Maximizing Neutrality in News Ordering

Rishi Advani
University of Illinois Chicago
Chicago, IL, USA
radvani2@uic.edu

Paolo Papotti
EURECOM
Biot, France
papotti@eurecom.fr

Abolfazl Asudeh
University of Illinois Chicago
Chicago, IL, USA
asudeh@uic.edu

ABSTRACT

The detection of fake news has received increasing attention over the past few years, but there are more subtle ways of deceiving one's audience. In addition to the content of news stories, their presentation can also be made misleading or biased. In this work, we study the impact of the ordering of news stories on audience perception. We introduce the problems of detecting cherry-picked news orderings and maximizing neutrality in news orderings. We prove hardness results and present several algorithms for approximately solving these problems. Furthermore, we provide extensive experimental results and present evidence of potential cherry-picking in the real world.

CCS CONCEPTS

• **Theory of computation** → *Problems, reductions and completeness*; **Approximation algorithms analysis**; **Theory and algorithms for application domains**; • **Mathematics of computing** → **Graph algorithms**; **Matchings and factors**; **Paths and connectivity problems**; • **Information systems** → *Social networks*; **Content ranking**.

KEYWORDS

media bias; news ordering; neutrality; cherry-picking

ACM Reference Format:
Rishi Advani, Paolo Papotti, and Abolfazl Asudeh. 2023. Maximizing Neutrality in News Ordering. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '23), August 6–10, 2023, Long Beach, CA, USA.* ACM, New York, NY, USA, 14 pages. https://doi.org/10.1145/3580305.3599425

1 INTRODUCTION

Access to information is a hallmark of modern democracy and society. Many people rely on online news sources or social media to understand the problems facing their communities, stay informed on current events, and determine who they would like to represent them in government. As such, they often have to blindly trust that their news sources are providing them with accurate information and presenting it in an unbiased way. Media organizations can take advantage of this trust to push their own agendas and spread disinformation when it benefits them financially or politically. As people become more aware of the prevalence of misinformation online, news sources risk losing their credibility if they are caught spreading outright lies. But even if the information they provide is technically accurate, there are still ways in which they can inject bias into its presentation [51].

One way in which this can be done is through deceptive ordering of news stories in a broadcast or web page. For example, suppose two headlines are placed next to each other in a user's feed:

- "Immigration rates are on the rise again"
- "Crime rates in major cities have reached historic highs"

Viewing one headline may influence the user's opinion of the story corresponding to the second headline — by affecting their belief in the veracity of the story, their stance (for or against) on the events in the story, or by inducing them to perceive a causal relationship when there is only correlation. We term this phenomenon "opinion priming". Viewing one headline primes[1] the user to form a certain opinion when shown the second headline.

In this particular case, the user may perceive a causal relationship between these two events, even though it is never explicitly stated by the news source. In reality, this correlation could be completely spurious, but a news organization with ulterior motives could use this psychological trick by placing the two stories next to each other to influence the views of its audience.

On the other hand, a socially responsible news corporation, or an organization auditing a less scrupulous corporation to hold them accountable, may seek to order news stories in a way that minimizes this risk of opinion priming. Alternatively stated, they may seek to *maximize the neutrality of a news ordering.*

Acknowledging that there are other objectives at play as well, including profitability for the news corporation and relevance to the user, in this paper, we focus on maximizing neutrality and leave simultaneous optimization of all these objectives as an important direction for future work.

Problem Novelty. In this paper, we study news ordering neutrality[2] from an algorithmic perspective. While there has been extensive work in recent years on different aspects of news coverage selection bias [18, 57, 63, 65], diversifying news recommendations [13, 46, 61, 69], and computational fact-checking [49, 72, 99], to the best of our knowledge, this paper is *the first to consider the impact of the ordering of news stories on neutrality.*

Contributions. Our contributions in this paper are as follows:

- We formalize the notion of news ordering and introduce the problems of (a) detecting cherry-picked news orderings and (b) maximizing neutrality in news orderings (§2).

KDD '23, August 6–10, 2023, Long Beach, CA, USA

ACM ISBN 979-8-4007-0103-0/23/08…$15.00
https://doi.org/10.1145/3580305.3599425

[1] The occurrence of priming has been extensively studied in related settings [2, 14, 25, 31, 82]. We performed a user study (§6.4) to confirm the existence of priming in our setting.

