorithm 3 Analytical Admissible Probability

Require: An argumentation framework $AF = (A, R)$, uniform probability p.
Ensure: A score for each argument $a \in A$.

for all argument $a \in A$ **do**
  $P_{\text{exists}} \leftarrow p$
  if AF contains self-attack (a, a) **then**
    $P_{\text{cf}} \leftarrow 0$
  else
    $P_{\text{cf}} \leftarrow 1$
  end if
  $P_{\text{defended}} \leftarrow 1$
  for all attacker b of a **do**
    if AF contains attack (a, b) **then**
      $P_{\text{defense\_vs\_b}} \leftarrow (1 - p) + p$ ▷ b not exist OR (b exists AND a attacks b)
    else
      $P_{\text{defense\_vs\_b}} \leftarrow (1 - p)$ ▷ b must not exist
    end if
    $P_{\text{defended}} \leftarrow P_{\text{defended}} \times P_{\text{defense\_vs\_b}}$
  end for
  $\text{score}(a) \leftarrow P_{\text{exists}} \times P_{\text{cf}} \times P_{\text{defended}}$
end for
return scores

Stable semantics (`ProbStable`) The score for an argument a is its probability of forming a stable extension on its own. For $\{a\}$ to be stable, it must be conflict-free and must attack every other argument that exists in the framework. Calculating this involves multiplying many small probabilities, which can lead to floating-point underflow. To ensure numerical stability, the implementation calculates the **log-probability** instead. The final score is the sum of the log-probabilities of each necessary event, as detailed in Algorithm 4.

Algorithm 4 Analytical Stable Log-Probability

Require: An argumentation framework $AF = (A, R)$, uniform probability p.
Ensure: A log-score for each argument $a \in A$.

for all argument $a \in A$ **do**
  if AF contains self-attack (a, a) **then**
    log-score$(a) \leftarrow -\infty$
    continue
  end if
  logP_exists $\leftarrow \log(p)$
  $N_{\text{non-attacked}} \leftarrow |\{d \in A \setminus \{a\} \mid (a, d) \notin R\}|$
  logP_attacks_all $\leftarrow N_{\text{non-attacked}} \times \log(1 - p)$ ▷ All non-attacked args must not exist
  log-score$(a) \leftarrow$ logP_exists + logP_attacks_all
end for
return log-scores

Exact and simulation-based framework

For computationally hard semantics, a more general approach is required. The implementation uses a function that selects the best strategy based on the framework's size. The final score for an argument is derived from the set of credulously accepted arguments within each subgraph. This set is determined by finding all extensions and taking their union, a baseline approach known as **Exhaustive Extension Enumeration (EEE)** [BTCV25].

Exact Calculation For small frameworks where the total number of subgraphs ($2^{|A|}$) is manageable (e.g., less than the number of samples intended for simulation), an exact calculation is performed. The algorithm iterates through every possible subgraph, calculates its exact probability of occurring, determines the credulously accepted arguments within it, and adds that probability to each accepted argument's total score. This provides a perfect, deterministic result without approximation.

Monte Carlo Simulation For larger frameworks where enumerating all subgraphs is infeasible, the system falls back to a Monte Carlo simulation. It generates a large number of random subgraphs based on the existence probability p for each argument. For each sample, it finds the accepted arguments and counts their occurrences. The final score for an argument is its total acceptance count divided by the number of samples, providing an estimate of its true acceptance probability. The dual-strategy approach is formalized in Algorithm 5.

Algorithm 5 Probabilistic Semantics (General Method)

Require: $AF = (A, R)$, probability p, samples N_s.
Ensure: A score for each argument $a \in A$.

Initialize scores for all $a \in A$ to 0.
if $2^{|A|} < N_s$ **then** ▷ Use Exact Calculation
  for all subgraph AF_X of AF **do**
    $P(X) \leftarrow p^{|X|} \cdot (1-p)^{|A \setminus X|}$
    $E \leftarrow$ FindCredulouslyAccepted(AF_X)
    for all argument $a \in E$ **do**
      score$(a) \leftarrow$ score$(a) + P(X)$
    end for
  end for
else ▷ Use Monte Carlo Simulation
  Initialize counts for all $a \in A$ to 0.
  for $i \leftarrow 1$ to N_s **do**
    $A_X \leftarrow \{a \in A \mid \text{random}() < p\}$
    $AF_X \leftarrow$ subgraph induced by A_X
    $E \leftarrow$ FindCredulouslyAccepted(AF_X)
    for all argument $a \in E$ **do**
      count$(a) \leftarrow$ count$(a) + 1$
    end for
  end for
  for all argument $a \in A$ **do**
    score$(a) \leftarrow$ count$(a)/N_s$
  end for
end if
return scores

Extension-finding algorithms

The core of the simulation framework is the 'FindCredulouslyAccepted' method used in Algorithm 5, which is implemented differently for each semantic.

Grounded The grounded extension is found using an iterative algorithm that calculates the least fixed point of the characteristic function.

Algorithm 6 Find Grounded Extension

Require: A subgraph $AF_X = (A_X, R_X)$.
Ensure: The grounded extension E_{gr} of AF_X.

$E_{gr} \leftarrow \emptyset$
loop
  $E_{new} \leftarrow E_{gr}$
  $D \leftarrow \{a \in A_X \mid a \text{ is defended by } E_{gr}\}$ ▷ Apply characteristic function
  $E_{gr} \leftarrow D$
  if $E_{gr} = E_{new}$ **then** ▷ Fixpoint reached
    return E_{gr}
  end if
end loop

Complete All complete extensions are enumerated using a SAT solver. The properties of a complete extension are encoded as a Boolean formula, and the solver finds all satisfying models.

Algorithm 7 Find All Complete Extensions

Require: A subgraph $AF_X = (A_X, R_X)$.
Ensure: The set of all complete extensions $\mathcal{E}_{co}$ of AF_X.

$\mathcal{E}_{co} \leftarrow \emptyset$
$CNF \leftarrow$ EncodeCompleteSemantics(AF_X)
solver $\leftarrow$ InitializeSATsolver(CNF)
while solver.solve() **do**
  $M \leftarrow$ solver.getModel()
  $E \leftarrow$ DecodeModelToExtension(M)
  Add E to $\mathcal{E}_{co}$
  solver.addClause($\neg M$) ▷ Add blocking clause to find next solution
end while
return $\mathcal{E}_{co}$

Preferred The preferred extensions are found by first generating all complete extensions and then filtering that set to keep only those that are not a strict subset of any other extension.

Algorithm 8 Find All Preferred Extensions

Require: A subgraph $AF_X = (A_X, R_X)$.
Ensure: The set of all preferred extensions $\mathcal{E}_{pr}$ of AF_X.

```
1: E_co ← FindAllCompleteExtensions(AF_X)          ▷ Using Algorithm 7
2: E_pr ← ∅
3: for all E_1 ∈ E_co do
4:     isMaximal ← true
5:     for all E_2 ∈ E_co do
6:         if E_1 ≠ E_2 and E_1 ⊂ E_2 then
7:             isMaximal ← false
8:             break
9:         end if
10:    end for
11:    if isMaximal then
12:        Add E_1 to E_pr
13:    end if
14: end for
15: return E_pr
```

Ideal The ideal extension is found using the **Conflict-Driven Ideal Search (CDIS)** algorithm [TCV+21], which computes the ideal extension directly without the need to first enumerate all preferred extensions.

Algorithm 9 Find Ideal Extension (CDIS Algorithm from [TCV+21])

Require: A subgraph $AF_X = (A_X, R_X)$.
Ensure: The ideal extension E_{id} of AF_X.

▷ **Phase 1: Compute Preferred Super-Core**
$P \leftarrow A_X$
loop
  $S \leftarrow$ FindAdmissibleAttacker(P, AF_X) ▷ SAT query to find admissible set S attacking P
  if $S = \emptyset$ **then**
    break
  end if
  $P \leftarrow P \setminus \{a \in P \mid \exists s \in S, (s, a) \in R_X\}$
end loop
▷ **Phase 2: Compute largest admissible subset of P**
$AF_P \leftarrow$ subgraph induced by P
loop
  $U \leftarrow \{a \in AF_P \mid a$ is not defended by $AF_P\}$
  if $U = \emptyset$ **then**
    break
  end if
  Remove arguments in U from AF_P
end loop
$E_{id} \leftarrow$ remaining arguments in AF_P
return E_{id}

4.2.4 Serialisability-based ranking semantics

The implementation of the **Serialisability-based ranking semantics (**Ser**)** is based on the methods described by Bengel and Thimm [BT23]. It is implemented in the `semantics/ser.py` module and delegates the most computationally expensive task, finding initial sets, to a boolean satisfiability (SAT) solver.

Algorithmic approach

The algorithm determines an argument's serialisation index by recursively constructing valid serialisation sequences. The main procedure explores the tree of possible sequences, where each node represents the choice of an initial set from the current state of the argumentation framework.

The search begins by finding all initial sets of the original framework; any argument within these sets is assigned a serialisation index of 1. For each of these initial sets, a recursive exploration begins. In each step, the framework is simplified by computing the **reduct** with respect to the set of arguments already accepted in the current sequence. The algorithm then finds the initial sets of this new, smaller framework and assigns the current step number as the index for any arguments found.

To manage the complexity of this search, the algorithm incorporates a critical **pruning optimization**. Before exploring a branch of the search tree (i.e., before computing the initial sets of a reduct), it first checks if any remaining arguments in that branch could possibly achieve a better (lower) serialisation index than one already found for them in a previous branch. If no improvement is possible, the entire branch is abandoned, significantly reducing the number of required computations. This recursive, pruning-based search is detailed in Algorithm 10.

Algorithm 10 Recursive Serialisation Index Search

Require: An accepted set $S_{accepted}$, current step k.
Global: Full framework AF, indices map ser_indices, max depth d_{max}.

procedure EXPLORESEQUENCES($S_{accepted}$, k)
  if $k > d_{max}$ **then return**
  end if
  $S_{attacked} \leftarrow$ all arguments attacked by $S_{accepted}$
  $A_{reduct} \leftarrow A \setminus (S_{accepted} \cup S_{attacked})$
  if $\forall a \in A_{reduct}$, ser_indices$[a] \leq k$ **then return**
  end if ▷ Pruning step
  $AF_{reduct} \leftarrow$ subgraph induced by A_{reduct}
  $\mathcal{IS} \leftarrow$ FindInitialSetsSAT(AF_{reduct}) ▷ Using Algorithm 11
  if $\mathcal{IS} = \emptyset$ **then return**
  end if
  for all $S_{new} \in \mathcal{IS}$ **do**
    for all argument $a \in S_{new}$ **do**
      ser_indices$[a] \leftarrow \min($ser_indices$[a], k)$
    end for
    EXPLORESEQUENCES($S_{accepted} \cup S_{new}$, $k + 1$)
  end for
end procedure

Implementation details

The most challenging part of the algorithm is finding all initial sets (minimal non-empty admissible sets) of a given framework, which is delegated to the **PySAT** library with the **Glucose4** solver. This is handled by a dedicated function that follows a multi-stage query process to ensure both admissibility and minimality.

