Dutch books and money pumps: rectifying vulnerabilities in LLMs through rationality

Alina Chadwick[0009−0006−3016−7663], Anson Kahng[0000−0003−1134−8400], and Jens Kipper[0000−0002−8138−0855]

University of Rochester

Abstract. It has often been claimed that a rational agent should adhere to the following two principles: 1) their judgments should be probabilistically coherent, and 2) their preferences should be transitive. A common rationale for this claim is that violations of these principles makes agents systematically exploitable by Dutch books (if 1) is violated) and money pumps (if 2) is violated). Prior research has revealed that LLMs commonly fail to adhere to 1) and 2). Simon Goldstein has recently argued that due to fundamental architectural limitations, they are in principle incapable of doing so.
Since LLMs are increasingly used as agents that perform real-world tasks (see, e.g., their use in automated negotiation), their vulnerability to Dutch books and money pumps is becoming a serious safety concern. To address this concern, we suggest that the underlying issue can be represented as an aggregation problem (of probabilistic judgments and of preferences, respectively). We therefore use methods from social epistemology and social choice theory to make LLMs resistant to adversarial attacks in the form of Dutch books and money pumps. Specifically, we introduce a dedicated rationality layer that ensures coherence through structured, explainable methods. This layer uses sampling of the LLM's underlying distribution to derive coherent probabilities and resolves intransitive preferences in an explainable manner using a hierarchical voting rule that iteratively finds maximum cuts in the pairwise majority graph.

Keywords: probabilistic coherence · transitivity of preferences · explainable AI.

1 Introduction

AI agents are becoming increasingly prevalent in decision-making systems. Agents are commonly taken to be subject to norms of rationality. Two central such norms are *probabilistic coherence* and *transitivity of preferences*. An agent is probabilistically coherent if their degrees of belief satisfy the Kolmogorov axioms of probability. Their preferences are transitive if, for any three options A, B, and C, if they prefer A to B, and they prefer B to C, then they prefer A to C.

One type of rationale for the idea that an agent needs to satisfy these norms is philosophical in nature. It has often been argued that to be an agent, one needs to at least approximately satisfy certain essential norms of rationality ([6], [5], [12], [13], [11]). This claim can be motivated by the idea that an agent's behavior must be explainable in terms of their beliefs and desires (or preferences). But this kind of explainability requires that the agent at least approximates expected preference satisfaction, judged by their belief system. Hence, a system that is not probabilistically coherent or that has intransitive preferences cannot be considered an agent.

Another reason why these norms matter is practical in nature. A system that violates them can be systematically exploited. First, a probabilistically incoherent agent is vulnerable to a *Dutch book* [17]. A Dutch book is a series of bets that the agent considers fair, based on their belief system, but that leads to a guaranteed loss. Second, an agent whose preferences are not transitive is vulnerable to a *money pump* [9]. Suppose that an agent prefers A to B and B to C, but prefers C to A, thus violating transitivity. Suppose further that they start with C. This agent is willing to trade C for B, and then B for A, and then A for C, each for a small cost. They have thus been used as a "money pump": they are back to their starting point, but have lost money (or some other value) in the process. If AI "agents" are to perform important tasks, this is a serious concern.

Today's AI agents are typically based on LLMs. LLMs have been empirically shown to violate both probabilistic coherence [23, 24, 7, 2] and transitivity of preferences [22, 20, 21]. Moreover, it has been argued

that by design, LLMs cannot be probabilistically coherent or have transitive preferences [8]. First, consider probabilistic coherence. Suppose that we ask the LLM to give a probability estimate. As a next-token predictor, it will provide this estimate on the basis of its internal token probabilities. Assume that the underlying token probabilities are what is called "semantically coherent" in [8]: the assignment of probabilities to strings of tokens consistently aligns with the assignment of probabilities to what these strings mean (e.g., physical events). If this is the case, then the probabilities assigned to strings of tokens can be interpreted as coherent subjective degrees of belief assigned to the meanings of these strings. Now assume further that the LLM greedily selects its output—i.e., it always outputs the token that is assigned the highest probability. (If greedy selection is replaced by sampling, the resulting incoherence is even worse.) Then if there is more than one string of output tokens that is assigned a non-negligible probability, one can always construct cases in which the LLM will output inconsistent probability estimates.

For instance, assume that A and B are mutually exclusive events. Asked about the probability of A, B, and $A \vee B$, the model might output '0.3', '0.3', and '0.5', respectively. These responses may reflect semantically coherent token probabilities, but the outputs selected based on these coherent probabilities are jointly incoherent.

[8] uses a similar type of argument to show that LLMs have intransitive preferences. Here, we assume that when the model is asked to give a preference ordering, the model's token probabilities conform to some distribution over preference orderings. If outputs expressing distinct intransitive preference orderings have non-negligible internal probabilities, one can construct cases in which greedy selection yields that A is preferred to B, B is preferred to C, but C is preferred to A, yielding a cycle. (If greedy selection is replaced by sampling, intransitivities become even more common.)

In both cases, the result is that the LLM is systematically exploitable, if construed as an agent: its incoherent probability estimates make it susceptible to a Dutch book, and its intransitive preferences can be used to turn it into a money pump. As AI agents are becoming more prevalent and are assigned increasingly influential tasks, this vulnerability to adversarial attacks is a growing cause of concern. (See, e.g., their use in automated negotiation [3, 10].)

In this paper, we describe a novel way to rectify these shortcomings, using methods from social epistemology and social choice theory. Specifically, we introduce a dedicated rationality layer that ensures coherence through structured, explainable methods, and thus makes LLMs immune to exploitation in the form of Dutch books or money pumps. This layer samples strings of tokens from the underlying distribution to derive coherent probabilities and resolves intransitive preferences in an explainable manner using a novel hierarchical voting rule that iteratively finds maximum cuts in the pairwise majority graph.

2 Theoretical Contributions

To rectify issues of probabilistic incoherence, we propose two general approaches. The first is based on intelligently querying, pooling, and normalizing output probabilities, and the second involves solving a quadratic program to minimally distort queried probabilities. Both result in provably coherent probability estimates when applied to a collection of mutually exclusive and collectively exhaustive outcomes. To correct intransitivities in LLM preferences, we use voting rules based on social choice theory that are guaranteed to produce transitive rankings even if given intransitive preferences. In particular, we focus on methods to approximate and explain the *Kemeny* (median) ranking that are based on pairwise comparisons of the form, "do you prefer A or B?" While we address these issues separately, we propose incorporating the solutions to both problems into one rational layer in Section 3.

2.1 Probabilistic Coherence

The source of probabilistic incoherence in LLMs identified by [8] is that when the underlying token probabilities are mapped to a probability estimate, information about the underlying probabilities is lost. This problem bears structural similarities with the "discursive dilemma" [14], which arises when individual beliefs are aggregated into a group belief. ([1] also discusses the discursive dilemma in the context of probabilistic incoherence in LLMs.) For example, it is possible that a majority of individuals believes all of $\neg A$, $\neg B$, and $A \vee B$, even if each individual's beliefs are coherent. Hence, aggregating individually consistent beliefs by

majority vote can lead to an inconsistent belief set at the group level. Similarly, aggregating semantically coherent probabilities by selection of the most likely output can lead to outputs that are jointly probabilistically incoherent.

A natural way to resolve this problem is by replacing greedy selection of outputs by pooling of the token probabilities, as follows. (Here, we propose linear pooling, but other pooling algorithms could also be used.) Suppose there is a set of m distinct, internally coherent probability distributions (hypotheses), denoted as $H_1, H_2, \ldots, H_m$. Each H_i assigns a probability $P_i(E)$ to any event E, and each $P_i(\cdot)$ function satisfies all axioms of probability.

Let us assume that the LLM (or its underlying model) is, in the terminology of [8], semantically coherent. As was mentioned above, we can then interpret its probability distribution over strings of tokens as a subjective probability distribution over states of the world—i.e., a credence function. This credence function assigns a weight or probability q_i to each hypothesis H_i, where $q_i \geqslant 0$ and $\sum_{i=1}^{m} q_i = 1$. This q_i represents the LLM's credence that H_i is the correct probabilistic model of the world. When the LLM is prompted for a probability estimate of an event E, it produces a set of possible probability statements as output tokens (e.g., "20%", "15%"). Let these statement values be $v_1, v_2, \ldots, v_k$. Crucially, the token probability that the LLM assigns to outputting a specific statement s_j (which has numerical value $v(s_j)$) is derived from the q_i values. Specifically, if a statement s_j corresponds to the probability $P_i(E)$ given by hypothesis H_i, its token probability (or a component of it) is q_i. The set of probability values the LLM considers outputting for event E are thus $\{P_1(E), P_2(E), \ldots, P_m(E)\}$. This reconstruction does not require that the LLM explicitly represents multiple hypotheses or assigns explicit weights to them. Rather, as mentioned above, we assume (following [8]) that the LLM's token probabilities are semantically coherent. Under this assumption, one can reconstruct the outputs as if they were generated by a weighted mixture over coherent hypotheses.