[2] This paper addresses one important technical piece of a larger socio-technical problem with many dimensions [51]. We discuss related work in more detail in §7.

- We present an algorithm to efficiently detect cherry-picked news orderings (§3). The algorithm uses random shuffling and tail inequalities to detect if the neutrality of the given ordering is significantly different from the mean.
- We study the problem of maximizing neutrality in news orderings. We prove results on the theoretical hardness of solving this problem and provide several approximation algorithms (§4 and §5). Our algorithms make (non-trivial) connections to other problems such as max-weight matching [37] and max-weight cycle cover [17], by using them as subroutines in the algorithms.
- We introduce new variations of the fundamental maximum traveling salesman problem and propose algorithms that can be used to solve problems with a broad range of applications. In particular, we define the PathMaxTSP problem of finding a Hamiltonian path with maximum total weight in a graph.
- We conduct comprehensive experiments on real and synthetic datasets to validate our theoretical results (§6). We were able to find *potential evidence of cherry-picked orderings in the real world*, further motivating our study. In addition, our user study with over 50 participants confirms the existence of priming in our setting.

We conclude the paper with a discussion of related work (§7) and directions for future research (§8).

2 PROBLEM SETUP

Let $\mathbf{t} = \{ t_1, t_2, \ldots, t_n \}$ be a set of n news stories to be presented by a news source. Let $\mathbf{s} = \{ s_1, s_2, \ldots, s_n \}$ be a permutation of the integers from 1 to n representing an ordering of those news stories: news story t_i is presented in position s_i.

When news headlines are placed near each other, the user's opinion of one may be influenced by the other. Our objective is twofold: we aim to detect when a news source has cherry-picked the ordering of its news stories, and to find the ordering that minimizes this risk of opinion priming.

To model this, we define a *pairwise opinion priming (POP) function* $C\colon \mathbf{t} \times \mathbf{t} \to \mathbb{R}$ that takes as input a pair[3] of stories (t_i, t_j), where $t_i \neq t_j$, and returns a real number in the range $[0, 1]$. An output of 1 indicates certainty that opinion priming will occur between two stories if they are in adjacent slots. An output of 0 indicates that no opinion priming will occur. Note that the likelihood of opinion priming occurring for a particular individual is impacted by their own beliefs and mentality and may differ from that of another individual. Thus, we consider the incidence of opinion priming over a group of individuals. More precisely, the function C reflects the *average* pairwise opinion priming over the audience.

The values of C can be determined in several ways. For example, a real audience's perception can be surveyed, an auditing agency can crowdsource answers to questions on opinion priming between pairs of news stories, or a domain expert can assign values based on their own judgment. We use crowdsourcing in our user studies to estimate the values of C, confirming this method's feasibility in practice. In this paper, we assume the values of C are given as input. Thus, the problems and solutions proposed in this paper are agnostic to the choice of technique for determining C.

We also consider the distance between two stories in an ordering. As the distance increases, any opinion priming between the pair of stories will diminish accordingly; the audience will not form as strong an association if the stories are presented far apart from each other. We define a *decay function* $D\colon \mathbb{N} \to \mathbb{R}$ that takes as input the distance between two distinct time slots and returns a real number in the range $[0, 1]$ with $D(1) = 1$ and D monotonic.

Using the POP and decay functions, we can now define the pairwise neutrality of a pair of news stories.

Definition 2.1 (Pairwise Neutrality). Given a set of news stories $\mathbf{t}$, an ordering $\mathbf{s}$, a POP function C, and a decay function D, the *pairwise neutrality* between distinct news stories t_i and t_j is defined as $N_{i,j} = 1 - D(|s_j - s_i|) \cdot C(t_i, t_j)$.

We now give an example to illustrate the concepts discussed so far. Suppose we have the following decay function.

$$D(d) = \begin{cases} 1 & \text{if } d = 1 \\ 0 & \text{otherwise} \end{cases} \tag{1}$$

This function treats pairs of headlines as having no risk of opinion priming if they are more than one position away from each other.

Example 2.2. Consider a set of news stories $\mathbf{t} = \{ t_1, t_2, t_3, t_4 \}$ with the following POP function C.