First, the properties of a non-empty admissible set are encoded into a conjunctive normal form (CNF) formula. The main SAT solver is then used to find a model for this formula, which corresponds to an admissible set. To guarantee this set is minimal, a second, temporary SAT query is performed. This check determines if any proper subset of the found set is also admissible. If no such subset exists, the original set is confirmed to be an initial set, and a "blocking clause" is added to the main solver to prevent finding this same set again. If a smaller admissible subset does exist, the current set is discarded as non-minimal, and a different blocking clause is added to guide the search towards

other solutions. This procedure, detailed in Algorithm 11, is repeated until all initial sets have been enumerated.

Algorithm 11 Finding Initial Sets via SAT

Require: An argumentation framework AF_X.
Ensure: A list of all initial sets $\mathcal{IS}$ of AF_X.

$\mathcal{IS} \leftarrow \emptyset$
$C_{base} \leftarrow$ EncodeAdmissible(AF_X)
Add clause to C_{base} to ensure non-emptiness.
$S_{main} \leftarrow$ InitializeSolver(C_{base})
while S_{main}.solve() **do**
    $M \leftarrow S_{main}$.getModel()
    $S_{adm} \leftarrow$ DecodeModelToExtension(M)
    isMinimal $\leftarrow$ true
    if $|S_{adm}| > 1$ **then**
        $C_{min} \leftarrow C_{base}$
        Add clauses to C_{min} to search for a proper subset of S_{adm}.
        $S_{temp} \leftarrow$ InitializeSolver(C_{min})
        if S_{temp}.solve() **then**
            isMinimal $\leftarrow$ false ▷ A smaller admissible set was found
        end if
    end if
    if isMinimal **then**
        Add S_{adm} to $\mathcal{IS}$
    end if
    Add blocking clause $\neg M$ to S_{main} to find a new solution.
end while
return $\mathcal{IS}$

4.3 Experimental harness and automation

The main script, `run_semantics.py`, serves as the experimental harness. It automates the entire analysis pipeline:

1. **File discovery:** It recursively finds all `.af` files within the specified benchmark directories.

2. **Timeout management:** Since some frameworks in our benchmarks are still too large to calculate our semantics on within a reasonable timeframe, a timeout of (600 seconds) is enforced. If a calculation exceeds this limit, the process is terminated, and the framework is marked as a timeout case to be skipped in future runs.

3. **Correlation and aggregation:** After calculating the rankings for a framework, the

script uses thes `scipy.stats` library to compute **Kendall's tau** and **Spearman's rho** for all pairs of semantics. The results are saved to individual CSV files.

5 Experimental setup

All experiments were conducted on a machine equipped with an **AMD Ryzen 9 5900X** processor and **32 GB** of memory, running the **Linux Mint 22.1** operating system.

The implementation, as described in Section 4, was executed using **Python** version **3.12.3**. The primary libraries leveraged for the analysis include **NetworkX 3.5**, **NumPy 2.2.6**, **SciPy 1.15.3**, **pandas 2.3.0**, and **PySAT 1.8.dev17** for the SAT-based computations.

The empirical evaluation utilized two distinct benchmark collections to accommodate the varying computational demands of the semantics. Due to the high complexity of the slow semantics (Ser, Pro_c, etc.), the experimental procedure was partitioned. To provide a clear overview of their scale and structural properties, key statistics for both datasets are summarized in Table 1 and Table 2.

Metric	**Value**
Total Frameworks	329
Arguments ($\|A\|$)	
Min	100
Max	2,500,000
Mean	29,790.98
Median	796
Attacks ($\|R\|$)	
Min	101
Max	23,429,120
Mean	1,002,470.24
Median	22,512
Mean Density	0.1405
Cyclicity	
Cyclic Frameworks	314
Acyclic Frameworks	15

Table 1: Statistical overview of the ICCMA 2023 benchmark dataset.

Metric	Value
Total Frameworks	82
Arguments ($\|A\|$)	
Min	4
Max	100
Mean	32.61
Median	28
Attacks ($\|R\|$)	
Min	0
Max	1,026
Mean	244.50
Median	67
Mean Density	0.1537
Cyclicity	
Cyclic Frameworks	50
Acyclic Frameworks	32

Table 2: Statistical overview of the Tweety-generated benchmark dataset.

The partitioning was as follows:

- The fast semantics (Cat, Dbs, Pro_s, Pro_a) were evaluated on the full set of benchmarks, which includes both the main track of the ICCMA 2023 and a second collection of smaller frameworks.
- The slow semantics (Ser, Pro_c, Pro_i, Pro_g, Pro_p) were evaluated exclusively on the second collection, which was generated using methods from the TweetyProject to ensure that computations could complete within the established 600-second timeout.

This design allows for a broad comparison of fast semantics on standard, large-scale benchmarks, while still enabling a feasible comparison across all semantics on a set of smaller, more manageable frameworks.

6 Results

This section presents the findings of the correlation analysis. The results are derived from the experimental procedure detailed in Section 3, using the setup described in Section 5. The primary goal is to provide a data-driven answer to the research questions concerning the degree of correlation between the selected ranking semantics (RQ1) and the influence of structural properties on their agreement (RQ2).

The results are presented as correlation matrices showing the **average Kendall's tau coefficients**. These tables represent a hybrid analysis. Comparisons between pairs of fast semantics (Cat, Dbs, Pro_a, and Pro_s) were conducted across the entire benchmark collection. In contrast, any comparison involving at least one slow semantic (e.g., Ser, Pro_p) was performed exclusively on the smaller, Tweety-generated dataset to ensure computational feasibility. For both fast and slow semantics, the tables also indicate the total number of AFs used in each comparison, which differ from the counts in Tables 1 and 2 due to some AFs reaching the timeout limit.

The corresponding data for Spearman's rho, along with median and standard deviation values, were found to be highly consistent with these findings and are available in the appendix.

6.1 Overall correlation

The primary analysis was conducted on the Tweety-generated dataset, as this was the collection where all implemented semantics could be run, allowing for a complete pairwise comparison. Tables 3, 4 and 5 show the average and median Kendall's tau correlation between all pairs of semantics along with the standard deviation.

	Cat	Dbs	Ser	Pro_a	Pro_c	Pro_g	Pro_i	Pro_p	Pro_s
Cat	1.000	0.279	0.246	0.086	0.257	0.270	0.276	0.216	0.017
Dbs	0.279	1.000	0.250	0.098	0.258	0.237	0.235	0.224	0.017
Ser	0.246	0.250	1.000	0.237	0.162	0.212	0.153	0.159	0.101
Pro_a	0.086	0.098	0.237	1.000	0.204	0.187	0.196	0.206	0.155
Pro_c	0.257	0.258	0.162	0.204	1.000	0.251	0.235	0.489	0.042
Pro_g	0.270	0.237	0.212	0.187	0.251	1.000	0.279	0.287	0.042
Pro_i	0.276	0.235	0.153	0.196	0.235	0.279	1.000	0.258	0.062
Pro_p	0.216	0.224	0.159	0.206	0.489	0.287	0.258	1.000	0.053
Pro_s	0.017	0.017	0.101	0.155	0.042	0.042	0.062	0.053	1.000

Table 3: Overall Hybrid Analysis (Fast-vs-Fast on 298 AFs, others on 65 Tweety AFs) - Kendall's Tau (τ) (Average)

The data in Table 3 shows a range of correlations. The highest average correlation is observed between the **probabilistic-complete (Pro_c) and probabilistic-preferred (Pro_p) semantics**, with a Kendall's τ of **0.489**. A notable, though more moderate, positive correlation is also present between the **Cat and Dbs semantics** (τ = **0.279**). Conversely, several semantic pairs exhibit minimal correlation. The **probabilistic-stable (Pro_s) semantics**, for instance, shows a very low average correlation with both **Cat** (τ = **0.017**) and other probabilistic semantics like **probabilistic-ideal (Pro_i)** (τ = **0.062**).

Table 5 details the consistency of these relationships. The correlation between **probabilistic-grounded (Pro_g) and probabilistic-preferred (Pro_p)** is the most varied, exhibiting the highest standard deviation at **0.382**. In contrast, the relationship between **Dbs and probabilistic-stable (Pro_s) semantics** is highly consistent, showing the lowest stan-

	Cat	Dbs	Ser	Pro_a	Pro_c	Pro_g	Pro_i	Pro_p	Pro_s
Cat	1.000	0.166	0.141	0.025	0.147	0.147	0.138	0.071	0.001
Dbs	0.166	1.000	0.167	0.034	0.123	0.124	0.116	0.112	0.005
Ser	0.141	0.167	1.000	0.149	0.033	0.118	0.074	0.064	0.073
Pro_a	0.025	0.034	0.149	1.000	0.074	0.073	0.068	0.124	0.075
Pro_c	0.147	0.123	0.033	0.074	1.000	0.121	0.086	0.409	0.022
Pro_g	0.147	0.124	0.118	0.073	0.121	1.000	0.124	0.133	-0.007
Pro_i	0.138	0.116	0.074	0.068	0.086	0.124	1.000	0.109	0.019
Pro_p	0.071	0.112	0.064	0.124	0.409	0.133	0.109	1.000	0.011
Pro_s	0.001	0.005	0.073	0.075	0.022	-0.007	0.019	0.011	1.000

Table 4: Overall Hybrid Analysis (Fast-vs-Fast on 298 AFs, others on 65 Tweety AFs) - Kendall's Tau (τ) (Median)

	Cat	Dbs	Ser	Pro_a	Pro_c	Pro_g	Pro_i	Pro_p	Pro_s
Cat	0.000	0.293	0.342	0.216	0.353	0.365	0.356	0.359	0.148
Dbs	0.293	0.000	0.338	0.218	0.341	0.347	0.357	0.343	0.134
Ser	0.342	0.338	0.000	0.309	0.333	0.327	0.327	0.330	0.241
Pro_a	0.216	0.218	0.309	0.000	0.350	0.313	0.312	0.332	0.262
Pro_c	0.353	0.341	0.333	0.350	0.000	0.366	0.375	0.287	0.261
Pro_g	0.365	0.347	0.327	0.313	0.366	0.000	0.371	0.382	0.240
Pro_i	0.356	0.357	0.327	0.312	0.375	0.371	0.000	0.354	0.257
Pro_p	0.359	0.343	0.330	0.332	0.287	0.382	0.354	0.000	0.244
Pro_s	0.148	0.134	0.241	0.262	0.261	0.240	0.257	0.244	0.000

Table 5: Overall Hybrid Analysis (Fast-vs-Fast on 298 AFs, others on 65 Tweety AFs) - Kendall's Tau (τ) (Standard Deviation)

dard deviation at **0.134**.