Now we calculate the linear pooled probability for event E, denoted as $P_{LP}(E)$, as follows:

$$P_{LP}(E) = \sum_{i=1}^{m} q_i \cdot P_i(E)$$

This is effectively the expected probability of E, according to the LLM's distribution over the coherent hypotheses. Then, rather than outputting the most likely token, we output the resulting pooled probability. This resolves the problem raised by [8], in that it guarantees that a semantically coherent LLM will output probabilistically coherent probability estimates.

However, in practice, this approach is insufficient, since extant LLMs are far from probabilistically coherent [23, 24, 7]. In our experiments, we found that not only did taking expected probabilities of events not improve coherence, but there were cases where the expected probabilities over tokens were even less coherent than the originally outputted probabilities. For example, for the query regarding the probability of event A, the LLM outputted 0.10 while $P_{LP}(A) = 0.1131$, for the probability of event B, the LLM outputted 0.05 while $P_{LP}(B) = 0.0579$, however, for the probability of event A or B, the LLM outputted 0.15 while $P_{LP}(A \cup B) = 0.1489$. In this case, the probabilities outputted by the LLM happened to be coherent to begin with, however, linear pooling over the token probabilities resulted in estimates that were no longer coherent.

We therefore propose the following approach that is designed to ensure probabilistic coherence not just in theory but also in practice. Let Ω be the sample space of interest. We first define or obtain a set of n events, $S = \{E_1, E_2, \ldots, E_n\}$, which forms a partition of Ω. This means that S is a set of collectively exhaustive and mutually exclusive events. We then ask the LLM to provide a probability estimate $P_{LLM}(E_i)$ for each event $E_i \in S$. Let this initial vector of estimates be $\mathbf{p}_{LLM} = [P_{LLM}(E_1), P_{LLM}(E_2), \ldots, P_{LLM}(E_n)]$.

To make these estimates available to post-processing by sound algorithms, we ask the LLM to translate these estimates into a formal language. Since we can expect a state-of-the-art LLM to provide probability estimates within the $[0, 1]$ range, we can ensure probabilistic coherence of the provided distribution by normalizing, thereby guaranteeing that the sum is unity. Once we have the coherent probability distribution $\mathbf{p}_{final}$ over the partition S, we can use standard probability calculus to determine the probabilities of any Boolean combinations of these events (such as disjunctions of events).

We provide some comments on this approach below:

– Our approach involves prompting the model for a distribution, rather than for probability estimates of individual events. In other contexts, it has been found that an LLM's probability estimates improve

if they are asked for a distribution [19]. Our own tests confirm this; prompting for a distribution of probabilities over all events in Ω results in outputted probability estimates that add up to around 1 (with the total probability ranging between 0.9 and 1.1 across our experiments). So this aspect of our approach is independently motivated.
- In our tests, we ask the LLM to provide the partition of Ω, S, and verify it manually. This could also be done entirely by the LLM. (See [18] for evidence that LLMs can produce coherent probability distributions.) However, in this case, one would need to ensure that the set of events obtained is in fact a partition.
- As described above, our approach does not pool token probabilities. To use these, we add the following intermediate step: Instead of prompting for just a single probability distribution, we sample many distributions from the model's internal token probabilities. We then use linear pooling to aggregate these distributions. We continue with the normalization step to ensure probabilistic coherence, as before. This intermediate step is not required to guarantee coherence, but it has the virtue of taking into account potentially valuable information contained in the model's token probabilities.

Combining the aforementioned methodology, a complete description of our approach is as follows. We begin by prompting the LLM for a distribution over the space S repeatedly (in our experiments, 100 times). For each of these outputs, we normalize the probability estimates in order to ensure coherence. Finally, we output a result from linear pooling over all 100 trials. This step is not required to ensure coherence (as each of the normalized versions of the 100 outputs is probabilistically coherent), however, pooling over 100 runs allows us to correct for some of the variance in the responses outputted by the LLM.

In the approach just described, we only elicit probability estimates for atomic propositions from the LLM. In some contexts, it may be useful to elicit probability estimates for molecular (i.e., non-atomic) propositions. For instance, instead of only querying atomic probabilities $P_{LLM}(E_i)$ for each event $E_i \in S$, we may query disjunctive combinations of events $P_{LLM}(E_i \cup E_j)$ for all pairs of events $E_i, E_j \in S$. This approach naturally extends to queries of larger size.

Because LLMs provide incoherent probability estimates among queries of different sizes—e.g., for many pairs $E_i, E_j \in S$, $P_{LLM}(E_i \cup E_j) \neq P_{LLM}(E_i) + P_{LLM}(E_j)$—naively normalizing is no longer sufficient to return a coherent probability distribution. Instead, we leverage tools from optimization to find coherent probabilities of atomic events that minimally distort the LLM's probabilities. Given probability estimates P_{LLM}, we calculate coherent (atomic) probabilities $\mathbf{p}_{final}$ by using the following quadratic program. In the following, let C represent the set of all queries to the LLM and let p be a valid probability distribution over C. For each query $q \in C$ of the form $E_i \cup \ldots \cup E_k$ and a given atomic probability distribution p, we define $p(q) := p(E_i) + \ldots + p(E_k)$. Furthermore, w_q is a nonnegative query-specific weight that determines how strongly the quadratic program considers each query.

$$\begin{aligned} \mathbf{p}_{final} = \arg\min_{p} & \sum_{q \in C} w_q (p(q) - P_{LLM}(q))^2 \\ \text{s.t.} & \sum_{E \in S} p(E) = 1, \\ & p(E) \geqslant 0, \, \forall E \in S. \end{aligned}$$

In other words, we find the coherent atomic probability distribution that minimizes weighted squared error relative to the input LLM distribution.

2.2 Transitivity of Preferences

To resolve the issue of intransitivity, we turn to *voting rules* that take as input (complete or partial) preferences and return a complete order over all alternatives. Importantly, these voting rules automatically return acyclic rankings over alternatives even when the given input preferences contain cycles. These preferences can be represented as a *tournament* or *pairwise majority graph* where an edge $A \to B$ indicates that more voters prefer alternative A over alternative B and the edge weight on $A \to B$ corresponds to the difference in the number of voters who prefer A to B over B to A. When working with LLMs, edge weights in the

tournament graph (e.g., the weight on $A \to B$) represent the difference in token probabilities when asked to choose between A and B. While there are different ways of getting a ranking from a tournament graph, we aim to identify a method that outputs a ranking that is closest to the *Kemeny ranking* (i.e., the median ranking), which we abbreviate *KR*. However, since calculating the Kemeny ranking (KR) is NP-hard, which intuitively means that some instances require exponential time to solve, we aim to develop methods that can approximate the Kemeny ranking on a potentially cyclic tournament graph.

Inspired by [15], we introduce a *hierarchical* voting rule called *Iterative Maximum Directed Cut*, or *Iterative Max Di-Cut (IMDC)*, which recursively cuts (i.e., partitions) the weighted tournament graph into disjoint sets so as to maximize the weight of directed edges crossing each cut. As a sub-procedure, we must repeatedly solve the maximum directed cut (Max Di-Cut) problem. We may represent the output of IMDC as a binary tree where the distance between alternatives represents "closeness". An example is illustrated in Figure 1.

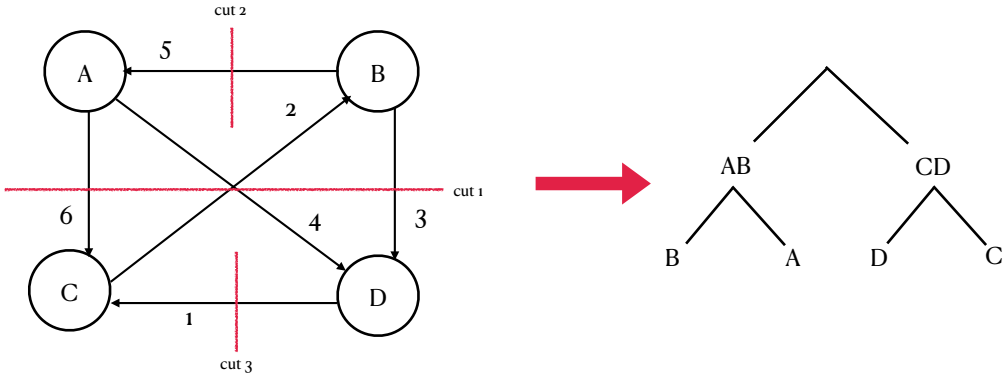


Fig. 1: An example of the IMDC procedure

Intuitively, IMDC works as follows. Starting from the complete weighted tournament graph, IMDC identifies the maximum-weight cut over all possible cuts that partition the graph into two segments. Here, a cut corresponds to a partition of the vertices into two disjoint subsets, and the weight of a cut is the sum of the directed edge weights that cross the cut. Because this is a directed graph, every maximum-weight cut imposes an order over its partitions. This process continues recursively on each segment until the graph is separated into single vertices. The order of the cuts naturally defines a binary tree in which leaf nodes correspond to vertices. We present the pseudocode in Algorithm 1.

However, solving the Max Di-Cut problem is also NP-hard, meaning it's not necessarily faster than solving for the Kemeny ranking. Because of that, beyond IMDC, we propose using IMDC to augment existing voting rules. To this end, we define two families of rules, *x-IMDC-o* and *x-IMDC-u*, where each x is an existing voting rule. The o and u designation determines whether or not the IMDC variant is allowed to change the order of alternatives from the output of rule x. Any rule ending in "-o" (for "ordered") cannot change the order and is used to generate a binary tree to explain the structure of the ranking, and any rule ending in "-u" (for "unordered") may make limited changes to the ranking returned by rule x.