C	t_1	t_2	t_3
t_2	0.1		
t_3	0.3	0.7	
t_4	0.2	0.8	1

If we order the stories in $\mathbf{t}$ as t_1, t_3, t_4, t_2, then we have the following values for $\mathbf{s}$.

s_1	s_2	s_3	s_4
1	4	2	3

For example, $s_3 = 2$ because t_3 is placed second in the ordering.

Using Equation 1 for the decay function, the pairwise neutrality between t_1 and t_2, for example, is

$$N_{1,2} = 1 - D(|s_2 - s_1|) \cdot C(t_1, t_2) = 1 - 0 \times 0.1 = 1\,.$$

The pairwise neutrality for all pairs of news stories is given below.

N	t_1	t_2	t_3
t_2	1		
t_3	0.7	1	
t_4	1	0.2	0

Using the notion of pairwise neutrality, we can now define neutrality for a whole news ordering. At a high-level, a news ordering is neutral if the pairwise neutrality between all pairs of news stories is "high". More formally, we use Definition 2.3 to quantify neutrality in a news ordering. For our purposes, an *aggregation function* is any function that takes a set as input and returns a single real number in $[0, 1]$ as output.

Definition 2.3 (News Ordering Neutrality). Given a set of news stories $\mathbf{t}$, a POP function C, a decay function D, and an aggregation function agg, the neutrality of a news ordering $\mathbf{s}$ is defined as

$$\mathrm{Neut}_{\mathrm{agg}}(\mathbf{s}) = \operatorname*{agg}_{1 \le i < j \le n} N_{i,j}\,,$$

[3]We could instead use *ordered* pairs if we wished to model the POP function as being affected by the order of the two stories, and nearly all the results in this paper would still hold. More details can be found in the appendix.

Table 1: Table of Notations

Notation	Description
n	The cardinality of the set $\mathbf{t}$
t_i	A news story in the set $\mathbf{t}$
s_i	The slot assigned to t_i in the ordering $\mathbf{s}$
C	The pairwise opinion priming function
D	The decay function
$N_{i,j}$	The pairwise neutrality between t_i and t_j
$\mathrm{Neut}_{\mathrm{agg}}(\mathbf{s})$	The neutrality of the ordering $\mathbf{s}$ under the aggregation function "agg"

where $N_{i,j}$ is the pairwise neutrality between t_i and t_j.

Analogously, for any aggregation function agg, we will denote the optimization problem of finding the ordering $\mathbf{s}$ that maximizes $\mathrm{Neut}_{\mathrm{agg}}$ by $\textsc{Neutrality}_{\mathrm{agg}}$.

We now define two aggregation functions that we will use throughout the paper.

Definition 2.4 (Conditional Average Aggregation). Given the pairwise neutrality values $N_{i,j}$ for a set of news stories, an ordering $\mathbf{s}$, and a decay function D, the conditional average is defined as the average of the pairwise neutrality values over the support of D. I.e., if D^+ is the set of pairs (i, j) where $D(|s_j - s_i|) > 0$, then the conditional average is the average of the neutrality values $N_{i,j}$ over all $(i, j) \in D^+$. If $D > 0$ for all inputs, then this is just a simple average. For brevity, we will refer to this function by "avg".

Definition 2.5 (Minimum Aggregation). The minimum aggregation function simply returns the minimum element in a set. We will refer to this function by "min".

Example 2.6. Consider the same set of news stories $\mathbf{t}$, ordering $\mathbf{s}$, POP function C, and decay function D from Example 2.2. Using avg as the aggregation function, we have

$$\mathrm{Neut}_{\mathrm{avg}}(\mathbf{s}) = (0.7 + 0.2 + 0)/3 = 0.3\,.$$

Similarly, the neutrality of $\mathbf{s}$ under min aggregation is

$$\mathrm{Neut}_{\mathrm{min}}(\mathbf{s}) = \min N_{i,j} = 0\,.$$

Having defined the notion of neutrality in news ordering, we will begin by studying how to detect cherry-picked news orderings in §3. Our main objective in this paper is to find news orderings that maximize neutrality, which we shall do in §4 and §5. While the techniques proposed in §3 are agnostic to the choice of decay function, in §4 and §5, we will restrict ourselves to the decay function given in Equation 1. This allows us to model the problem using the language of graph theory. Analyzing more complex decay functions is an important direction for future work.