Finally, a comparison between the average (Table 3) and median (Table 4) values reveals that median correlations are frequently lower than their corresponding averages. For example, the Cat and Dbs pairing has an average correlation of **0.279** but a median of just **0.166**.

6.2 Impact of framework structure

Beyond the overall summary, the analysis also investigated how correlations behave within specific subsets of argumentation frameworks, specifically comparing acyclic and cyclic graphs. This reveals that the presence of cycles is a critical factor influencing the relationships between the semantics.

In **acyclic frameworks**, as detailed in Table 7, the correlations between many of the semantics are notably strong. A very high level of agreement is observed between the **Cat**, **Dbs** and **probabilistic-admissible (**Pro_a**)** semantics, with the correlation between Cat and Dbs reaching **0.862**. Among the other probabilistic semantics, the relation-

	Cat	Dbs	Ser	Pro_a	Pro_c	Pro_g	Pro_i	Pro_p	Pro_s
Cat	1.000	0.230	0.072	0.034	0.077	0.120	0.127	0.046	-0.003
Dbs	0.230	1.000	0.076	0.046	0.092	0.091	0.088	0.072	-0.000
Ser	0.072	0.076	1.000	0.105	0.042	0.120	0.043	0.055	0.061
Pro_a	0.034	0.046	0.105	1.000	0.056	0.055	0.073	0.079	0.145
Pro_c	0.077	0.092	0.042	0.056	1.000	0.093	0.055	0.417	0.009
Pro_g	0.120	0.091	0.120	0.055	0.093	1.000	0.111	0.100	0.004
Pro_i	0.127	0.088	0.043	0.073	0.055	0.111	1.000	0.078	0.028
Pro_p	0.046	0.072	0.055	0.079	0.417	0.100	0.078	1.000	0.013
Pro_s	-0.003	-0.000	0.061	0.145	0.009	0.004	0.028	0.013	1.000

Table 6: Cyclic Hybrid Analysis (Fast-vs-Fast on 275 AFs, others on 45 Tweety AFs) - Kendall's Tau (τ) (Average)

	Cat	Dbs	Ser	Pro_a	Pro_c	Pro_g	Pro_i	Pro_p	Pro_s
Cat	1.000	0.862	0.637	0.713	0.663	0.607	0.611	0.599	0.252
Dbs	0.862	1.000	0.643	0.716	0.633	0.565	0.567	0.564	0.231
Ser	0.637	0.643	1.000	0.532	0.430	0.420	0.401	0.392	0.190
Pro_a	0.713	0.716	0.532	1.000	0.536	0.482	0.473	0.491	0.277
Pro_c	0.663	0.633	0.430	0.536	1.000	0.604	0.642	0.651	0.115
Pro_g	0.607	0.565	0.420	0.482	0.604	1.000	0.657	0.710	0.128
Pro_i	0.611	0.567	0.401	0.473	0.642	0.657	1.000	0.662	0.139
Pro_p	0.599	0.564	0.392	0.491	0.651	0.710	0.662	1.000	0.144
Pro_s	0.252	0.231	0.190	0.277	0.115	0.128	0.139	0.144	1.000

Table 7: Acyclic Hybrid Analysis (Fast-vs-Fast on 23 AFs, others on 20 Tweety AFs) - Kendall's Tau (τ) (Average)

ship between **probabilistic-complete (**Pro_c**)** and **probabilistic-preferred (**Pro_p**)** is also strong, with a τ of **0.651**.

This landscape of high agreement shifts significantly when cycles are introduced (Table 6). The strong correlation between **Cat** and **Dbs**, for example, plummets from **0.862** in acyclic graphs to just **0.230** in cyclic ones. Similarly, the correlation between **probabilistic-ideal (**Pro_i**)** and **probabilistic-grounded (**Pro_g**)** drops dramatically, from **0.657** to **0.111**. While the link between the **probabilistic-complete** and **probabilistic-preferred** semantic also weakens, it remains the most robust among the probabilistic approaches in cyclic frameworks, with a τ of **0.417**.

6.3 Impact of framework density

Further dissecting the data, the analysis considered the impact of framework density by comparing sparse graphs (where the number of attacks is low relative to the number of arguments) against dense graphs. This reveals how the level of interconnectedness affects semantic agreement across all framework types.

	Cat	Dbs	Ser	Pro_a	Pro_c	Pro_g	Pro_i	Pro_p	Pro_s
Cat	1.000	0.196	0.792	0.067	0.533	0.510	0.523	0.496	0.027
Dbs	0.196	1.000	0.789	0.072	0.518	0.501	0.523	0.493	0.024
Ser	0.792	0.789	1.000	0.566	0.413	0.399	0.445	0.377	0.272
Pro_a	0.067	0.072	0.566	1.000	0.457	0.420	0.362	0.365	0.144
Pro_c	0.533	0.518	0.413	0.457	1.000	0.539	0.433	0.581	0.137
Pro_g	0.510	0.501	0.399	0.420	0.539	1.000	0.535	0.576	0.236
Pro_i	0.523	0.523	0.445	0.362	0.433	0.535	1.000	0.533	0.237
Pro_p	0.496	0.493	0.377	0.365	0.581	0.576	0.533	1.000	0.277
Pro_s	0.027	0.024	0.272	0.144	0.137	0.236	0.237	0.277	1.000

Table 8: Sparse Hybrid Analysis (Fast-vs-Fast on 143 AFs, others on 7 Tweety AFs) - Kendall's Tau (τ) (Average)

	Cat	Dbs	Ser	Pro_a	Pro_c	Pro_g	Pro_i	Pro_p	Pro_s
Cat	1.000	0.321	0.052	0.030	0.049	0.063	0.111	0.016	0.007
Dbs	0.321	1.000	0.063	0.035	0.065	0.057	0.077	0.048	0.005
Ser	0.052	0.063	1.000	0.099	0.054	0.178	0.055	0.079	0.060
Pro_a	0.030	0.035	0.099	1.000	0.051	0.042	0.076	0.076	0.214
Pro_c	0.049	0.065	0.054	0.051	1.000	0.078	0.039	0.410	-0.002
Pro_g	0.063	0.057	0.178	0.042	0.078	1.000	0.074	0.072	0.016
Pro_i	0.111	0.077	0.055	0.076	0.039	0.074	1.000	0.059	0.048
Pro_p	0.016	0.048	0.079	0.076	0.410	0.072	0.059	1.000	0.001
Pro_s	0.007	0.005	0.060	0.214	-0.002	0.016	0.048	0.001	1.000

Table 9: Dense Hybrid Analysis (Fast-vs-Fast on 98 AFs, others on 30 Tweety AFs) - Kendall's Tau (τ) (Average)

In **sparse frameworks**, as detailed in Table 8, the correlations present a complex picture. Surprisingly, the non-probabilistic **Serialisability-based (Ser) semantics** shows a very strong positive correlation with both **Cat** (τ = **0.792**) and **Dbs** (τ = **0.789**). Furthermore, many probabilistic semantics show moderate to strong positive correlations with each other in this setting, such as between **probabilistic-preferred (Pro_p) and probabilistic-grounded (Pro_g)** (τ = **0.576**).

This pattern changes dramatically in **dense frameworks** (Table 9). The strong positive correlations involving the **Ser** semantics disappear entirely, with its correlation to **Cat** plummeting to just **0.052**. The correlations between many probabilistic semantics also weaken considerably; for instance, the relationship between Pro_a and Pro_i drops from 0.362 to a near-zero correlation of **0.076**. Meanwhile, the correlation between **Cat** and **Dbs** (τ = **0.321**) becomes more pronounced in dense graphs, while the strong link between **probabilistic-complete (Pro_c) and probabilistic-preferred (Pro_p)** weakens but remains significant (τ = **0.410**).

7 Discussion

These results quantify the relationships between key ranking semantics. This section interprets these findings to answer the research questions posed in Section 3, exploring the conceptual similarities and divergences of the semantics and the significant influence of framework topology on their agreement.

The first finding, in response to **RQ1**, is that the overall correlation between most pairs of semantics is generally low. As shown in Table 3, the majority of average correlation scores are below 0.3, indicating that choosing a ranking semantics is not trivial and significantly impacts argument evaluation. The most notable exception is the strong correlation between the probabilistic-complete (Pro_c) and probabilistic-preferred (Pro_p) semantics ($\tau =$ **0.489**). This result is makes sense, as preferred extensions are defined as the maximal complete extensions. A similar, though more moderate, correlation is observed between probabilistic-grounded (Pro_g) and probabilistic-complete (Pro_c) semantics ($\tau =$ **0.251**), which is also expected, as the grounded extension is the minimal complete extension.

Regarding **RQ2**, the analysis reveals that framework structure is an important factor in the level of agreement between semantics. The contrast between acyclic (Table 7) and cyclic (Table 6) frameworks is particularly clear. In acyclic graphs, the absence of cyclical dependencies simplifies the argumentative structure, leading to a high degree of consensus. One of our most interesting findings was how closely the Cat and Dbs semantics agreed in frameworks without any cycles, showing a Kendall's Tau correlation of **0.862**. This is surprising because the two methods evaluate arguments in fundamentally different ways. The Categoriser-based (Cat) semantics takes a local approach by primarily considering the strength of an argument's immediate attackers , whereas the Discussion-based (Dbs) semantics uses a global perspective that accounts for all the long attack and defense chains. The introduction of cycles, however, fundamentally breaks this consensus, with the Cat and Dbs correlation plummeting to **0.230**. This drop shows that the two semantics deal with cycles and mutual attacks in very different ways.

The impact of framework density also reveals important behavioral patterns. The analysis of sparse versus dense graphs (Tables 8 and 9) highlights the unique nature of the Serialisability-based (Ser) semantics. Its very strong correlation with Cat ($\tau =$ **0.792**) and Dbs ($\tau =$ **0.789**) in sparse frameworks is interesting because Ser's constructive, global logic for building admissible sets is conceptually different from the more local attacker-based logic of Cat and the path-counting, global logic of Dbs. A possible explanation is that in sparse graphs with fewer complex interactions, the way Ser builds admissible sets ends up producing results similar to these simpler evaluations. This alignment vanishes in dense frameworks, where the complexity of constructing admissible sets increases and Ser's correlation with Cat drops to just **0.052**. Furthermore, the effect of density is not uniform: the Cat and Dbs correlation strengthens in dense graphs, whereas the link between Pro_c and Pro_p, while still strong, weakens from **0.581** in sparse graphs to **0.410** in dense ones.