The *x-IMDC-o* family of rules works by running IMDC on the weighted majority graph with the following additional constraint: All partitions must divide the output of rule x into contiguous pieces, and all cuts must be ordered in accordance with the output of rule x. Therefore, the output of *x-IMDC-o* will always exactly align with the output of rule x, and it is mostly useful as an explanatory tool that provides insight into which divisions are most meaningful.

The *x-IMDC-u* family of rules works very similarly, with one major difference: Although all partitions still must divide the output of rule x into contiguous pieces, cuts do not have to be ordered in accordance with the output of rule x. This means that the output of *x-IMDC-u* many not always exactly align with the output of rule x. In practice, we see that the output of *x-IMDC-u* is not guaranteed to be closer to the Kemeny ranking in theory, but it is often closer in practice. We use *x-IMDC-u* as a tool to both refine the ranking output by rule x and explain the structure of the ranking.

Algorithm 1 Iterative Max Di-Cut (IMDC)

```
g ← graph
first_part ← [], second_part ← []
max_cut ← None, max_weight ← 0
if len(first_part) = 1 or len(second_part) = 1 then
    return first_part + second_part
end if
for cut in all possible cuts in g do
    if |cut_weight| > max_weight then
        max_weight ← |cut_weight|, max_cut ← cut
    end if
end for
first_part, second_part = max_cut
g_1 ← g \ nodes in second_part
g_2 ← g \ nodes in first_part
IMDC(g_1, first_part)
IMDC(g_2, second_part)
return first_part + second_part
```

For example, consider *B-IMDC*, a version of IMDC that takes as input the the Borda ranking on a given tournament graph. Then, the *B-IMDC-o* ranking is the result of a procedure that takes in the Borda ranking and makes cuts that already exist in the Borda ranking (e.g., if the Borda ranking is $A \succ B \succ C \succ D$, possible cuts would be $A \gg B \succ C \succ D, A \succ B \gg C \succ D$,or $A \succ B \succ C \gg D$, where '$\gg$' denotes a cut). The pseudocode for this procedure is displayed in Algorithm 2. This procedure gives us strictly more information than the Borda ranking, as we can visualize how "far" the alternatives in the ranking are from each other and also restricts the number of possible cuts to explore in the IMDC procedure. On the other hand, the *B-IMDC-u* ranking is the result of a procedure that takes in the Borda ranking and makes cuts that already exist in the Borda ranking, except in this case, the order of the alternatives may swap if the edge weight on that cut is positive. For example, the Borda ranking might be $A \succ B \succ C \succ D$, but the B-IMDC-u ranking could output $A \succ B \succ D \succ C$ if, after cutting away A and B leaves you with one (perhaps not highly weighted) edge $D \to C$. The pseudocode for this procedure is visualized in Algorithm 3. We introduce the concepts of x-IMDC-o and x-IMDC-u for other rules, including Copeland, Plurality, and Veto (giving us C-IMDC, P-IMDC, and V-IMDC, respectively).

Finally, note that x-IMDC-o and x-IMDC-u variants run in polynomial time because only very few possible cuts must be evaluated at each stage of the process (each of the $m-1$ possible cuts in the given ranking is considered at most $m-1$ times). This means that our proposed methods are computationally feasible on real-world instances.

Our main transitivity-related findings can be summarized in Table 1. Specific theorem statements that establish structure on x-IMDC-u rules may be found below, as well as in the appendix.

Theorem 1. *Both K-IMDC-u and K-IMDC-o output the Kemeny ranking.*

Proof. The K-IMDC-o ranking is equivalent to the Kemeny ranking by definition. Therefore, we only need to prove that K-IMDC-o and K-IMDC-u are equivalent.

Consider, without loss of generality, some cut $A \succ B$, where $A \succ B$ holds in the true Kemeny ranking (and therefore also in the K-IMDC-o ranking). Assume, for the sake of contradiction, that the K-IMDC-u ranking reverses this cut (in other words, $B \succ A$ in the K-IMDC-u ranking). This would mean that the weight on $B \to A$ is positive, which would make $B \succ A$ a better candidate for the Kemeny ranking than $A \succ B$. In other words, we can derive the Kemeny ranking by reversing the lowest cost set of edges to make our graph acyclic. Then, if $B \to A$ is positive, reversing the edge to have $A \succ B$ in a potential Kemeny ranking would be a "worse" candidate for the Kemeny ranking than one that includes $B \succ A$. Therefore, $B \succ A$ as a part of a ranking would be closer to the median ranking than one containing $A \succ B$ and is therefore a "better" candidate for the Kemeny ranking, which is a contradiction.

Algorithm 2 Ordered Iterative Max Di-Cut (x-IMDC-o)

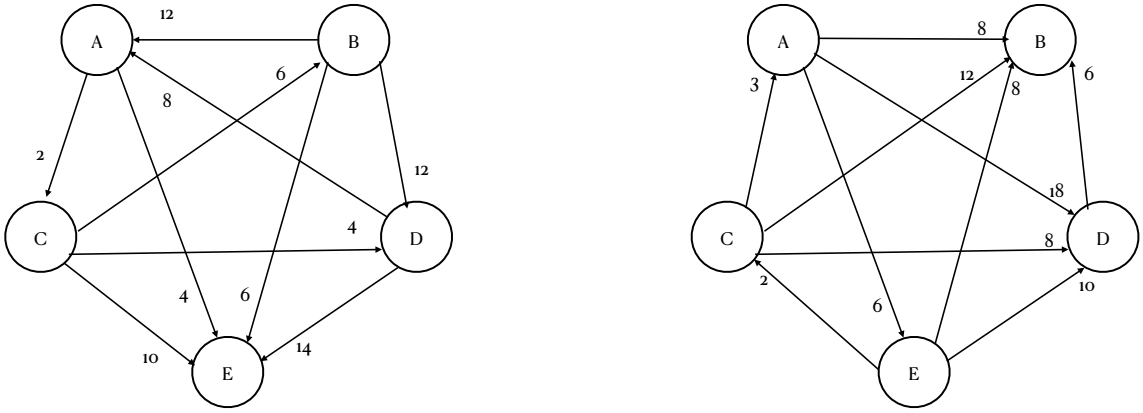
```
g ← graph, ranking ← x(g)
first_part ← [], second_part ← []
max_cut ← None, max_weight ← 0
if len(first_part) = 1 or len(second_part) = 1 then
    return first_part + second_part
end if
for cut in partitions of the given ranking do
    if |cut_weight| > max_weight then
        max_weight ← |cut_weight|, max_cut ← cut
    end if
end for
first_part, second_part = max_cut
g_1 ← g \ nodes in second_part
g_2 ← g \ nodes in first_part
IMDC(g_1, first_part)
IMDC(g_2, second_part)
return first_part + second_part
```

Algorithm 3 Unordered Iterative Max Di-Cut (x-IMDC-u)

```
g ← graph, ranking ← x(g)
first_part ← [], second_part ← []
max_cut ← None, max_weight ← 0
if len(first_part) = 1 or len(second_part) = 1 then
    return first_part + second_part
end if
for cut in partitions of the ranking do
    if |cut_weight| > max_weight then
        max_weight ← |cut_weight|
        if cut_weight > 0 then
            max_cut ← cut
        else
            first_part, second_part ← cut
            max_cut ← second_part, first_part
        end if
    end if
end for
first_part, second_part = max_cut
g_1 ← g \ nodes in second_part
g_2 ← g \ nodes in first_part
IMDC(g_1, first_part)
IMDC(g_2, second_part)
return first_part + second_part
```

	Kemeny	**Copeland**	**Borda**	**Plurality**	**Veto**
X-IMDC-u (acyclic graph)	Same as Kemeny Ranking (Theorem 1)	Same as Kemeny Ranking (Theorem 7)	Always closer to Kemeny Ranking than B-IMDC-o (Theorem 8)	Always closer to Kemeny Ranking than P-IMDC-o (Theorem 8)	Always closer to Kemeny Ranking than V-IMDC-o (Theorem 8)
X-IMDC-u (cyclic graph)	Same as Kemeny Ranking (Theorem 1)	Can be further from Kemeny Ranking than C-IMDC-o and different from all the CRs (Theorem 2)	Can be further from Kemeny Ranking than B-IMDC-o (Theorem 3)	Can be further from Kemeny Ranking than P-IMDC-o (Theorem 4)	Can be further from Kemeny Ranking than V-IMDC-o (Theorem 5)
X-IMDC-o	Same as Kemeny Ranking (by design)	Same as Copeland Ranking (by design)	Same as Borda Ranking (by design)	Same as Plurality Ranking (by design)	Same as Veto Ranking (by design)