We define a graph representation of the problem as follows. For each news story t_i, we include a vertex v_i. For each pair of distinct stories t_i and t_j, we include an edge between v_i and v_j with weight $N_{i,j}$. For brevity, henceforth in this paper, assume all graphs are simple, complete, undirected, weighted, and have nonnegative edge weights unless otherwise specified. The requirement that the graphs are simple and complete is equivalent to stating that every pair of distinct vertices is joined by exactly one edge.

We define some graph theory terms that are used in the paper.

Definition 2.7 (Hamiltonian cycle). In a graph $G = (V, E)$, a *Hamiltonian cycle* is a simple cycle that includes all vertices in V.

Definition 2.8 (Hamiltonian path). In a graph $G = (V, E)$, a *Hamiltonian path* is a simple path that includes all vertices in V.

Definition 2.9 ($\textsc{HamPath}$). Given a graph G, $\textsc{HamPath}$ is the problem of determining if there exists a Hamiltonian path in G.

With the restriction of the decay function to Equation 1, the problem of finding an ordering of news stories that maximizes Neut is equivalent to finding a Hamiltonian path that maximizes Neut. But first, in §3 we propose our algorithm for detecting cherry-picking in an ordering (with any decay function).

3 DETECTING CHERRY-PICKED ORDERINGS

We begin by illustrating how to detect cherry-picked news orderings. Suppose we have a set of news stories $\mathbf{t}$, a POP function C, a decay function D, and an aggregation function agg. Then, given a news ordering $\mathbf{s}$, we can deduce that it was likely cherry-picked if $\mathrm{Neut}_{\mathrm{agg}}(\mathbf{s})$ differs significantly from the average neutrality over all possible orderings of $\mathbf{t}$. If $\mathrm{Neut}_{\mathrm{agg}}(\mathbf{s})$ is significantly lower than the average, then we have successfully detected bias in the ordering. On the other hand, if $\mathrm{Neut}_{\mathrm{agg}}(\mathbf{s})$ is significantly higher than the average, then we can determine that the specified ordering was deliberately chosen in the interest of fairness.

If we knew the population mean and standard deviation, we could use Chebyshev's inequality to obtain an upper bound on the deviation from the mean. However, the number of possible orderings is combinatorially large ($n!$), so we cannot compute the neutrality for all of them. If we instead generate a sample of r random orderings using Fisher-Yates shuffles [36, 42], we can use the Saw-Yang-Mo inequality [80], which only requires the *sample* mean and standard deviation, to obtain an upper bound on deviation from the sample mean. For convenience, we use a simplified (and slightly looser) form of Kabán's variant of the inequality [58]:

$$\Pr\left(\left|X - \overline{Y}\right| \geq \lambda\sigma\sqrt{\frac{r+1}{r}}\right) \leq \frac{1}{\lambda^2} + \frac{1}{r}\,, \tag{2}$$

where $\overline{Y}$ is the sample mean, σ is the unbiased sample standard deviation,[4] and the value for λ is set such that the difference between the neutrality of the given ordering and $\overline{Y}$ is $\lambda\sigma\sqrt{(r+1)/r}$.

Example 3.1. If we use, say, $r = 50$ samples with $\lambda = 5$, then using Equation 2, we have the following.

$$\Pr\left(\left|X - \overline{Y}\right| \geq 5\sigma\sqrt{\frac{51}{50}}\right) \leq \frac{1}{5^2} + \frac{1}{50} = \frac{3}{50}$$

The probability that the neutrality of a truly random ordering is greater than $5\sqrt{51/50} \approx 5.05$ sample standard deviations from the sample mean is less than 6%. Thus, if the neutrality of our given ordering is that far from the sample mean, it is highly likely that it was cherry-picked.

Users can select the value for parameter r based on their problem size, aggregation function's complexity, access to computational

[4] Usually σ is used for population deviation and s for sample deviation, but we chose to avoid the use of s to avoid confusion with our notation $\mathbf{s}$ for news orderings.

resources, and error tolerance. We suggest $r = 300$ as a reasonable starting point.

Now, we analyze the time complexity of the detection procedure. We can compute r random permutations in $O(rn)$ time using Fisher-Yates shuffles. We can compute the neutrality of the r orderings in $O(rn^2)$ time. Computing the sample mean and standard deviation of the r values takes $O(r)$ time and evaluating the test statistic takes $O(1)$ time. Thus, overall, the algorithm takes $O(rn^2)$ time.