Finally, these empirical results provide some insight into **RQ3** on how these semantics can be interpreted. The general lack of high correlation suggests that there is no single, monolithic concept of "argument strength." Instead, each semantic formalizes a different, intuition about what makes an argument plausible. This is particularly evident in the probabilistic approaches. The analytical methods (Pro_a and Pro_s), which calculate the probability of a single argument forming an extension, show weak correlation with the Monte Carlo-based methods ($Pro_g, Pro_c, Pro_p, Pro_i$), which evaluate an argument's acceptance within the context of entire sampled subgraphs. Some of the gap might also be due to the inaccuracy of Monte Carlo simulations, which only sample frameworks instead of checking them all. More research is needed here, for example by running Monte Carlo simulations for all probabilistic semantics using all exactyl the same set of samples. The moderate correlation between the local, iterative approach of Cat and the global, path-based logic of Dbs shows that they capture related, but still different, aspects of how an argument has influence.

The structural analysis gives further insight. In acyclic graphs, where the argument structure is straightforward, many semantics end up agreeing, showing that different ways of judging argument strength can lead to similar results. Adding cycles and higher density makes each semantic reveal its own distinct way of evaluating arguments. The behavior of the Ser semantics is a key example: its strong alignment with local semantics in sparse graphs suggests its constructive nature is conceptually close to simpler methods in low-complexity scenarios, but it follows a much more distinct logic as the graph becomes more interconnected.

8 Conclusion

This thesis conducted an empirical correlation analysis to investigate the relationships between ranking-based semantics in abstract argumentation. By implementing a suite of semantics, including discussion-based, categoriser-based, serialisability-based, and various instantiations of probabilistic semantics, and applying them to a diverse collection of benchmark datasets, this work provides a quantitative foundation for understanding their comparative behaviour.

The findings offer answers to the initial research questions. In response to **RQ1**, the analysis confirms that the overall correlation between most semantics is low, with most average Kendall's τ scores falling below 0.3. This highlights their conceptual differences and shows that the choice of a ranking semantic is a critical decision with significant consequences for argument evaluation. The main exceptions were the strong correlation between probabilistic-complete and probabilistic-preferred semantics ($\tau =$ **0.489**) and the moderate correlation between probabilistic-grounded and probabilistic-complete semantics ($\tau =$ **0.251**). These results empirically support the close theoretical relationships where preferred and grounded semantics are the maximal and minimal forms of complete semantics, respectively.

Answering **RQ2**, the study demonstrated that framework topology has a big influence on semantic agreement. Cycles turned out to be a main reason for differ-

ences, drastically lowering the correlation of many pairs, such as Cat and Dbs. Framework density also revealed significant shifts in semantic alignment. Most notably, the Serialisability-based (Ser) semantics showed a very strong correlation with the more locally-focused Cat and Dbs semantics in sparse frameworks, an alignment that completely disappeared in dense graphs. Finally, regarding **RQ3**, the results show that while the semantics differ in complex cases, their strong agreement in simpler, acyclic graphs suggests they share a common basis for judging strength when the structure is clear.

The primary contribution of this work is the empirical evidence that there is no "one-size-fits-all" ranking semantics. For practitioners, this implies that the selection of a semantics should be a conscious choice informed by the expected structure of the argumentation scenario and the specific notion of "argument strength" required for the application.

This research also opens several directions for future work. One interesting finding is the strong agreement between the globally constructive Ser semantics, the locally focused Cat, and the globally path-based Dbs semantics in sparse frameworks. An agreement that disappears completely in dense graphs. This pattern deserves deeper theoretical study. Future work could also include a wider range of semantics or try different ways of measuring correlation. Another important step would be to explore whether the differences in rankings seen here matter in real-world uses, to see which approaches are the most useful in specific domains. In addition, more research is needed on the role of Monte Carlo simulation accuracy, for example by running all probabilistic semantics using Monte Carlo (even those currently calculated analytically) with exactly the same set of samples to make their results more comparable.

References

[ABN13] Leila Amgoud and Jonathan Ben-Naim. Ranking-based semantics for argumentation frameworks. In *International Conference on Scalable Uncertainty Management*, pages 134–147. Springer, 2013.

[Ban22] Stefan Banach. Sur les opérations dans les ensembles abstraits et leur application aux équations intégrales. *Fundamenta mathematicae*, 3(1):133–181, 1922.

[BBU20] Ringo Baumann, Gerhard Brewka, and Markus Ulbricht. Revisiting the foundations of abstract argumentation–semantics based on weak admissibility and weak defense. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 2742–2749, 2020.

[BT22] Lydia Blümel and Matthias Thimm. A ranking semantics for abstract argumentation based on serialisability. In *Computational Models of Argument*, pages 104–115. IOS Press, 2022.

[BT23] Lars Bengel and Matthias Thimm. Towards parallelising extension construction for serialisable semantics in abstract argumentation. In *Proceedings of the 20th International Conference on Principles of Knowledge Representation and Reasoning*, pages 732–736, 2023.

[BTCV25] Lars Bengel, Matthias Thimm, Federico Cerutti, and Mauro Vallati. Algorithms for computing the set of acceptable arguments. *International Journal of Approximate Reasoning*, page 109478, 2025.

[Del17] Jérôme Delobelle. *Ranking-based Semantics for Abstract Argumentation*. PhD thesis, Université d'Artois, 2017.

[Dun95] Phan Minh Dung. On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n-person games. *Artificial intelligence*, 77(2):321–357, 1995.

[FFP+13] Bettina Fazzinga, Sergio Flesca, Francesco Parisi, et al. On the complexity of probabilistic abstract argumentation. In *IJCAI*, pages 898–904, 2013.

[Ken38] Maurice G Kendall. A new measure of rank correlation. *Biometrika*, 30(1-2):81–93, 1938.

[PLZL14] Fuan Pu, Jian Luo, Yulai Zhang, and Guiming Luo. Argument ranking with categoriser function. In *Knowledge Science, Engineering and Management: 7th International Conference, KSEM 2014, Sibiu, Romania, October 16-18, 2014. Proceedings 7*, pages 290–301. Springer, 2014.

[Spe04] Charles Spearman. The proof and measurement of association between two things. *The American Journal of Psychology*, 15(1):72–101, 1904.

[TCR18] Matthias Thimm, Federico Cerutti, and Tjitze Rienstra. Probabilistic graded semantics. In *Computational Models of Argument*, pages 369–380. IOS Press, 2018.

[TCV+21] Matthias Thimm, Federico Cerutti, Mauro Vallati, et al. Skeptical reasoning with preferred semantics in abstract argumentation without computing preferred extensions. In *IJCAI*, pages 2069–2075, 2021.

[XC18] Yuming Xu and Claudette Cayrol. Initial sets in abstract argumentation frameworks. *Journal of Applied Non-Classical Logics*, 28(2-3):260–279, 2018.

Appendix

Grounded Extensions for All Non-Empty Subgraphs of AF_{ex}

This appendix presents the grounded extensions for all 255 non-empty subgraphs of the example argumentation framework AF_{ex}. (defined in Section 4). The results are grouped by the number of arguments in each subgraph. The notation GE stands for Grounded Extension.

Subgraph	GE	Subgraph	GE	Subgraph	GE
Subgraphs of Size 1 (8 entries)					
a	a	b	b	c	c
d	d	e	e	f	f
g	g	h	h		
Subgraphs of Size 2 (28 entries)					
a,b	a	a,c	a,c	a,d	a
a,e	a	a,f	a,f	a,g	a,g
a,h	a,h	b,c	b	b,d	b,d
b,e	b,e	b,f	b,f	b,g	b,g
b,h	b,h	c,d	c,d	c,e	c,e
c,f	f	c,g	c,g	c,h	c,h
d,e	d,e	d,f	d,f	d,g	g
d,h	d	e,f	e,f	e,g	e,g
e,h	e	f,g	f,g	f,h	f,h
g,h	h				
Subgraphs of Size 3 (56 entries)					
a,b,c	a,c	a,b,d	a	a,b,e	a
a,b,f	a,f	a,b,g	a,g	a,b,h	a,h
a,c,d	a,c	a,c,e	a,c	a,c,f	a,f
a,c,g	a,c,g	a,c,h	a,c,h	a,d,e	a
a,d,f	a,f	a,d,g	a,g	a,d,h	a,h
a,e,f	a,f	a,e,g	a,g	a,e,h	a,h
a,f,g	a,f,g	a,f,h	a,f,h	a,g,h	a,h
b,c,d	b,d	b,c,e	b,e	b,c,f	b,f
b,c,g	b,g	b,c,h	b,h	b,d,e	b,d,e
b,d,f	b,d,f	b,d,g	b,g	b,d,h	b,d
b,e,f	b,e,f	b,e,g	b,e,g	b,e,h	b,e
b,f,g	b,f,g	b,f,h	b,f,h	b,g,h	b,h
c,d,e	c,d,e	c,d,f	d,f	c,d,g	c,g
c,d,h	c,d	c,e,f	e,f	c,e,g	c,e,g
c,e,h	c,e	c,f,g	f,g	c,f,h	f,h
c,g,h	c,h	d,e,f	d,e,f	d,e,g	e,g
d,e,h	d,e	d,f,g	f,g	d,f,h	d,f
d,g,h	∅	e,f,g	e,f,g	e,f,h	e,f
e,g,h	e,g	f,g,h	f,h		
Subgraphs of Size 4 (70 entries)					
a,b,c,d	a,c	a,b,c,e	a,c	a,b,c,f	a,f
a,b,c,g	a,c,g	a,b,c,h	a,c,h	a,b,d,e	a