Table 1: Comparison of x-IMDC variants across voting rules



(a) C-IMDC-u counterexample

(b) B-IMDC-u counterexample

Fig. 2: C-IMDC-u and B-IMDC-u counterexamples

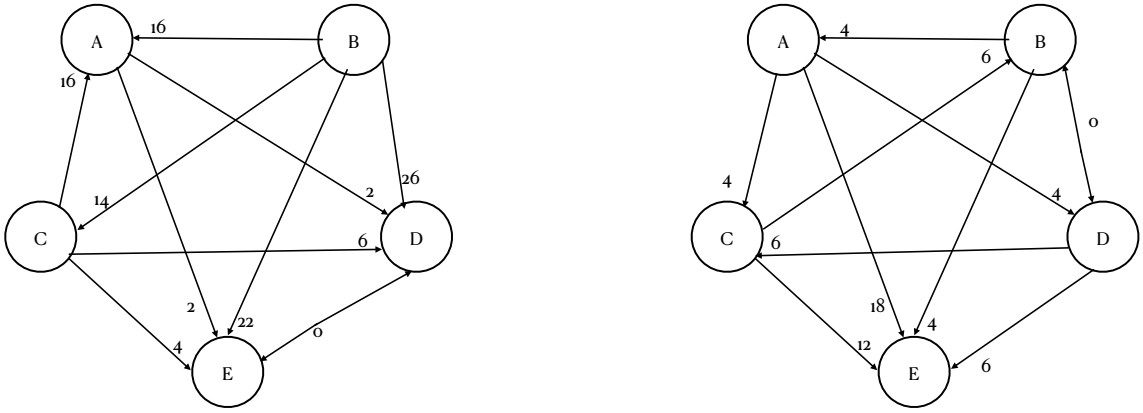
Theorem 2. *C-IMDC-u may result in a ranking that is further from the Kemeny ranking than C-IMDC-o. C-IMDC-u also may result in a ranking that is not equivalent to any of the valid Copeland rankings (i.e., Copeland rankings that result from tie-breaking).*

Proof. Consider the tournament graph in Figure 2a. In this setup, it is easy to see that the Kemeny ranking is $C \succ B \succ D \succ A \succ E$. Because this is a cyclic graph (and there are Copeland ties), there are many valid Copeland rankings that result from different tie-breaking methods. they are $B \succ C \succ A \succ D \succ E$, $B \succ C \succ D \succ A \succ E$, $C \succ B \succ A \succ D \succ E$, and $C \succ B \succ D \succ A \succ E$. However, on this tournament graph given Copeland ranking $B \succ C \succ A \succ D \succ E$, C-IMDC-u returns $B \succ D \succ A \succ C \succ E$. This is because, in the C-IMDC-u procedure, E is cut away first, then B, then in the cycle $A \to C \to D$, D is cut away from A and C, and finally A is cut from C because the edge weight on $A \to C$ is positive. Not only is this C-IMDC-u output further from the Kemeny ranking than the input Copeland ranking (equivalently, the output of C-IMDC-o), but it also is distinct from all of the valid Copeland rankings that could come from this tournament graph.

Theorem 3. *B-IMDC-u may result in a ranking that is further from the Kemeny ranking than B-IMDC-o.*

Proof. Consider the tournament graph in Figure 2b. In this setup, the Kemeny ranking is $A \succ E \succ C \succ D \succ B$. The Borda ranking (equivalently the B-IMDC-o ranking) is $A \succ C \succ E \succ D \succ B$. However, the

B-IMDC-u procedure will give us $C \succ A \succ E \succ D \succ B$ by first separating B, then D, then, in the cycle of $A \to E \to C$, E is eliminated, leaving one positive edge from $C \to A$. This is further from the Kemeny ranking than the Borda ranking, proving the claim.



(a) P-IMDC-u counterexample

(b) V-IMDC-u counterexample

Fig. 3: P-IMDC-u and V-IMDC-u counterexamples

Theorem 4. *P-IMDC-u may result in a ranking that is further from the Kemeny ranking than P-IMDC-o.*

Proof. Consider the tournament graph in Figure 3a. In this scenario, the Kemeny ranking is $B \succ C \succ A \succ D \succ E$ and the Plurality ranking (= P-IMDC-o) is $B \succ A \succ C \succ D \succ E$. However, the P-IMDC-u ranking is $B \succ C \succ D \succ E \succ A$, separating B first, then A, then C, leaving $D \to E$. This ranking is farther from the Kemeny ranking than the Plurality ranking is.

Theorem 5. *V-IMDC-u may result in a ranking that is further from the Kemeny ranking than V-IMDC-o.*

Proof. Consider the tournament graph in Figure 3b. In this scenario, the Kemeny ranking is $A \succ D \succ C \succ B \succ E$ and the Veto ranking is $C \succ A \succ D \succ B \succ E$. However, given the Veto ranking, the V-IMDC-u ranking is $B \succ A \succ D \succ C \succ E$, which is derived by first separating E, then C, then B, leaving $A \to D$. This ranking is farther from the Kemeny ranking than the Veto ranking is.

While the aforementioned results showcase that, in the worst case, x-IMDC-u can be further from the Kemeny ranking than the original rule (x) itself for Borda, Plurality, and Veto, we find that, in the average case, applying the IMDC-u procedure to the Borda and Copeland rules results in a ranking that approximates the Kemeny ranking best (better than Borda or Copeland on their own, as well as Plurality, Veto, P-IMDC-u, and V-IMDC-u). These empirical results are described in Section 4.2.

3 A Unified Rationality Layer

Here, we provide a blueprint for combining our interventions for probabilistic coherence and transitive rankings into one rational layer. This approach is applicable to any situation in which a collection of atomic, disjoint outcomes has a specific total "mass"—in the context of probability or utility, it is common to specify that the total mass is 1—and this "mass" is additive—this is the case by definition for probability, as well as for additive utility functions. Put another way, we require the model's preferences to exactly align with comparative judgments about probability or utility.

The unified approaches detailed below are meant to apply to situations in which the value of an option depends on its likelihood. This is the case, for example, in betting scenarios. These approaches are thus to be applied in situations in which preferences and likelihoods are independent. (In such cases, the model may prefer A to B but view B as more likely than A, without inconsistency.) In the latter types of situations, one may resolve issues of probabilistic incoherence and intransitivity separately by running each intervention on its own. As a running example, let us assume that we are in a setting where an agent must determine nonnegative utilities for each atomic outcome in a given set, and the sum of utilities of all outcomes must be 1.

We propose two approaches based on applying our interventions in series. Broadly, the first can be thought of as first applying our intervention for transitivity, and then our intervention for probabilistic coherence, and the second can be interpreted as just using our approach for probabilistic coherence (as transitivity follows immediately). One benefit of the first approach is that it intuitively refines cardinal utilities in two stages: The first stage (transitivity) determines the ordinal ranking of all alternatives by decreasing utility, and the second stage (probabilistic coherence) then assigns probabilities that are consistent with the transitive ranking. The second approach directly solves for cardinal utilities; because they are coherent, they immediately define a transitive ordering.

Approach 1: Transitive preferences → probabilistic coherence. First, we run our intervention to obtain transitive preferences. Specifically, the LLM queries all pairwise comparisons between alternatives, and it interprets token probabilities as edge weights in a weighted majority graph. For instance, if when the LLM is asked to choose between option A and option B it outputs token A with probability 0.8 and token B with probability 0.2, this corresponds to an edge from A to B with weight $0.8 - 0.2 = 0.6$. Then, we may use any voting rule that works on weighted majority graphs to find a transitive ranking. One natural choice would be the Kemeny rule, as it returns the median ranking, but if this is too computationally expensive, another rule such as Borda or Copeland (or even IMDC-refinements of Borda and Copeland such as B-IMDC-u or C-IMDC-u) may be used. After this step, we are left with a coherent ranking σ over all alternatives.

The second step is to run our quadratic program for probabilistic coherence with the added constraint that all utilities must respect σ, i.e., $u(\sigma_i) \geqslant u(\sigma_{i+1})$ for all $i \in \{1, \ldots, m-1\}$, where σ_i is the i^{th} element in σ and there are m total elements in σ. In particular, the quadratic program is the following.

$$\begin{aligned} \mathbf{u}_{final} = \arg\min_{u} & \sum_{x \in C} w_x (u(x) - U_{LLM}(x))^2 \\ \text{s.t.} & \sum_{x \in S} u(x) = 1, \\ & u(x) \geqslant 0, \, \forall x \in S, \text{ and} \\ & u(x) \geqslant u(y), \, \forall x \succ_\sigma y. \end{aligned}$$

Here, U_{LLM} is the utility distribution produced by the LLM, w_x is the option-specific weight for each option $x \in S$, $u(x)$ is the utility of option x, and $x \succ_\sigma y$ means that x is preferred to y in ranking σ.

Approach 2: Probabilistic coherence $\implies$ transitive preferences. Alternatively, it is possible to simply run our approach for probabilistic coherence for the setting of utilities. Because the output is probabilistically coherent, this means that all utilities will be nonnegative and collectively sum to 1. Additionally, ordering the alternatives by decreasing (in the case of ties, non-increasing) utility leads to a transitive ranking.

4 Empirical Results

In this section, we implement the approaches described in Section 2. We used GPT-4o for all of our experiments. As a state-of-the-art model, it should serve as an appropriate testbed. To ensure probabilistic coherence, we implement the normalized approach and the quadratic program, and to ensure transitivity, we implement the x-IMDC-u procedures.