Furthermore, if we make certain assumptions, we can obtain a running time linear in n. If we use the decay function given by Equation 1 along with an aggregation function that can be computed in linear time (e.g., avg or min), we can compute the neutrality of the r orderings in $O(rn)$ time for an overall running time of $O(rn)$.

4 MAXIMIZING NEUTRALITY UNDER AVERAGE AGGREGATION

In the previous section, we considered the problem of detecting cherry-picked news orderings. Now, we move on to the main focus of our paper: finding news orderings with maximum neutrality.

Before beginning the technical content, we stop to emphasize the importance of computational approaches to this problem. Due to the combinatorially large number of possible orderings, the task is infeasible for a human with even a very small number of news headlines. For example, *with only 10 headlines, there are 3,628,800 potential orderings to consider.* Thus, even in contexts where few stories are presented (e.g., a television broadcast), computational approaches are important. Furthermore, there are contexts in which the number of stories grows much larger (e.g., scrolling through a social media feed), where computational approaches are critical.

First, we consider the scenario where our aggregation function is the avg function. This is a natural aggregation function to use; if we are equally invested in the pairwise neutrality of each pair of stories, it makes sense to maximize the average (mean) value. Note that this is exactly equivalent to maximizing the sum of the pairwise neutrality of each pair but with the added benefit that the neutrality will always be a value in the range $[0, 1]$, so it is easier to make intuitive judgments about whether it is "high" or "low".

In the graph theory representation, the problem is now equivalent to finding a Hamiltonian path with maximum weight. To the best of our knowledge, *we are the first to study this problem.* We will call this problem the "path maximum traveling salesman problem", or PathMaxTSP.

Definition 4.1 (PathMaxTSP). Given a graph G, *PathMaxTSP* is the problem of finding a Hamiltonian path with maximum total weight.

Theorem 4.2. *PathMaxTSP is NP-hard.*[5]

Corollary 4.3. *Neutrality$_{\text{avg}}$ is NP-hard.*

Given the above hardness results, in the rest of the section, we design approximation algorithms to solve PathMaxTSP.

Between our algorithms ApproxMat and ApproxCC, ApproxCC has improved efficiency with the same approximation factor, so we advocate for its use over ApproxMat in all cases. We include ApproxMat in our exposition for its comparative simplicity and in the hope that it inspires future work in this area. Our algorithm Approx3CC achieves the best approximation factor but has an unreasonably slow running time.

[5]Proofs of all theorems stated in this section can be found in the appendix.

4.1 Approximation via Iterated Matching

The first algorithm, ApproxMat (pseudocode in the appendix), works by making a connection to the well-known max-weight matching problem, where in a weighted graph, the goal is to find a set of disjoint edges with maximum total weight [62].

In each iteration k, the algorithm constructs a graph G_{k+1} used in the next iteration. To do so, it first finds a max-weight matching in the graph G_k. Then, for every pair in the matching, it adds a "super node" to G_{k+1}. The super node represents a path in the original graph G. To perform this merge, the algorithm joins the represented paths by the pair of endpoints with maximum edge weight. If $|V_k|$ is odd, one of nodes in G_k remains unmatched and gets added to G_{k+1} as is. The weight of the edge between each pair of nodes in G_{k+1} is the maximum edge weight between the ends of their represented paths. The algorithm continues this process until there is only one super node left. The path represented by the final super node is a Hamiltonian path in the original graph.

Example 4.4. Consider a set of stories $\mathbf{t} = \{ t_1, \ldots, t_6 \}$ with pairwise neutrality values as shown in the graph of Figure 1a (for visual clarity, we omit four edges with weight zero). ApproxMat starts by finding the max-weight matching $\{(t_1, t_3), (t_2, t_4), (t_5, t_6)\}$, as highlighted in the figure. Next, the algorithm replaces the pairs in the matching with super nodes: $\langle t_1, t_3 \rangle$, $\langle t_2, t_4 \rangle$, and $\langle t_5, t_6 \rangle$. In the second iteration, the algorithm selects edge (t_4, t_5) with weight 1 to join the super nodes $\langle t_2, t_4 \rangle$ and $\langle t_5, t_6 \rangle$ (Figure 1b). In the final iteration, the algorithm matches $\langle t_2, t_4, t_5, t_6 \rangle$ to $\langle t_1, t_3 \rangle$, via edge (t_1, t_2), creating the final super node, $\langle t_3, t_1, t_2, t_4, t_5, t_6 \rangle$. The neutrality of the resulting ordering under avg aggregation is $4.1/5 = 0.82$.