Continued on next page

Subgraph	GE	Subgraph	GE	Subgraph	GE
a,b,d,f	a,f	a,b,d,g	a,g	a,b,d,h	a,h
a,b,e,f	a,f	a,b,e,g	a,g	a,b,e,h	a,h
a,b,f,g	a,f,g	a,b,f,h	a,f,h	a,b,g,h	a,h
a,c,d,e	a,c	a,c,d,f	a,f	a,c,d,g	a,c,g
a,c,d,h	a,c,h	a,c,e,f	a,f	a,c,e,g	a,c,g
a,c,e,h	a,c,h	a,c,f,g	a,f,g	a,c,f,h	a,f,h
a,c,g,h	a,c,h	a,d,e,f	a,f	a,d,e,g	a,g
a,d,e,h	a,h	a,d,f,g	a,f,g	a,d,f,h	a,f,h
a,d,g,h	a,h	a,e,f,g	a,f,g	a,e,f,h	a,f,h
a,e,g,h	a,h	a,f,g,h	a,f,h	b,c,d,e	b,d,e
b,c,d,f	b,d,f	b,c,d,g	b,g	b,c,d,h	b,d
b,c,e,f	b,e,f	b,c,e,g	b,e,g	b,c,e,h	b,e
b,c,f,g	b,f,g	b,c,f,h	b,f,h	b,c,g,h	b,h
b,d,e,f	b,d,e,f	b,d,e,g	b,e,g	b,d,e,h	b,d,e
b,d,f,g	b,f,g	b,d,f,h	b,d,f	b,d,g,h	b
b,e,f,g	b,e,f,g	b,e,f,h	b,e,f	b,e,g,h	b,e,g
b,f,g,h	b,f,h	c,d,e,f	d,e,f	c,d,e,g	c,e,g
c,d,e,h	c,d,e	c,d,f,g	f,g	c,d,f,h	d,f
c,d,g,h	c	c,e,f,g	e,f,g	c,e,f,h	e,f
c,e,g,h	c,e,g	c,f,g,h	f,h	d,e,f,g	e,f,g
d,e,f,h	d,e,f	d,e,g,h	e,g	d,f,g,h	f
e,f,g,h	e,f,g				
Subgraphs of Size 5 (56 entries)					
a,b,c,d,e	a,c	a,b,c,d,f	a,f	a,b,c,d,g	a,c,g
a,b,c,d,h	a,c,h	a,b,c,e,f	a,f	a,b,c,e,g	a,c,g
a,b,c,e,h	a,c,h	a,b,c,f,g	a,f,g	a,b,c,f,h	a,f,h
a,b,c,g,h	a,c,h	a,b,d,e,f	a,f	a,b,d,e,g	a,g
a,b,d,e,h	a,h	a,b,d,f,g	a,f,g	a,b,d,f,h	a,f,h
a,b,d,g,h	a,h	a,b,e,f,g	a,f,g	a,b,e,f,h	a,f,h
a,b,e,g,h	a,h	a,b,f,g,h	a,f,h	a,c,d,e,f	a,f
a,c,d,e,g	a,c,g	a,c,d,e,h	a,c,h	a,c,d,f,g	a,f,g
a,c,d,f,h	a,f,h	a,c,d,g,h	a,c,h	a,c,e,f,g	a,f,g
a,c,e,f,h	a,f,h	a,c,e,g,h	a,c,h	a,c,f,g,h	a,f,h
a,d,e,f,g	a,f,g	a,d,e,f,h	a,f,h	a,d,e,g,h	a,h
a,d,f,g,h	a,f,h	a,e,f,g,h	a,f,h	b,c,d,e,f	b,d,e,f
b,c,d,e,g	b,e,g	b,c,d,e,h	b,d,e	b,c,d,f,g	b,f,g
b,c,d,f,h	b,d,f	b,c,d,g,h	b	b,c,e,f,g	b,e,f,g
b,c,e,f,h	b,e,f	b,c,e,g,h	b,e,g	b,c,f,g,h	b,f,h
b,d,e,f,g	b,e,f,g	b,d,e,f,h	b,d,e,f	b,d,e,g,h	b,e,g
b,d,f,g,h	b,f	b,e,f,g,h	b,e,f,g	c,d,e,f,g	e,f,g
c,d,e,f,h	d,e,f	c,d,e,g,h	c,e,g	c,d,f,g,h	f
c,e,f,g,h	e,f,g	d,e,f,g,h	e,f,g		
Subgraphs of Size 6 (28 entries)					
a,b,c,d,e,f	a,f	a,b,c,d,e,g	a,c,g	a,b,c,d,e,h	a,c,h
a,b,c,d,f,g	a,f,g	a,b,c,d,f,h	a,f,h	a,b,c,d,g,h	a,c,h
a,b,c,e,f,g	a,f,g	a,b,c,e,f,h	a,f,h	a,b,c,e,g,h	a,c,h
a,b,c,f,g,h	a,f,h	a,b,d,e,f,g	a,f,g	a,b,d,e,f,h	a,f,h

Continued on next page

Subgraph	GE	Subgraph	GE	Subgraph	GE
a,b,d,e,g,h	a,h	a,b,d,f,g,h	a,f,h	a,b,e,f,g,h	a,f,h
a,c,d,e,f,g	a,f,g	a,c,d,e,f,h	a,f,h	a,c,d,e,g,h	a,c,h
a,c,d,f,g,h	a,f,h	a,c,e,f,g,h	a,f,h	a,d,e,f,g,h	a,f,h
b,c,d,e,f,g	b,e,f,g	b,c,d,e,f,h	b,d,e,f	b,c,d,e,g,h	b,e,g
b,c,d,f,g,h	b,f	b,c,e,f,g,h	b,e,f,g	b,d,e,f,g,h	b,e,f,g
c,d,e,f,g,h	e,f,g				
Subgraphs of Size 7 (8 entries)					
a,b,c,d,e,f,g	a,f,g	a,b,c,d,e,f,h	a,f,h	a,b,c,d,e,g,h	a,c,h
a,b,c,d,f,g,h	a,f,h	a,b,c,e,f,g,h	a,f,h	a,b,d,e,f,g,h	a,f,h
a,c,d,e,f,g,h	a,f,h	b,c,d,e,f,g,h	b,e,f,g		
Subgraphs of Size 8 (1 entries)					
a,b,c,d,e,f,g,h	a,f,h				

Comprehensive Correlation Results

	Cat	Dbs	Ser	Pro_a	Pro_c	Pro_g	Pro_i	Pro_p	Pro_s
Cat	1.000	0.279	0.246	0.086	0.257	0.270	0.276	0.216	0.017
Dbs	0.279	1.000	0.250	0.098	0.258	0.237	0.235	0.224	0.017
Ser	0.246	0.250	1.000	0.237	0.162	0.212	0.153	0.159	0.101
Pro_a	0.086	0.098	0.237	1.000	0.204	0.187	0.196	0.206	0.155
Pro_c	0.257	0.258	0.162	0.204	1.000	0.251	0.235	0.489	0.042
Pro_g	0.270	0.237	0.212	0.187	0.251	1.000	0.279	0.287	0.042
Pro_i	0.276	0.235	0.153	0.196	0.235	0.279	1.000	0.258	0.062
Pro_p	0.216	0.224	0.159	0.206	0.489	0.287	0.258	1.000	0.053
Pro_s	0.017	0.017	0.101	0.155	0.042	0.042	0.062	0.053	1.000

Table 11: Overall Hybrid Analysis (Fast-vs-Fast on 298 AFs, others on 65 Tweety AFs) - Kendall's Tau (τ) (Average)

	Cat	Dbs	Ser	Pro_a	Pro_c	Pro_g	Pro_i	Pro_p	Pro_s
Cat	1.000	0.166	0.141	0.025	0.147	0.147	0.138	0.071	0.001
Dbs	0.166	1.000	0.167	0.034	0.123	0.124	0.116	0.112	0.005
Ser	0.141	0.167	1.000	0.149	0.033	0.118	0.074	0.064	0.073
Pro_a	0.025	0.034	0.149	1.000	0.074	0.073	0.068	0.124	0.075
Pro_c	0.147	0.123	0.033	0.074	1.000	0.121	0.086	0.409	0.022
Pro_g	0.147	0.124	0.118	0.073	0.121	1.000	0.124	0.133	-0.007
Pro_i	0.138	0.116	0.074	0.068	0.086	0.124	1.000	0.109	0.019
Pro_p	0.071	0.112	0.064	0.124	0.409	0.133	0.109	1.000	0.011
Pro_s	0.001	0.005	0.073	0.075	0.022	-0.007	0.019	0.011	1.000

Table 12: Overall Hybrid Analysis (Fast-vs-Fast on 298 AFs, others on 65 Tweety AFs) - Kendall's Tau (τ) (Median)

	Cat	Dbs	Ser	Pro_a	Pro_c	Pro_g	Pro_i	Pro_p	Pro_s
Cat	0.000	0.293	0.342	0.216	0.353	0.365	0.356	0.359	0.148
Dbs	0.293	0.000	0.338	0.218	0.341	0.347	0.357	0.343	0.134
Ser	0.342	0.338	0.000	0.309	0.333	0.327	0.327	0.330	0.241
Pro_a	0.216	0.218	0.309	0.000	0.350	0.313	0.312	0.332	0.262
Pro_c	0.353	0.341	0.333	0.350	0.000	0.366	0.375	0.287	0.261
Pro_g	0.365	0.347	0.327	0.313	0.366	0.000	0.371	0.382	0.240
Pro_i	0.356	0.357	0.327	0.312	0.375	0.371	0.000	0.354	0.257
Pro_p	0.359	0.343	0.330	0.332	0.287	0.382	0.354	0.000	0.244
Pro_s	0.148	0.134	0.241	0.262	0.261	0.240	0.257	0.244	0.000

Table 13: Overall Hybrid Analysis (Fast-vs-Fast on 298 AFs, others on 65 Tweety AFs) - Kendall's Tau (τ) (Standard Deviation)

	Cat	Dbs	Ser	Pro_a	Pro_c	Pro_g	Pro_i	Pro_p	Pro_s
Cat	1.000	0.323	0.270	0.095	0.296	0.298	0.309	0.238	0.011
Dbs	0.323	1.000	0.275	0.109	0.300	0.269	0.265	0.251	0.013
Ser	0.270	0.275	1.000	0.247	0.175	0.213	0.163	0.179	0.088
Pro_a	0.095	0.109	0.247	1.000	0.228	0.204	0.226	0.237	0.134
Pro_c	0.296	0.300	0.175	0.228	1.000	0.272	0.259	0.545	0.038
Pro_g	0.298	0.269	0.213	0.204	0.272	1.000	0.307	0.315	0.029
Pro_i	0.309	0.265	0.163	0.226	0.259	0.307	1.000	0.286	0.066
Pro_p	0.238	0.251	0.179	0.237	0.545	0.315	0.286	1.000	0.056
Pro_s	0.011	0.013	0.088	0.134	0.038	0.029	0.066	0.056	1.000

Table 14: Overall Hybrid Analysis (Fast-vs-Fast on 298 AFs, others on 65 Tweety AFs) - Spearman's Rho (ρ) (Average)