4.1 Probabilistic Coherence

In our first set of experiments, we implement the procedures described in Section 2. Specifically, we first prompt the LLM for a distribution over the space S 100 times, normalizing each output to ensure coherence.

For the sake of this experiment, our sample space S consists of ten contenders for the 2026 FIFA men's World Cup (United States, Argentina, France, Brazil, England, Germany, Morocco, Netherlands, Portugal, Spain)—picked by GPT-4o—and an "Other" category. We ask for probability estimates for each of these countries winning the 2026 World Cup. Table 2a displays an example of one of the outputs and the corresponding normalized probability values.

Country	Raw Probability (%)	Normalized Probability
France	0.20	0.19048
Argentina	0.15	0.14286
Brazil	0.15	0.14286
United States	0.10	0.09524
England	0.10	0.09524
Germany	0.10	0.09524
Spain	0.05	0.04762
Morocco	0.05	0.04762
Netherlands	0.05	0.04762
Portugal	0.05	0.04762
Other	0.05	0.04762

(a) Comparison of raw and normalized probabilities

Country	Probability
France	0.18952
Argentina	0.14295
Brazil	0.14282
United States	0.09530
England	0.09530
Germany	0.09503
Spain	0.04806
Netherlands	0.04792
Portugal	0.04779
Other	0.04765
Morocco	0.04765

(b) Normalized expected probabilities

Table 2: Coherent probabilities for 2026 World Cup winner predictions

After prompting and normalizing 100 times, we performed linear pooling over all 100 trials. These values are, by design, coherent. We can calculate disjoint probabilities (e.g., the probability that either Morocco or the Netherlands win the 2026 FIFA World Cup) naively from the atomic values we get from running this procedure. Table 2b displays the results from linear pooling (calculating expected values) over probability estimates for the 100 trials.

In our second set of experiments, we implement the quadratic program described in Sections 2 and 3. For these experiments, we use the same 2026 FIFA World Cup setting, but expand our contenders list to the following options: Australia, Austria, Belgium, Bolivia, Croatia, Denmark, Ecuador, Egypt,Germany, Iran, Iraq, Italy, Jamaica, Japan, Mexico, Nigeria, Paraguay, Peru, Poland, Portugal, Qatar, Senegal, Solomon Islands, South Korea, Switzerland, Tunisia, United Arab Emirates. Probability estimates for each of these countries (e.g., the probability that Paraguay wins the 2026 FIFA World Cup) as well as estimates for all disjunctive pairs of countries (e.g., the probability that Poland or the UAE wins the 2026 FIFA World Cup) are included in the quadratic program (i.e., $\mathbf{p}_{final}$). The raw atomic probabilities outputted from the LLM as well as our new estimates found by the quadratic program are illustrated in Table 3. Disjunctive probabilities are also calculated in this procedure, but are omitted from the table due to space constraints. By design, the probabilities outputted by the quadratic program sum to 1, whereas the raw probabilities sum to 2.121 in this case, indicating that our approach solves this issue of incoherence over the sample space S.

One final set of experiments we performed involves "unleashing" the LLM to output its entire chain of thought, rather than outputting a single probability. Our rationale behind this is that, by strong-arming the LLM to only output a single number as the probability estimate, we may be constraining it too much and, perhaps, if we unleash the LLM to have no bounds on its output, the outputted estimates might be superior. In our experiments, we found that "unleashing" the LLM did not result in more coherent probability estimates than in the constrained version.

4.2 Transitivity

To complement our theoretical results from the previous section, we aim to test how the various voting rules we study as well as their x-IMDC-u counterparts perform on synthetic data. We generate three series of experiments (varying the number of voters, $n = 10$, $n = 25$, $n = 50$) testing our voting rules on different

Country	**Raw Probability**	**QP Probability**
Australia	0.15	0.0534
Austria	0.05	0.0000
Belgium	0.15	0.0716
Bolivia	0.02	0.0000
Croatia	0.15	0.0555
Denmark	0.15	0.0640
Ecuador	0.05	0.0000
Egypt	0.05	0.0049
Germany	0.15	0.1330
Iran	0.05	0.0116
Iraq	0.01	0.0000
Italy	0.25	0.1670
Jamaica	0.05	0.0000
Japan	0.15	0.0716
Mexico	0.10	0.0806
Nigeria	0.05	0.0156
Paraguay	0.02	0.0000
Peru	0.05	0.0325
Poland	0.05	0.0431
Portugal	0.15	0.0976
Qatar	0.05	0.0095
Senegal	0.05	0.0193
Solomon Islands	0.001	0.0000
South Korea	0.05	0.0216
Switzerland	0.05	0.0211
Tunisia	0.02	0.0147
United Arab Emirates	0.05	0.0116

Table 3: Raw vs. Quadratic Program Probabilities

synthetic elections generated from statistical cultures using the map of elections data [16, 4]. To generate the synthetic elections, we use the mapof-elections python library, which allows us to generate votes from different statistical cultures. These include the Impartial Culture model, the Mallows model (with dispersion parameter $\phi =\in \{0.0, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9\}$), the single-peaked Conitzer and Walsh models, and the Polya-Eggenberger urn models (with $\alpha \in \{0.01, 0.02, 0.05, 0.10\}$. Each cell in Tables 4, 5, and 6 corresponds to the mean Kendall tau difference $\pm$ the standard deviation.

The results, for varying levels of n are displayed in Tables 4, 5, and 6 (where Tables 5 and 6 can be found in the Appendix). For each statistical culture in each table, we calculated the average *Kendall-tau* distance between the ranking outputted by each voting rule (including traditional voting rules (*Borda, Copeland, Plurality, and Veto*) and our non-traditional x-IMDC-u versions of each voting rule (*B-IMDC-u, C-IMDC-u, P-IMDC-u, V-IMDC-u*) and the Kemeny ranking over 1000 runs.

Throughout the experiments, C-IMDC-u and B-IMDC-u consistently outperformed the other voting rules, with Borda and Copeland performing well at times. Between C-IMDC-u and B-IMDC-u, B-IMDC-u often outperformed C-IMDC-u under Mallows distributions of error, whereas C-IMDC-u often outperformed B-IMDC-u under Urn distributions of error. Additionally, for each traditional voting rule, the x-IMDC-u version of that voting rule outperformed the voting rule itself in terms of closeness to the Kemeny Ranking; Across the board, Plurality performed fairly poorly, however running P-IMDC-u was able to recover some of that Kendall-tau distance.

This result suggests that, in practice, running the IMDC-u procedure on any of the voting rules we study will result in a ranking closer to the Kemeny ranking, and specifically running B-IMDC-u or C-IMDC-u can approximate Kemeny in polynomial time.

Distribution	Borda	B-IMDC-u	Copeland	C-IMDC-u	Plurality	P-IMDC-u	Veto	V-IMDC-u
Impartial Culture	1.1130 ± 0.9956	**0.6720**± 0.9536	0.9410 ± 1.0491	0.7340 ± 0.9799	2.7150 ± 1.6890	0.9850 ± 1.0770	2.8380 ± 1.7535	1.0080 ± 1.1512
Mallows (0.0)	0.0000 ± 0.0000	0.0000 ± 0.0000	0.0000 ± 0.0000	0.0000 ± 0.0000	0.0000 ± 0.0000	0.0000 ± 0.0000	0.0000 ± 0.0000	0.0000 ± 0.0000
Mallows (0.1)	**0.0000**± 0.0000	**0.0000**± 0.0000	**0.0000**± 0.0000	**0.0000**± 0.0000	0.0480 ± 0.2275	**0.0000**± 0.0000	0.0300 ± 0.1764	**0.0000**± 0.0000
Mallows (0.2)	**0.0000**± 0.0000	**0.0000**± 0.0000	**0.0000**± 0.0000	**0.0000**± 0.0000	0.1680 ± 0.4242	0.0060 ± 0.1094	0.1340 ± 0.3771	**0.0000**± 0.0000
Mallows (0.3)	0.0010 ± 0.0316	**0.0000**± 0.0000	**0.0000**± 0.0000	**0.0000**± 0.0000	0.2030 ± 0.4359	0.0070 ± 0.1139	0.2180 ± 0.4344	**0.0000**± 0.0000
Mallows (0.4)	0.0080 ± 0.0891	0.0040 ± 0.0632	0.0040 ± 0.0632	0.0040 ± 0.0632	0.2760 ± 0.4961	0.0110 ± 0.1375	0.2530 ± 0.4767	**0.0030**± 0.0547
Mallows (0.5)	0.0540 ± 0.2348	0.0160 ± 0.1255	**0.0220**± 0.1534	0.0230 ± 0.1565	0.3670 ± 0.5733	0.0360 ± 0.2253	0.3470 ± 0.5718	0.0150 ± 0.1216
Mallows (0.6)	0.1450 ± 0.3607	**0.0540**± 0.2261	0.0810 ± 0.2766	0.0790 ± 0.2736	0.5410 ± 0.6533	0.1120 ± 0.3515	0.5880 ± 0.7034	0.0700 ± 0.2668
Mallows (0.7)	0.3530 ± 0.5538	**0.1590**± 0.3948	0.2130 ± 0.4380	0.2110 ± 0.4344	0.8990 ± 0.8841	0.2810 ± 0.5625	0.8900 ± 0.8802	0.2410 ± 0.5012
Mallows (0.8)	0.6410 ± 0.7229	**0.3520**± 0.6054	0.4290 ± 0.6583	0.3970 ± 0.6114	1.4370 ± 1.1736	0.5340 ± 0.7997	1.4350 ± 1.1615	0.5180 ± 0.7498
Mallows (0.9)	0.9830 ± 0.8692	**0.5970**± 0.8493	0.7810 ± 0.8865	0.6400 ± 0.8373	2.3000 ± 1.5920	0.8970 ± 0.9906	2.2640 ± 1.5501	0.8780 ± 1.0306
Conitzer	0.4850 ± 0.6100	**0.3200**± 0.5137	0.3930 ± 0.5629	0.3800 ± 0.5458	4.2360 ± 1.9840	0.7400 ± 0.8167	1.8630 ± 0.8608	0.3430 ± 0.5269
Walsh	0.2640 ± 0.4884	**0.1660**± 0.3906	0.2260 ± 0.4462	0.2160 ± 0.4284	0.9820 ± 0.8187	0.2470 ± 0.4628	1.6580 ± 0.6094	0.1860 ± 0.4142
Urn (0.01)	1.0690 ± 0.9619	**0.6420**± 0.9883	0.8550 ± 0.9843	0.6630 ± 0.9308	2.8000 ± 1.7881	1.0200 ± 1.1202	2.7340 ± 1.6841	0.9450 ± 1.0896
Urn (0.02)	1.0840 ± 0.9412	**0.6350**± 0.9352	0.8630 ± 1.0263	0.6730 ± 0.9638	2.7410 ± 1.6470	0.9790 ± 1.1162	2.6830 ± 1.6140	0.9450 ± 1.1222
Urn (0.05)	1.0270 ± 0.9269	**0.5030**± 0.8418	0.7610 ± 1.0493	0.5460 ± 0.9234	2.7310 ± 1.6166	0.8660 ± 1.0474	2.7370 ± 1.6200	0.8570 ± 1.0496
Urn (0.10)	0.9540 ± 0.8536	0.4910 ± 0.8513	0.6540 ± 0.9996	**0.4480**± 0.8087	2.7170 ± 1.6241	0.8070 ± 1.0540	2.8150 ± 1.6660	0.8590 ± 1.0697