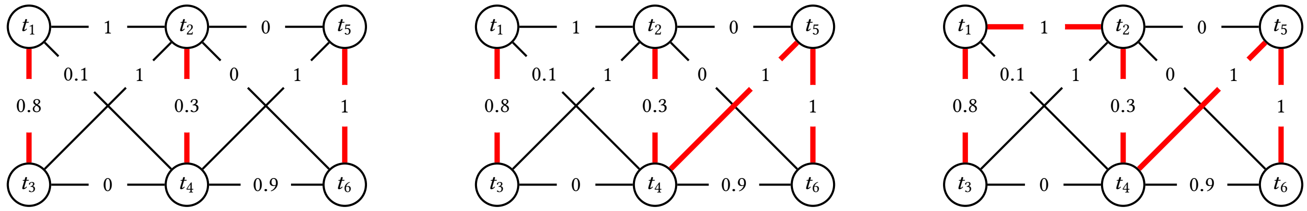
Theorem 4.5. *ApproxMat returns a 1/2-approximation for PathMaxTSP.*

We have not yet explicitly specified a subroutine to compute a max-weight matching. Classically, this can be done in $O(n^4)$ time using Edmonds' blossom algorithm [37]. Alternatively, it can be done in $O(n^2 \varepsilon^{-1} \log \varepsilon^{-1})$ time for any fixed error ε [33]. Properly implemented, the runtime of each iteration of the loop is dominated by the cost of the matching. If we use Edmonds' blossom algorithm, then each iteration takes $O(|V_k|^4)$ time. By the master theorem [23], the overall runtime is then $O(n^4)$. While polynomial, ApproxMat has a high time complexity. Therefore, next we propose our algorithm ApproxCC that, while maintaining the same approximation ratio, reduces time complexity by a factor of n.

4.2 Approximation via Iterated Cycle Cover

The second algorithm, ApproxCC (pseudocode in Algorithm 1), finds a max-weight cycle cover, defined as a set of cycles[6] of maximum total weight such that every vertex is included in exactly one cycle. Then, it removes the min-weight edge from each cycle. The resulting paths are treated as super nodes (as in ApproxMat) and the process is repeated until there is only one super node remaining. The final super node implicitly gives a Hamiltonian path in the original graph.

[6]Here, cycles of length 2 are allowed.



(a) First iteration **(b)** Second iteration **(c)** Third (final) iteration

Figure 1: (Example 4.4) ApproxMat returns the ordering $\langle t_3, t_1, t_2, t_4, t_5, t_6 \rangle$

Algorithm 1 Approximating PathMaxTSP via iterated cycle cover

procedure ApproxCC($G = (V, E)$)
  $k \leftarrow 0$; $G_k \leftarrow G$; $size \leftarrow n$
  while $size > 1$ **do**
    Compute a max-weight cycle cover C in G_k.
    Construct a graph $G_{k+1} = (V_{k+1}, E_{k+1})$ as follows.
    $V_{k+1} \leftarrow \{ \}$
    for all $c \in C$ **do**
      Remove the min-weight edge, and let $(p_1), \ldots, (p_a)$ be the resulting path.
      $p \leftarrow p_1$
      for all $i \in [2, a]$ **do**
        Let $u_1, \ldots, u_b$ be the path denoted by p and $v_1, \ldots, v_d$ be the path denoted by p_i.
        if $w(u_b, v_1) > w(u_b, v_d)$ **then** $p \leftarrow p, v_1, \ldots, v_d$
        else $p \leftarrow p, v_d, \ldots, v_1$
      Add (p) to V_{k+1}.
    for all pairs $(u_1, \ldots, u_a), (v_1, \ldots, v_b)$ in V_{k+1} **do**
      Add the edge $((u_1, \ldots, u_a), (v_1, \ldots, v_b))$ to E_{k+1} with weight $w(u_a, v_1)$.
    $size \leftarrow |V_{k+1}|$; $k \leftarrow k + 1$
  return the Hamiltonian path $v_1, \ldots, v_n$ in G where $(v_1, \ldots, v_n)$ is the sole vertex in V_k.