	Cat	Dbs	Ser	Pro_a	Pro_c	Pro_g	Pro_i	Pro_p	Pro_s
Cat	1.000	0.219	0.191	0.032	0.223	0.204	0.198	0.090	0.001
Dbs	0.219	1.000	0.249	0.043	0.183	0.189	0.154	0.162	0.001
Ser	0.191	0.249	1.000	0.182	0.053	0.140	0.093	0.089	0.068
Pro_a	0.032	0.043	0.182	1.000	0.105	0.130	0.110	0.181	0.078
Pro_c	0.223	0.183	0.053	0.105	1.000	0.172	0.111	0.498	0.006
Pro_g	0.204	0.189	0.140	0.130	0.172	1.000	0.185	0.185	-0.006
Pro_i	0.198	0.154	0.093	0.110	0.111	0.185	1.000	0.157	0.006
Pro_p	0.090	0.162	0.089	0.181	0.498	0.185	0.157	1.000	0.011
Pro_s	0.001	0.001	0.068	0.078	0.006	-0.006	0.006	0.011	1.000

Table 15: Overall Hybrid Analysis (Fast-vs-Fast on 298 AFs, others on 65 Tweety AFs) - Spearman's Rho (ρ) (Median)

	Cat	Dbs	Ser	Pro_a	Pro_c	Pro_g	Pro_i	Pro_p	Pro_s
Cat	0.000	0.302	0.373	0.241	0.372	0.389	0.369	0.377	0.174
Dbs	0.302	0.000	0.366	0.246	0.356	0.368	0.378	0.360	0.162
Ser	0.373	0.366	0.000	0.357	0.363	0.348	0.362	0.361	0.291
Pro_a	0.241	0.246	0.357	0.000	0.393	0.342	0.336	0.374	0.311
Pro_c	0.372	0.356	0.363	0.393	0.000	0.380	0.397	0.284	0.317
Pro_g	0.389	0.368	0.348	0.342	0.380	0.000	0.384	0.402	0.278
Pro_i	0.369	0.378	0.362	0.336	0.397	0.384	0.000	0.365	0.304
Pro_p	0.377	0.360	0.361	0.374	0.284	0.402	0.365	0.000	0.293
Pro_s	0.174	0.162	0.291	0.311	0.317	0.278	0.304	0.293	0.000

Table 16: Overall Hybrid Analysis (Fast-vs-Fast on 298 AFs, others on 65 Tweety AFs) - Spearman's Rho (ρ) (Standard Deviation)

	Cat	Dbs	Ser	Pro_a	Pro_c	Pro_g	Pro_i	Pro_p	Pro_s
Cat	1.000	0.230	0.072	0.034	0.077	0.120	0.127	0.046	-0.003
Dbs	0.230	1.000	0.076	0.046	0.092	0.091	0.088	0.072	-0.000
Ser	0.072	0.076	1.000	0.105	0.042	0.120	0.043	0.055	0.061
Pro_a	0.034	0.046	0.105	1.000	0.056	0.055	0.073	0.079	0.145
Pro_c	0.077	0.092	0.042	0.056	1.000	0.093	0.055	0.417	0.009
Pro_g	0.120	0.091	0.120	0.055	0.093	1.000	0.111	0.100	0.004
Pro_i	0.127	0.088	0.043	0.073	0.055	0.111	1.000	0.078	0.028
Pro_p	0.046	0.072	0.055	0.079	0.417	0.100	0.078	1.000	0.013
Pro_s	-0.003	-0.000	0.061	0.145	0.009	0.004	0.028	0.013	1.000

Table 17: Cyclic Hybrid Analysis (Fast-vs-Fast on 275 AFs, others on 45 Tweety AFs) - Kendall's Tau (τ) (Average)

	Cat	Dbs	Ser	Pro_a	Pro_c	Pro_g	Pro_i	Pro_p	Pro_s
Cat	1.000	0.147	0.044	0.019	0.051	0.060	0.096	0.033	-0.001
Dbs	0.147	1.000	0.037	0.025	0.079	0.082	0.082	0.068	0.001
Ser	0.044	0.037	1.000	0.084	0.019	0.088	0.016	0.045	0.074
Pro_a	0.019	0.025	0.084	1.000	0.001	0.040	0.043	0.073	0.070
Pro_c	0.051	0.079	0.019	0.001	1.000	0.088	0.047	0.386	0.014
Pro_g	0.060	0.082	0.088	0.040	0.088	1.000	0.056	0.086	-0.007
Pro_i	0.096	0.082	0.016	0.043	0.047	0.056	1.000	0.060	0.022
Pro_p	0.033	0.068	0.045	0.073	0.386	0.086	0.060	1.000	0.011
Pro_s	-0.001	0.001	0.074	0.070	0.014	-0.007	0.022	0.011	1.000

Table 18: Cyclic Hybrid Analysis (Fast-vs-Fast on 275 AFs, others on 45 Tweety AFs) - Kendall's Tau (τ) (Median)

	Cat	Dbs	Ser	Pro_a	Pro_c	Pro_g	Pro_i	Pro_p	Pro_s
Cat	0.000	0.242	0.180	0.094	0.152	0.234	0.216	0.135	0.072
Dbs	0.242	0.000	0.178	0.107	0.125	0.169	0.187	0.123	0.072
Ser	0.180	0.178	0.000	0.176	0.165	0.188	0.133	0.154	0.149
Pro_a	0.094	0.107	0.176	0.000	0.221	0.136	0.141	0.193	0.241
Pro_c	0.152	0.125	0.165	0.221	0.000	0.188	0.167	0.194	0.164
Pro_g	0.234	0.169	0.188	0.136	0.188	0.000	0.208	0.204	0.127
Pro_i	0.216	0.187	0.133	0.141	0.167	0.208	0.000	0.140	0.174
Pro_p	0.135	0.123	0.154	0.193	0.194	0.204	0.140	0.000	0.130
Pro_s	0.072	0.072	0.149	0.241	0.164	0.127	0.174	0.130	0.000

Table 19: Cyclic Hybrid Analysis (Fast-vs-Fast on 275 AFs, others on 45 Tweety AFs) - Kendall's Tau (τ) (Standard Deviation)

	Cat	Dbs	Ser	Pro_a	Pro_c	Pro_g	Pro_i	Pro_p	Pro_s
Cat	1.000	0.276	0.086	0.038	0.106	0.141	0.154	0.062	-0.009
Dbs	0.276	1.000	0.089	0.054	0.127	0.118	0.115	0.095	-0.007
Ser	0.086	0.089	1.000	0.097	0.044	0.105	0.048	0.071	0.052
Pro_a	0.038	0.054	0.097	1.000	0.064	0.056	0.095	0.103	0.123
Pro_c	0.106	0.127	0.044	0.064	1.000	0.117	0.068	0.478	0.008
Pro_g	0.141	0.118	0.105	0.056	0.117	1.000	0.136	0.125	-0.011
Pro_i	0.154	0.115	0.048	0.095	0.068	0.136	1.000	0.101	0.036
Pro_p	0.062	0.095	0.071	0.103	0.478	0.125	0.101	1.000	0.020
Pro_s	-0.009	-0.007	0.052	0.123	0.008	-0.011	0.036	0.020	1.000

Table 20: Cyclic Hybrid Analysis (Fast-vs-Fast on 275 AFs, others on 45 Tweety AFs) - Spearman's Rho (ρ) (Average)

	Cat	Dbs	Ser	Pro_a	Pro_c	Pro_g	Pro_i	Pro_p	Pro_s
Cat	1.000	0.194	0.059	0.022	0.070	0.090	0.123	0.039	-0.001
Dbs	0.194	1.000	0.056	0.035	0.105	0.115	0.106	0.096	0.000
Ser	0.059	0.056	1.000	0.069	0.020	0.069	0.022	0.074	0.079
Pro_a	0.022	0.035	0.069	1.000	0.013	0.026	0.086	0.115	0.069
Pro_c	0.070	0.105	0.020	0.013	1.000	0.124	0.022	0.455	0.001
Pro_g	0.090	0.115	0.069	0.026	0.124	1.000	0.080	0.130	-0.006
Pro_i	0.123	0.106	0.022	0.086	0.022	0.080	1.000	0.103	0.033
Pro_p	0.039	0.096	0.074	0.115	0.455	0.130	0.103	1.000	0.018
Pro_s	-0.001	0.000	0.079	0.069	0.001	-0.006	0.033	0.018	1.000

Table 21: Cyclic Hybrid Analysis (Fast-vs-Fast on 275 AFs, others on 45 Tweety AFs) - Spearman's Rho (ρ) (Median)

	Cat	Dbs	Ser	Pro_a	Pro_c	Pro_g	Pro_i	Pro_p	Pro_s
Cat	0.000	0.260	0.236	0.127	0.202	0.284	0.244	0.172	0.108
Dbs	0.260	0.000	0.226	0.143	0.178	0.229	0.236	0.172	0.108
Ser	0.236	0.226	0.000	0.229	0.209	0.199	0.187	0.206	0.213
Pro_a	0.127	0.143	0.229	0.000	0.286	0.187	0.188	0.262	0.292
Pro_c	0.202	0.178	0.209	0.286	0.000	0.234	0.216	0.204	0.226
Pro_g	0.284	0.229	0.199	0.187	0.234	0.000	0.249	0.250	0.163
Pro_i	0.244	0.236	0.187	0.188	0.216	0.249	0.000	0.180	0.229
Pro_p	0.172	0.172	0.206	0.262	0.204	0.250	0.180	0.000	0.186
Pro_s	0.108	0.108	0.213	0.292	0.226	0.163	0.229	0.186	0.000

Table 22: Cyclic Hybrid Analysis (Fast-vs-Fast on 275 AFs, others on 45 Tweety AFs) - Spearman's Rho (ρ) (Standard Deviation)

	Cat	Dbs	Ser	Pro_a	Pro_c	Pro_g	Pro_i	Pro_p	Pro_s
Cat	1.000	0.862	0.637	0.713	0.663	0.607	0.611	0.599	0.252
Dbs	0.862	1.000	0.643	0.716	0.633	0.565	0.567	0.564	0.231
Ser	0.637	0.643	1.000	0.532	0.430	0.420	0.401	0.392	0.190
Pro_a	0.713	0.716	0.532	1.000	0.536	0.482	0.473	0.491	0.277
Pro_c	0.663	0.633	0.430	0.536	1.000	0.604	0.642	0.651	0.115
Pro_g	0.607	0.565	0.420	0.482	0.604	1.000	0.657	0.710	0.128
Pro_i	0.611	0.567	0.401	0.473	0.642	0.657	1.000	0.662	0.139
Pro_p	0.599	0.564	0.392	0.491	0.651	0.710	0.662	1.000	0.144
Pro_s	0.252	0.231	0.190	0.277	0.115	0.128	0.139	0.144	1.000