Table 4: Mean ± standard deviation of distance to Kemeny output for different voting rules and distributions (n = 50)

5 Future work

Our approach ensures both probabilistic coherence and transitivity of preferences, at least if the event spaces and their Boolean combinations considered are not too complex to be computationally tractable. Our approach also ensures that the probability estimates and preference rankings determined are sufficiently faithful to those provided by the LLM (or by its underlying token probabilities). However, while (approximate) probabilistic coherence and transitivity of preferences are necessary conditions for rationality, they are arguably not sufficient.

For example, a model's probability estimates may be coherent but still fail to be calibrated or accurate. In future work, it will therefore be crucial to measure (perhaps through testing in games) how our approach affects an LLM's accuracy and calibration, and potentially to implement measures for improving these. For example, other pooling algorithms that maintain coherence may turn out to improve accuracy. Additionally, with the release of *Deep Reasoning* versions of many LLM models, it is possible that pooling algorithms over estimates outputted by those models are more coherent and accurate than our estimates on non-reasoning models, such as GPT-4o.

On the side of a model's preference rankings, an important open challenge is to align these with human preferences. The transparency of our approach makes it suitable for targeted interventions to ensure alignment, but this, too, will be left to future work.

Additionally, in terms of combining our probabilistic coherence and transitivity approaches in one rational layer, an immediate avenue for future work involves investigating the following question: does a two-step refinement (first ordinal, then cardinal) outperform a single probability-based approach? Further, exploring which voting rules do well as a first (ordinal) step is a direction to be explored; Kemeny may not always be the right benchmark to shoot for.

As we demonstrated, our approach can in principle resolve a model's vulnerability to adversarial attacks through Dutch books and money pumps. However, it has yet to be determined how useful our system is in real-world scenarios, such as automated negotiations.

A Appendix

A.1 Deferred Proofs

The following proofs detail theoretical results relating to IMDC on a directed acyclic graph (DAG). These results are omitted from the main body of the paper, as our application space assumes that LLMs have intransitive (i.e., cyclic) preferences to begin with.

Theorem 6 (IMDC on an Directed Acyclic Graph). *On an acyclic tournament graph, iterative max di-cut (IMDC) will always return the same ranking as the Kemeny ranking.*

Proof. To prove the above theorem, we must first prove the following claim:

Lemma 1. *In the iterative max di-cut procedure, each edge is only cut once.*

This lemma follows by design. At any stage in our iterative max di-cut procedure, we separate the current tournament graph or subgraph into two parts, say A and B, cutting some edges, say set $\hat{E}$. In the next iteration, we iterate on the edges in A and the edges in B, and never touch the edges $\hat{E}$ again. Therefore, the claim follows by design.

We now aim to prove the following lemma:

Lemma 2. *In our iterative max di-cut procedure on an directed acyclic graph, we use each edge in a "positive" way.*

Proof. At every step of the iterative max di-cut procedure on an directed acyclic graph, there exists a cut that uses every edge positively. This follows from the nature of a directed acyclic graph (i.e., there is always one or more alternative that beats the other(s) in an acyclic tournament graph). As a result, it will never be optimal to reverse any edges (i.e., use any edges in a "negative" way), as it was always yield a higher max cut to use the edges positively.

From these two claims, it follows that the IMDC procedure on an acyclic tournament graph yields the same ranking as the Kemeny ranking. On a DAG, computing the Kemeny ranking is trivial; There is always one alternative that beats the rest, then one clear second-place alternative, and so on. IMDC on a DAG preserves the same ordering since every edge is used positively (i.e., no edges are reversed) and no edge is cut more than once.

Corollary 1 (K-IMDC on an Acyclic Graph). *On an acyclic tournament graph, Kemeny iterative max di-cut (K-IMDC) will always return the same ranking as the Kemeny ranking.*

Proof. K-IMDC is a restricted version of IMDC. By that nature, since IMDC returns the same ranking as the Kemeny ranking on a directed tournament graph, as does K-IMDC. Additionally, K-IMDC is restricted to only making cuts corresponding to separations in the Kemeny ranking, therefore it should never output any ranking besides the Kemeny ranking.

Theorem 7 (C-IMDC-u on a DAG). *On an acyclic tournament graph (a DAG), the unordered version of the Copeland-IMDC rule (C-IMDC-u) outputs the Kemeny ranking.*

Proof. On a DAG, the Copeland ranking and the Kemeny ranking are the same (i.e., there is one Condorcet-consistent ranking, which is outputted by both the Copeland rule and the Kemeny rule). Trivially, as a result, C-IMDC-o = CR = KR. However, C-IMDC-u also outputs the Kemeny ranking. A similar proof to that of Theorem 2 can be constructed. Consider, without loss of generality, some cut $A \succ B$ where $A \succ B$ holds in the true Copeland (and Kemeny) ranking. This implies that A beats B in the tournament graph and the weight on $A \to B$ is positive. Suppose, for the sake of contradiction, that the C-IMDC-u process reverses this cut (i.e., $B \succ A$ is in the C-IMDC-u ranking). Given that our graph is acyclic, the only way to obtain such a ranking would be if the direction of the edge would be $B \to A$, which is a contradiction.

Theorem 8 (IMDC-u for positional scoring rules on an acyclic graph). *The unordered version of IMDC of any positional scoring rule on a DAG will, if it is not the same as the original positional scoring rule ranking, always get closer to the Kemeny ranking than the original positional scoring rule ranking.*

Proof. On a directed acyclic graph, there exists one true Kemeny ranking. Once given a ranking outputted by a positional scoring rule (e.g., Borda, Veto, or Plurality), the x-IMDC-u procedure will take in that ranking and perform iterative max di-cut given that ranking, potentially swapping alternatives or groups of alternatives if the edge weight across the cut is positive. In the worst case, no swaps will be made, and the outputted x-IMDC-u ranking will be the same as the ranking outputted by that voting rule. However, if any swaps are made during cut χ, that is because the edge weight across that cut is positive, while the original ranking used that same weight in a "negative" way. Therefore, between the x-IMDC-u ranking and the original PSR ranking, the total weight of the edges flipped to the true Kemeny ranking is reduced. As a result, the x-IMDC-u ranking will be closer to the Kemeny ranking than the original ranking outputted by the positional scoring rule or equal to that original input ranking.

A.2 Voter Profiles

For each of the counterexamples detailed in Section 2, we provide the entire voter profiles that result in each of the tournament graphs.

Theorem 2. *C-IMDC-u may result in a ranking that is further from the Kemeny ranking than C-IMDC-o. C-IMDC-u also may result in a ranking that is not equivalent to any of the valid Copeland rankings (i.e., Copeland rankings that result from tie-breaking).*

Voter profile: Suppose you have 30 voters where 8 of the voters have the ranking $B \succ C \succ D \succ A \succ E$, 7 voters have the ranking $D \succ C \succ B \succ A \succ E$, 6 voters have the ranking $C \succ D \succ A \succ B \succ E$, 5 voters have the ranking $B \succ A \succ C \succ D \succ E$, and 4 voters have the ranking $D \succ A \succ C \succ B \succ E$. This voter profile results in the tournament graph pictured in the proof of Theorem 2.

Theorem 3. *B-IMDC-u may result in a ranking that is further from the Kemeny ranking than B-IMDC-o.*

Voter profile: Suppose you have 34 voters where 10 of the voters have the ranking $C \succ A \succ E \succ D \succ B$, 9 voters have the ranking $A \succ E \succ D \succ C \succ B$, 7 voters have the ranking $E \succ C \succ A \succ D \succ B$, 5 voters have the ranking $C \succ E \succ A \succ D \succ B$, and 3 voters have the ranking $E \succ D \succ C \succ A \succ B$. This voter profile results in the tournament graph pictured in the proof of Theorem 3.

Theorem 4. *P-IMDC-u may result in a ranking that is further from the Kemeny ranking than P-IMDC-o.*

Voter profile: Suppose you have 44 voters where 14 of the voters have the ranking $B \succ A \succ C \succ D \succ E$, 12 voters have the ranking $A \succ C \succ B \succ D \succ E$, 10 voters have the ranking $C \succ B \succ A \succ E \succ D$, 4 voters have the ranking $D \succ A \succ C \succ B \succ E$, and 4 voters have the ranking $E \succ C \succ A \succ B \succ D$. This voter profile results in the tournament graph pictured in the proof of Theorem 4 and ensures that the Plurality ranking is $B \succ A \succ C \succ D \succ E$.

Theorem 5. *V-IMDC-u may result in a ranking that is further from the Kemeny ranking than V-IMDC-o.*

Voter profile: Suppose you have 32 voters where 8 of the voters have the ranking $A \succ D \succ B \succ C \succ E$, 7 voters have the ranking $D \succ C \succ A \succ B \succ E$, 6 voters have the ranking $C \succ A \succ D \succ E \succ B$, 5 voters have the ranking $B \succ C \succ A \succ E \succ D$, 4 voters have the ranking $A \succ B \succ D \succ E \succ C$, and 3 voters have the ranking $C \succ D \succ B \succ E \succ A$. This voter profile results in the tournament graph pictured in the proof of Theorem 5 and ensures that the Veto ranking is $C \succ A \succ D \succ B \succ E$.

A.3 Additional Empirical Results

Our additional results transitivity-related results for $n = 5$ and $n = 25$ can be found in Tables 5 and 6 below.

A.4 Code Base

The code used to run simulations for this paper can be found at the following link:
https://github.com/alinachadwick/Coherence-Transitivity.

Distribution	Borda	B-IMDC-u	Copeland	C-IMDC-u	Plurality	P-IMDC-u	Veto	V-IMDC-u
Impartial Culture	1.3420 ± 1.0120	**1.0380**± 1.0419	1.2820 ± 1.0390	1.1600 ± 1.0399	2.9700 ± 1.7510	1.3520 ± 1.2416	3.0190 ± 1.7128	1.3340 ± 1.2074
Mallows (0.0)	0.0000 ± 0.0000	0.0000 ± 0.0000	0.0000 ± 0.0000	0.0000 ± 0.0000	0.0000 ± 0.0000	0.0000 ± 0.0000	0.0000 ± 0.0000	0.0000 ± 0.0000
Mallows (0.1)	**0.0040**± 0.0632	**0.0040**± 0.0632	0.0050 ± 0.0706	0.0050 ± 0.0706	0.0580 ± 0.2504	0.0050 ± 0.0706	0.0640 ± 0.2683	0.0050 ± 0.0706
Mallows (0.2)	0.0500 ± 0.2270	**0.0390**± 0.1988	0.0550 ± 0.2367	0.0550 ± 0.2367	0.1880 ± 0.4635	0.0500 ± 0.2226	0.2000 ± 0.4881	0.0500 ± 0.2226
Mallows (0.3)	0.1430 ± 0.3531	**0.1120**± 0.3187	0.1860 ± 0.3969	0.1860 ± 0.3969	0.4520 ± 0.6871	0.1550 ± 0.3783	0.4240 ± 0.6532	0.1540 ± 0.3827
Mallows (0.4)	0.3110 ± 0.5201	**0.2430**± 0.4777	0.3760 ± 0.5665	0.3740 ± 0.5607	0.7650 ± 0.8512	0.3260 ± 0.5586	0.7800 ± 0.8464	0.3200 ± 0.5458
Mallows (0.5)	0.5420 ± 0.6486	**0.4270**± 0.6009	0.5720 ± 0.6381	0.5740 ± 0.6457	1.0970 ± 0.9447	0.5340 ± 0.6836	1.0470 ± 0.9527	0.5030 ± 0.6830
Mallows (0.6)	0.7320 ± 0.7660	**0.6010**± 0.7566	0.7750 ± 0.7855	0.7660 ± 0.7848	1.4860 ± 1.1820	0.7470 ± 0.8401	1.4590 ± 1.2304	0.7010 ± 0.8697
Mallows (0.7)	0.9680 ± 0.8659	**0.8140**± 0.8753	0.9680 ± 0.8909	0.9470 ± 0.8927	1.8820 ± 1.3498	0.9870 ± 0.9827	2.0040 ± 1.3849	0.9970 ± 1.0574
Mallows (0.8)	1.2080 ± 0.9714	**0.9580**± 0.9429	1.1350 ± 0.9485	1.0530 ± 0.9365	2.4170 ± 1.5210	1.2350 ± 1.1363	2.3570 ± 1.5849	1.1320 ± 1.1126
Mallows (0.9)	1.3240 ± 1.0402	**1.1110**± 1.0497	1.2480 ± 1.0419	1.1790 ± 1.0276	2.7630 ± 1.6660	1.2940 ± 1.2349	2.7600 ± 1.6508	1.3300 ± 1.1741
Conitzer	0.2790 ± 0.5152	**0.1980**± 0.4370	0.2020 ± 0.4398	0.2340 ± 0.4706	4.5050 ± 2.1020	0.5540 ± 0.7167	1.6860 ± 0.6913	0.2410 ± 0.4724
Walsh	0.2890 ± 0.5136	**0.1540**± 0.3904	0.1720 ± 0.4057	0.1880 ± 0.4276	0.7640 ± 0.7476	0.2740 ± 0.5303	1.6010 ± 0.5640	0.2530 ± 0.4912
Urn (0.01)	0.7660 ± 0.8956	0.3280 ± 0.6862	0.3260 ± 0.7902	**0.2590**± 0.6815	2.8860 ± 1.5477	0.6380 ± 0.9391	2.8920 ± 1.5744	0.6100 ± 0.8824
Urn (0.02)	0.8050 ± 0.8898	0.3340 ± 0.6759	0.3850 ± 0.8942	**0.2860**± 0.7174	2.8930 ± 1.5414	0.6530 ± 0.9600	2.8400 ± 1.5668	0.6100 ± 0.9147
Urn (0.05)	0.7540 ± 0.9189	0.3240 ± 0.7208	0.3540 ± 0.8254	**0.2600**± 0.6978	2.8000 ± 1.5279	0.6050 ± 0.9497	2.8230 ± 1.5271	0.5970 ± 0.8805
Urn (0.10)	0.7730 ± 0.9101	0.3030 ± 0.6369	0.3660 ± 0.8619	**0.2460**± 0.6587	2.7780 ± 1.5752	0.6620 ± 1.0123	2.8510 ± 1.5567	0.6440 ± 0.9467

Table 5: Mean ± standard deviation of distance to Kemeny output for different voting rules and distributions (n = 10)