Table 23: Acyclic Hybrid Analysis (Fast-vs-Fast on 23 AFs, others on 20 Tweety AFs) - Kendall's Tau (τ) (Average)

	Cat	Dbs	Ser	Pro_a	Pro_c	Pro_g	Pro_i	Pro_p	Pro_s
Cat	1.000	1.000	0.535	0.814	0.706	0.591	0.620	0.646	0.059
Dbs	1.000	1.000	0.535	0.752	0.706	0.591	0.620	0.553	0.124
Ser	0.535	0.535	1.000	0.505	0.346	0.337	0.298	0.270	0.016
Pro_a	0.814	0.752	0.505	1.000	0.552	0.449	0.468	0.478	0.126
Pro_c	0.706	0.706	0.346	0.552	1.000	0.830	0.810	0.800	0.025
Pro_g	0.591	0.591	0.337	0.449	0.830	1.000	0.941	0.876	-0.011
Pro_i	0.620	0.620	0.298	0.468	0.810	0.941	1.000	0.800	-0.063
Pro_p	0.646	0.553	0.270	0.478	0.800	0.876	0.800	1.000	-0.003
Pro_s	0.059	0.124	0.016	0.126	0.025	-0.011	-0.063	-0.003	1.000

Table 24: Acyclic Hybrid Analysis (Fast-vs-Fast on 23 AFs, others on 20 Tweety AFs) - Kendall's Tau (τ) (Median)

	Cat	Dbs	Ser	Pro_a	Pro_c	Pro_g	Pro_i	Pro_p	Pro_s
Cat	0.000	0.215	0.294	0.270	0.341	0.383	0.381	0.407	0.403
Dbs	0.215	0.000	0.279	0.253	0.374	0.416	0.419	0.425	0.350
Ser	0.294	0.279	0.000	0.339	0.441	0.454	0.469	0.471	0.357
Pro_a	0.270	0.253	0.339	0.000	0.358	0.388	0.402	0.398	0.423
Pro_c	0.341	0.374	0.441	0.358	0.000	0.418	0.395	0.382	0.392
Pro_g	0.383	0.416	0.454	0.388	0.418	0.000	0.380	0.350	0.375
Pro_i	0.381	0.419	0.469	0.402	0.395	0.380	0.000	0.356	0.372
Pro_p	0.407	0.425	0.471	0.398	0.382	0.350	0.356	0.000	0.378
Pro_s	0.403	0.350	0.357	0.423	0.392	0.375	0.372	0.378	0.000

Table 25: Acyclic Hybrid Analysis (Fast-vs-Fast on 23 AFs, others on 20 Tweety AFs) - Kendall's Tau (τ) (Standard Deviation)

	Cat	Dbs	Ser	Pro_a	Pro_c	Pro_g	Pro_i	Pro_p	Pro_s
Cat	1.000	0.882	0.684	0.770	0.724	0.650	0.657	0.635	0.253
Dbs	0.882	1.000	0.695	0.769	0.691	0.608	0.604	0.601	0.244
Ser	0.684	0.695	1.000	0.583	0.470	0.455	0.423	0.423	0.167
Pro_a	0.770	0.769	0.583	1.000	0.597	0.538	0.520	0.540	0.263
Pro_c	0.724	0.691	0.470	0.597	1.000	0.621	0.687	0.695	0.105
Pro_g	0.650	0.608	0.455	0.538	0.621	1.000	0.690	0.744	0.119
Pro_i	0.657	0.604	0.423	0.520	0.687	0.690	1.000	0.703	0.133
Pro_p	0.635	0.601	0.423	0.540	0.695	0.744	0.703	1.000	0.137
Pro_s	0.253	0.244	0.167	0.263	0.105	0.119	0.133	0.137	1.000

Table 26: Acyclic Hybrid Analysis (Fast-vs-Fast on 23 AFs, others on 20 Tweety AFs) - Spearman's Rho (ρ) (Average)

	Cat	Dbs	Ser	Pro_a	Pro_c	Pro_g	Pro_i	Pro_p	Pro_s
Cat	1.000	1.000	0.647	0.898	0.850	0.757	0.821	0.802	0.075
Dbs	1.000	1.000	0.656	0.907	0.850	0.757	0.821	0.714	0.119
Ser	0.647	0.656	1.000	0.595	0.459	0.475	0.458	0.415	0.008
Pro_a	0.898	0.907	0.595	1.000	0.620	0.524	0.545	0.570	0.134
Pro_c	0.850	0.850	0.459	0.620	1.000	0.916	0.895	0.904	0.037
Pro_g	0.757	0.757	0.475	0.524	0.916	1.000	0.976	0.934	-0.038
Pro_i	0.821	0.821	0.458	0.545	0.895	0.976	1.000	0.850	-0.105
Pro_p	0.802	0.714	0.415	0.570	0.904	0.934	0.850	1.000	0.008
Pro_s	0.075	0.119	0.008	0.134	0.037	-0.038	-0.105	0.008	1.000

Table 27: Acyclic Hybrid Analysis (Fast-vs-Fast on 23 AFs, others on 20 Tweety AFs) - Spearman's Rho (ρ) (Median)

	Cat	Dbs	Ser	Pro_a	Pro_c	Pro_g	Pro_i	Pro_p	Pro_s
Cat	0.000	0.191	0.282	0.257	0.306	0.362	0.366	0.410	0.434
Dbs	0.191	0.000	0.258	0.255	0.347	0.396	0.416	0.423	0.382
Ser	0.282	0.258	0.000	0.364	0.452	0.468	0.501	0.491	0.405
Pro_a	0.257	0.255	0.364	0.000	0.349	0.375	0.403	0.408	0.464
Pro_c	0.306	0.347	0.452	0.349	0.000	0.411	0.375	0.368	0.453
Pro_g	0.362	0.396	0.468	0.375	0.411	0.000	0.358	0.348	0.423
Pro_i	0.366	0.416	0.501	0.403	0.375	0.358	0.000	0.330	0.419
Pro_p	0.410	0.423	0.491	0.408	0.368	0.348	0.330	0.000	0.438
Pro_s	0.434	0.382	0.405	0.464	0.453	0.423	0.419	0.438	0.000

Table 28: Acyclic Hybrid Analysis (Fast-vs-Fast on 23 AFs, others on 20 Tweety AFs) - Spearman's Rho (ρ) (Standard Deviation)

	Cat	Dbs	Ser	Pro_a	Pro_c	Pro_g	Pro_i	Pro_p	Pro_s
Cat	1.000	0.196	0.792	0.067	0.533	0.510	0.523	0.496	0.027
Dbs	0.196	1.000	0.789	0.072	0.518	0.501	0.523	0.493	0.024
Ser	0.792	0.789	1.000	0.566	0.413	0.399	0.445	0.377	0.272
Pro_a	0.067	0.072	0.566	1.000	0.457	0.420	0.362	0.365	0.144
Pro_c	0.533	0.518	0.413	0.457	1.000	0.539	0.433	0.581	0.137
Pro_g	0.510	0.501	0.399	0.420	0.539	1.000	0.535	0.576	0.236
Pro_i	0.523	0.523	0.445	0.362	0.433	0.535	1.000	0.533	0.237
Pro_p	0.496	0.493	0.377	0.365	0.581	0.576	0.533	1.000	0.277
Pro_s	0.027	0.024	0.272	0.144	0.137	0.236	0.237	0.277	1.000

Table 29: Sparse Hybrid Analysis (Fast-vs-Fast on 143 AFs, others on 7 Tweety AFs) - Kendall's Tau (τ) (Average)

	Cat	Dbs	Ser	Pro_a	Pro_c	Pro_g	Pro_i	Pro_p	Pro_s
Cat	1.000	0.076	0.934	0.013	0.253	0.253	0.294	0.410	0.001
Dbs	0.076	1.000	0.934	0.021	0.253	0.189	0.294	0.410	0.001
Ser	0.934	0.934	1.000	0.527	0.187	0.116	0.099	0.011	0.011
Pro_a	0.013	0.021	0.527	1.000	0.314	0.162	0.015	0.162	0.095
Pro_c	0.253	0.253	0.187	0.314	1.000	0.382	0.263	0.559	-0.132
Pro_g	0.253	0.189	0.116	0.162	0.382	1.000	0.421	0.621	-0.015
Pro_i	0.294	0.294	0.099	0.015	0.263	0.421	1.000	0.448	-0.077
Pro_p	0.410	0.410	0.011	0.162	0.559	0.621	0.448	1.000	-0.095
Pro_s	0.001	0.001	0.011	0.095	-0.132	-0.015	-0.077	-0.095	1.000

Table 30: Sparse Hybrid Analysis (Fast-vs-Fast on 143 AFs, others on 7 Tweety AFs) - Kendall's Tau (τ) (Median)

	Cat	Dbs	Ser	Pro_a	Pro_c	Pro_g	Pro_i	Pro_p	Pro_s
Cat	0.000	0.256	0.228	0.205	0.408	0.428	0.418	0.472	0.175
Dbs	0.256	0.000	0.233	0.183	0.422	0.434	0.418	0.474	0.149
Ser	0.228	0.233	0.000	0.409	0.521	0.543	0.485	0.546	0.468
Pro_a	0.205	0.183	0.409	0.000	0.374	0.396	0.441	0.461	0.232
Pro_c	0.408	0.422	0.521	0.374	0.000	0.408	0.505	0.416	0.550
Pro_g	0.428	0.434	0.543	0.396	0.408	0.000	0.418	0.452	0.497
Pro_i	0.418	0.418	0.485	0.441	0.505	0.418	0.000	0.422	0.490
Pro_p	0.472	0.474	0.546	0.461	0.416	0.452	0.422	0.000	0.488
Pro_s	0.175	0.149	0.468	0.232	0.550	0.497	0.490	0.488	0.000

Table 31: Sparse Hybrid Analysis (Fast-vs-Fast on 143 AFs, others on 7 Tweety AFs) - Kendall's Tau (τ) (Standard Deviation)