Distribution	Borda	B-IMDC-u	Copeland	C-IMDC-u	Plurality	P-IMDC-u	Veto	V-IMDC-u
Impartial Culture	1.0810 ± 0.9588	0.5060 ± 0.8859	0.8250 ± 1.1356	**0.4950**± 0.9513	2.6850 ± 1.6150	0.8780 ± 1.0706	2.6930 ± 1.6654	0.9000 ± 1.0896
Mallows (0.0)	0.0000 ± 0.0000	0.0000 ± 0.0000	0.0000 ± 0.0000	0.0000 ± 0.0000	0.0000 ± 0.0000	0.0000 ± 0.0000	0.0000 ± 0.0000	0.0000 ± 0.0000
Mallows (0.1)	**0.0000**± 0.0000	**0.0000**± 0.0000	**0.0000**± 0.0000	**0.0000**± 0.0000	0.0510 ± 0.2578	**0.0000**± 0.0000	0.0570 ± 0.2565	**0.0000**± 0.0000
Mallows (0.2)	0.0020 ± 0.0447	**0.0000**± 0.0000	**0.0000**± 0.0000	**0.0000**± 0.0000	0.1750 ± 0.4611	0.0010 ± 0.0316	0.1690 ± 0.4366	0.0010 ± 0.0316
Mallows (0.3)	0.0200 ± 0.1401	**0.0000**± 0.0000	**0.0000**± 0.0000	**0.0000**± 0.0000	0.2580 ± 0.5174	0.0030 ± 0.0707	0.3060 ± 0.5681	0.0100 ± 0.1411
Mallows (0.4)	0.0520 ± 0.2221	0.0020 ± 0.0447	**0.0000**± 0.0000	**0.0000**± 0.0000	0.3750 ± 0.6006	0.0080 ± 0.0891	0.4090 ± 0.6149	0.0190 ± 0.1633
Mallows (0.5)	0.1680 ± 0.3846	0.0230 ± 0.1500	**0.0060**± 0.1094	0.0120 ± 0.1337	0.5920 ± 0.7225	0.0560 ± 0.2427	0.6320 ± 0.7191	0.0850 ± 0.3404
Mallows (0.6)	0.3180 ± 0.5149	0.0740 ± 0.3076	**0.0390**± 0.2821	0.0400 ± 0.2578	0.8750 ± 0.8719	0.1350 ± 0.3936	0.8460 ± 0.8860	0.1690 ± 0.4320
Mallows (0.7)	0.4740 ± 0.6370	0.1460 ± 0.4460	0.1440 ± 0.5362	**0.1120**± 0.4601	1.1680 ± 0.9842	0.2520 ± 0.5429	1.2130 ± 1.0624	0.2870 ± 0.5992
Mallows (0.8)	0.7550 ± 0.7985	0.3230 ± 0.6980	0.4010 ± 0.8700	**0.2910**± 0.7258	1.8050 ± 1.3102	0.5410 ± 0.8155	1.7360 ± 1.2923	0.5300 ± 0.8694
Mallows (0.9)	0.9780 ± 0.8924	0.4980 ± 0.8993	0.6590 ± 1.0844	**0.3980**± 0.8522	2.3900 ± 1.6101	0.7930 ± 1.0915	2.3780 ± 1.5854	0.8110 ± 1.0528
Conitzer	0.5780 ± 0.6359	0.2080 ± 0.4300	**0.0000**± 0.0000	0.0950 ± 0.3378	4.0440 ± 1.9591	0.7770 ± 0.8591	1.8960 ± 1.1083	0.3020 ± 0.5010
Walsh	0.3150 ± 0.5099	0.1320 ± 0.3503	**0.0000**± 0.0000	0.0480 ± 0.2318	1.1800 ± 0.9383	0.3100 ± 0.5497	1.5150 ± 0.6166	0.2060 ± 0.4333
Urn (0.01)	1.0110 ± 0.9507	**0.5040**± 0.8847	0.7960 ± 1.1932	0.5140 ± 0.9575	2.5970 ± 1.6791	0.8610 ± 1.1140	2.8640 ± 1.7334	0.8720 ± 1.1049
Urn (0.02)	1.0650 ± 0.9475	0.5760 ± 0.9535	0.7920 ± 1.2034	**0.5280**± 0.9911	2.7380 ± 1.7065	0.9120 ± 1.1329	2.7840 ± 1.6704	0.8910 ± 1.1001
Urn (0.05)	1.0060 ± 0.8960	**0.4490**± 0.8137	0.7110 ± 1.0717	0.4960 ± 0.9460	2.6690 ± 1.5893	0.8650 ± 1.0667	2.6920 ± 1.5836	0.8230 ± 1.0568
Urn (0.10)	1.0410 ± 0.8753	0.4710 ± 0.8694	0.6730 ± 1.1078	**0.4050**± 0.9054	2.7380 ± 1.6764	0.7740 ± 1.0460	2.8230 ± 1.6231	0.8450 ± 1.0964

Table 6: Mean ± standard deviation of distance to Kemeny output for different voting rules and distributions (n = 25)

References

1. Betz, G., Richardson, K.: Judgment aggregation, discursive dilemma and reflective equilibrium: Neural language models as self-improving doxastic agents. Frontiers in Artificial Intelligence **5**, 900943 (2022). https://doi.org/10.3389/frai.2022.900943, https://www.frontiersin.org/articles/10.3389/frai.2022.900943/full
2. Betz, G., Richardson, K.: Probabilistic coherence, logical consistency, and bayesian learning: Neural language models as epistemic agents. PLOS ONE **18**(2), e0281372 (2023). https://doi.org/10.1371/journal.pone.0281372, https://doi.org/10.1371/journal.pone.0281372
3. Bianchi, F., Chia, P.J., Yuksekgonul, M., Tagliabue, J., Jurafsky, D., Zou, J.: How well can llms negotiate? negotiationarena platform and analysis. arXiv (2024)
4. Boehmer, N., Bredereck, R., Faliszewski, P., Niedermeier, R., Szufa, S.: Putting a compass on the map of elections. arXiv preprint arXiv:2105.07815 (2021)
5. Davidson, D.: Radical interpretation. Dialectica **27**(3-4), 313–328 (1973). https://doi.org/10.1111/j.1746-8361.1973.tb00623.x, https://onlinelibrary.wiley.com/doi/10.1111/j.1746-8361.1973.tb00623.x
6. Dennett, D.C.: Intentional systems. The Journal of Philosophy **68**(4), 87–106 (1971). https://doi.org/10.2307/2025382, https://www.jstor.org/stable/2025382
7. Freedman, G., Toni, F.: Exploring the potential for large language models to demonstrate rational probabilistic beliefs. arXiv preprint arXiv:2504.13644 (2025)
8. Goldstein, S.: LLMs can never be ideally rational. PhilArchive (2024), https://philarchive.org/rec/GOLLCN
9. Gustafsson, J.E.: Money-Pump Arguments. Cambridge Elements: Elements in Decision Theory and Philosophy, Cambridge University Press (2022). https://doi.org/10.1017/9781108754750, https://doi.org/10.1017/9781108754750
10. Hua, W., Liu, O., Li, L., Amayuelas, A., Chen, J., Jiang, L., Jin, M., Fan, L., Sun, F., Wang, W., Wang, X., Zhang, Y.: Game-theoretic llm: Agent workflow for negotiation games. arXiv (2024)
11. Jackson, F., Pettit, P.: In defence of folk psychology. Philosophical Studies **59**(1), 31–54 (1990). https://doi.org/10.1007/BF00368390, https://link.springer.com/article/10.1007/BF00368390
12. Lewis, D.: Radical interpretation. Synthese **27**(3-4), 331–344 (1974). https://doi.org/10.1007/BF00484599, https://link.springer.com/article/10.1007/BF00484599
13. Lewis, D.: Reduction of mind. In: Guttenplan, S. (ed.) A Companion to the Philosophy of Mind, pp. 412–431. Blackwell, Oxford (1994), https://philarchive.org/rec/LEWROM
14. Pettit, P.: Deliberative democracy and the discursive dilemma. Noûs **35**(Supplement 1), 268–299 (2001). https://doi.org/10.1111/0029-4624.35.s1.11
15. Sekar, S., Sikdar, S., Xia, L.: Condorcet consistent bundling with social choice. In: Proceedings of the 16th Conference on Autonomous Agents and MultiAgent Systems. pp. 33–41 (2017)
16. Szufa, S., Faliszewski, P., Skowron, P., Slinko, A., Talmon, N.: Drawing a map of elections in the space of statistical cultures. In: Proceedings of the 19th International Conference on Autonomous Agents and Multiagent Systems. pp. 1341–1349 (2020)
17. Vineberg, S.: Dutch book arguments. In: Zalta, E.N. (ed.) The Stanford Encyclopedia of Philosophy. Metaphysics Research Lab, Stanford University (2012), https://plato.stanford.edu/entries/dutch-book/
18. Wang, C., Szarvas, G., Balazs, G., Danchenko, P., Ernst, P.: Calibrating verbalized probabilities for large language models. arXiv preprint arXiv:2410.06707 (2024), https://arxiv.org/abs/2410.06707
19. Wang, V., Zhang, M.J., Choi, E.: Improving LLM-as-a-judge inference with the judgment distribution. arXiv preprint arXiv:2503.03064 (2025), https://arxiv.org/abs/2503.03064
20. Xu, Y., Ruis, L., Rocktäschel, T., Kirk, R.: Investigating non-transitivity in llm-as-a-judge. arXiv preprint arXiv:2502.14074 (2025)
21. Yu, Y., Liu, Y., He, M., Tao, S., Meng, W., Yang, X., Zhang, L., Ma, H., Su, C., Yang, H., Li, F.: Elspr: Evaluator llm training data self-purification on non-transitive preferences via tournament graph reconstruction. arXiv preprint arXiv:2505.17691 (2025), https://arxiv.org/abs/2505.17691
22. Zhao, X., Wang, K., Peng, W.: Measuring the inconsistency of large language models in preferential ranking. In: Proceedings of the 1st Workshop on Towards Knowledgeable Language Models (KnowLLM 2024). pp. 171–176. Association for Computational Linguistics, Bangkok, Thailand (August 2024). https://doi.org/10.18653/v1/2024.knowllm-1.14, https://aclanthology.org/2024.knowllm-1.14/
23. Zhu, J.Q., Griffiths, T.L.: Incoherent probability judgments in large language models. In: Proceedings of the 46th Annual Conference of the Cognitive Science Society (2024), https://arxiv.org/abs/2401.16646
24. Zhu, J.Q., Yan, H., Griffiths, T.L.: Recovering event probabilities from large language model embeddings via axiomatic constraints. arXiv preprint arXiv:2505.07883 (2025)

Acknowledgments. The authors thank Milos Bisenic for running experiments on probabilistic coherence.

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