	Cat	Dbs	Ser	Pro_a	Pro_c	Pro_g	Pro_i	Pro_p	Pro_s
Cat	1.000	0.232	0.812	0.069	0.567	0.547	0.524	0.508	0.019
Dbs	0.232	1.000	0.803	0.079	0.556	0.547	0.523	0.510	0.017
Ser	0.812	0.803	1.000	0.590	0.401	0.398	0.434	0.356	0.228
Pro_a	0.069	0.079	0.590	1.000	0.497	0.459	0.368	0.376	0.086
Pro_c	0.567	0.556	0.401	0.497	1.000	0.554	0.440	0.607	0.055
Pro_g	0.547	0.547	0.398	0.459	0.554	1.000	0.567	0.586	0.204
Pro_i	0.524	0.523	0.434	0.368	0.440	0.567	1.000	0.560	0.198
Pro_p	0.508	0.510	0.356	0.376	0.607	0.586	0.560	1.000	0.250
Pro_s	0.019	0.017	0.228	0.086	0.055	0.204	0.198	0.250	1.000

Table 32: Sparse Hybrid Analysis (Fast-vs-Fast on 143 AFs, others on 7 Tweety AFs) - Spearman's Rho (ρ) (Average)

	Cat	Dbs	Ser	Pro_a	Pro_c	Pro_g	Pro_i	Pro_p	Pro_s
Cat	1.000	0.106	0.982	0.017	0.371	0.283	0.324	0.554	-0.001
Dbs	0.106	1.000	0.982	0.029	0.371	0.286	0.324	0.554	-0.001
Ser	0.982	0.982	1.000	0.635	0.319	0.187	0.093	0.009	0.008
Pro_a	0.017	0.029	0.635	1.000	0.425	0.243	0.043	0.267	0.080
Pro_c	0.371	0.371	0.319	0.425	1.000	0.404	0.311	0.586	-0.292
Pro_g	0.283	0.286	0.187	0.243	0.404	1.000	0.511	0.723	0.002
Pro_i	0.324	0.324	0.093	0.043	0.311	0.511	1.000	0.571	-0.130
Pro_p	0.554	0.554	0.009	0.267	0.586	0.723	0.571	1.000	-0.140
Pro_s	-0.001	-0.001	0.008	0.080	-0.292	0.002	-0.130	-0.140	1.000

Table 33: Sparse Hybrid Analysis (Fast-vs-Fast on 143 AFs, others on 7 Tweety AFs) - Spearman's Rho (ρ) (Median)

	Cat	Dbs	Ser	Pro_a	Pro_c	Pro_g	Pro_i	Pro_p	Pro_s
Cat	0.000	0.271	0.230	0.223	0.381	0.395	0.419	0.493	0.186
Dbs	0.271	0.000	0.239	0.204	0.392	0.395	0.419	0.492	0.162
Ser	0.230	0.239	0.000	0.453	0.551	0.559	0.495	0.569	0.505
Pro_a	0.223	0.204	0.453	0.000	0.352	0.370	0.441	0.480	0.287
Pro_c	0.381	0.392	0.551	0.352	0.000	0.395	0.502	0.411	0.603
Pro_g	0.395	0.395	0.559	0.370	0.395	0.000	0.397	0.478	0.536
Pro_i	0.419	0.419	0.495	0.441	0.502	0.397	0.000	0.412	0.525
Pro_p	0.493	0.492	0.569	0.480	0.411	0.478	0.412	0.000	0.539
Pro_s	0.186	0.162	0.505	0.287	0.603	0.536	0.525	0.539	0.000

Table 34: Sparse Hybrid Analysis (Fast-vs-Fast on 143 AFs, others on 7 Tweety AFs) - Spearman's Rho (ρ) (Standard Deviation)

	Cat	Dbs	Ser	Pro_a	Pro_c	Pro_g	Pro_i	Pro_p	Pro_s
Cat	1.000	0.321	0.052	0.030	0.049	0.063	0.111	0.016	0.007
Dbs	0.321	1.000	0.063	0.035	0.065	0.057	0.077	0.048	0.005
Ser	0.052	0.063	1.000	0.099	0.054	0.178	0.055	0.079	0.060
Pro_a	0.030	0.035	0.099	1.000	0.051	0.042	0.076	0.076	0.214
Pro_c	0.049	0.065	0.054	0.051	1.000	0.078	0.039	0.410	-0.002
Pro_g	0.063	0.057	0.178	0.042	0.078	1.000	0.074	0.072	0.016
Pro_i	0.111	0.077	0.055	0.076	0.039	0.074	1.000	0.059	0.048
Pro_p	0.016	0.048	0.079	0.076	0.410	0.072	0.059	1.000	0.001
Pro_s	0.007	0.005	0.060	0.214	-0.002	0.016	0.048	0.001	1.000

Table 35: Dense Hybrid Analysis (Fast-vs-Fast on 98 AFs, others on 30 Tweety AFs) - Kendall's Tau (τ) (Average)

	Cat	Dbs	Ser	Pro_a	Pro_c	Pro_g	Pro_i	Pro_p	Pro_s
Cat	1.000	0.256	0.040	0.037	0.022	0.043	0.102	0.010	0.018
Dbs	0.256	1.000	0.055	0.026	0.065	0.051	0.078	0.033	0.009
Ser	0.040	0.055	1.000	0.082	0.021	0.138	0.032	0.067	0.060
Pro_a	0.037	0.026	0.082	1.000	0.000	0.033	0.051	0.064	0.068
Pro_c	0.022	0.065	0.021	0.000	1.000	0.089	0.018	0.362	0.007
Pro_g	0.043	0.051	0.138	0.033	0.089	1.000	0.041	0.056	-0.010
Pro_i	0.102	0.078	0.032	0.051	0.018	0.041	1.000	0.056	0.023
Pro_p	0.010	0.033	0.067	0.064	0.362	0.056	0.056	1.000	-0.011
Pro_s	0.018	0.009	0.060	0.068	0.007	-0.010	0.023	-0.011	1.000

Table 36: Dense Hybrid Analysis (Fast-vs-Fast on 98 AFs, others on 30 Tweety AFs) - Kendall's Tau (τ) (Median)

	Cat	Dbs	Ser	Pro_a	Pro_c	Pro_g	Pro_i	Pro_p	Pro_s
Cat	0.000	0.233	0.131	0.079	0.104	0.175	0.176	0.100	0.091
Dbs	0.233	0.000	0.114	0.095	0.105	0.168	0.188	0.118	0.088
Ser	0.131	0.114	0.000	0.191	0.169	0.193	0.140	0.146	0.147
Pro_a	0.079	0.095	0.191	0.000	0.228	0.133	0.145	0.205	0.324
Pro_c	0.104	0.105	0.169	0.228	0.000	0.172	0.134	0.181	0.174
Pro_g	0.175	0.168	0.193	0.133	0.172	0.000	0.141	0.193	0.138
Pro_i	0.176	0.188	0.140	0.145	0.134	0.141	0.000	0.113	0.187
Pro_p	0.100	0.118	0.146	0.205	0.181	0.193	0.113	0.000	0.144
Pro_s	0.091	0.088	0.147	0.324	0.174	0.138	0.187	0.144	0.000

Table 37: Dense Hybrid Analysis (Fast-vs-Fast on 98 AFs, others on 30 Tweety AFs) - Kendall's Tau (τ) (Standard Deviation)

	Cat	Dbs	Ser	Pro_a	Pro_c	Pro_g	Pro_i	Pro_p	Pro_s
Cat	1.000	0.381	0.067	0.030	0.065	0.067	0.144	0.021	-0.001
Dbs	0.381	1.000	0.080	0.035	0.089	0.068	0.097	0.063	-0.006
Ser	0.067	0.080	1.000	0.089	0.057	0.156	0.062	0.105	0.056
Pro_a	0.030	0.035	0.089	1.000	0.052	0.028	0.094	0.097	0.223
Pro_c	0.065	0.089	0.057	0.052	1.000	0.094	0.046	0.460	-0.007
Pro_g	0.067	0.068	0.156	0.028	0.094	1.000	0.092	0.089	-0.005
Pro_i	0.144	0.097	0.062	0.094	0.046	0.092	1.000	0.075	0.062
Pro_p	0.021	0.063	0.105	0.097	0.460	0.089	0.075	1.000	-0.001
Pro_s	-0.001	-0.006	0.056	0.223	-0.007	-0.005	0.062	-0.001	1.000

Table 38: Dense Hybrid Analysis (Fast-vs-Fast on 98 AFs, others on 30 Tweety AFs) - Spearman's Rho (ρ) (Average)

	Cat	Dbs	Ser	Pro_a	Pro_c	Pro_g	Pro_i	Pro_p	Pro_s
Cat	1.000	0.318	0.055	0.044	0.027	0.073	0.155	0.029	0.018
Dbs	0.318	1.000	0.051	0.021	0.094	0.076	0.097	0.066	0.000
Ser	0.055	0.051	1.000	0.064	0.031	0.119	0.036	0.089	0.062
Pro_a	0.044	0.021	0.064	1.000	-0.009	0.018	0.092	0.093	0.081
Pro_c	0.027	0.094	0.031	-0.009	1.000	0.097	0.013	0.444	-0.003
Pro_g	0.073	0.076	0.119	0.018	0.097	1.000	0.068	0.091	-0.029
Pro_i	0.155	0.097	0.036	0.092	0.013	0.068	1.000	0.095	0.045
Pro_p	0.029	0.066	0.089	0.093	0.444	0.091	0.095	1.000	-0.014
Pro_s	0.018	0.000	0.062	0.081	-0.003	-0.029	0.045	-0.014	1.000

Table 39: Dense Hybrid Analysis (Fast-vs-Fast on 98 AFs, others on 30 Tweety AFs) - Spearman's Rho (ρ) (Median)

	Cat	Dbs	Ser	Pro_a	Pro_c	Pro_g	Pro_i	Pro_p	Pro_s
Cat	0.000	0.243	0.190	0.114	0.150	0.230	0.213	0.133	0.140
Dbs	0.243	0.000	0.157	0.135	0.149	0.229	0.233	0.169	0.137
Ser	0.190	0.157	0.000	0.243	0.208	0.198	0.199	0.196	0.211
Pro_a	0.114	0.135	0.243	0.000	0.289	0.182	0.190	0.282	0.356
Pro_c	0.150	0.149	0.208	0.289	0.000	0.222	0.183	0.186	0.238
Pro_g	0.230	0.229	0.198	0.182	0.222	0.000	0.189	0.238	0.171
Pro_i	0.213	0.233	0.199	0.190	0.183	0.189	0.000	0.157	0.243
Pro_p	0.133	0.169	0.196	0.282	0.186	0.238	0.157	0.000	0.205
Pro_s	0.140	0.137	0.211	0.356	0.238	0.171	0.243	0.205	0.000

Table 40: Dense Hybrid Analysis (Fast-vs-Fast on 98 AFs, others on 30 Tweety AFs) - Spearman's Rho (ρ) (Standard Deviation)