In this algorithm, we reduce the problem of computing a cycle cover to that of computing a bipartite matching. The original reduction is due to Tutte [90]; an accessible presentation of the specific case we are interested in is given by Nikolaev and Kozlova [73].

Example 4.6. Consider a set of stories $\mathbf{t} = \{ t_1, \ldots, t_6 \}$ with pairwise neutrality values as shown in the graph of Figure 2a (for visual clarity, we omit four edges with weight zero). ApproxCC starts by finding the max-weight cycle cover $\{(t_1, t_2, t_3), (t_4, t_5, t_6)\}$, as highlighted in the figure. Then, it removes the min-weight edge from each cycle (Figure 2b). Next, the algorithm replaces these paths with super nodes: $\langle t_1, t_2, t_3 \rangle$ and $\langle t_4, t_5, t_6 \rangle$. In the second iteration, the algorithm joins the two super nodes to form the cycle $(t_3, t_2, t_1, t_4, t_5, t_6)$ and removes the edge (t_6, t_3) to create the final super node (Figure 2c). The neutrality of the resulting ordering under avg aggregation is $4.1/5 = 0.82$.

Theorem 4.7. *ApproxCC returns a* $1/2$*-approximation for PathMaxTSP.*

We can compute a max-weight bipartite matching in $O(n^3)$ time using the Hungarian method [38, 88]. It can also be done in expected time $O(n^2 \log n)$ if the edge weights are i.i.d. random variables [60]. Again, we can use the linear-time approximation algorithm for general max-weight matching instead. Properly implemented, the runtime of each iteration of the loop is dominated by the cost of the bipartite matching. If we use the Hungarian method, then each iteration takes $O(|V_k|^3)$ time. By the master theorem, the overall runtime is then $O(n^3)$.

4.3 Approximation via 3-Cycle Cover

So far, both algorithms proposed are $1/2$-approximation algorithms. Our third algorithm, Approx3CC (pseudocode in the appendix) improves the approximation factor to $2/3$, but at a high computation cost. It finds a max-weight 3-cycle cover, defined as a set of cycles of maximum total weight such that every vertex is included in exactly one cycle and every cycle has length at least 3. It then removes the min-weight edge from each cycle and arbitrarily joins the resulting paths to form a Hamiltonian path.

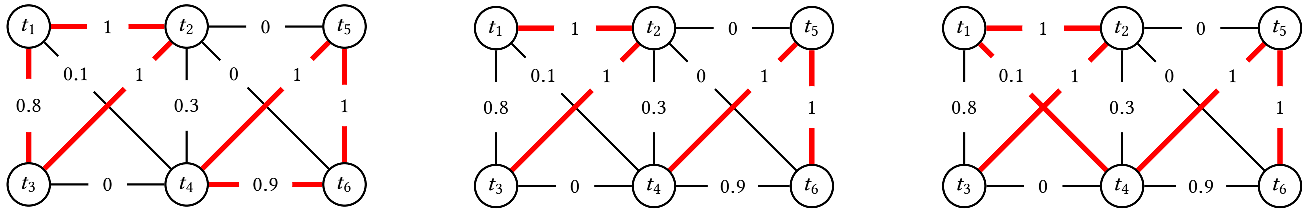
We reduce the problem of computing a max-weight 3-cycle cover to that of computing a max-weight matching on a more complex graph. A thorough presentation of the reduction is given by Eppstein [39]. The original reduction was a generalization of this argument given by Tutte [90].

We have not yet explicitly specified a subroutine to compute a max-weight matching. We can use Edmonds' blossom algorithm or the linear time approximation algorithm for max-weight matching. The time to compute the matching dominates the rest of the computation, so the overall time complexity of Approx3CC is the time complexity of running the preferred algorithm on a graph with $|V| = 2n^2 - 4n$ and $|E| = n^3 - 3.5n^2 + 2.5n$. With the blossom algorithm, this leads to a overall runtime of $O(n^7)$. As such, this algorithm is not practical (we do not use it in our experiments), but it is of theoretical interest, as evidenced by the following theorem.

Theorem 4.8. *Approx3CC returns a* $2/3$*-approximation for PathMaxTSP.*

5 MAXIMIZING NEUTRALITY UNDER MIN AGGREGATION

Now, we consider the scenario where our aggregation function f is the min function: it returns the smallest element in a totally ordered set. This is another well motivated aggregation function; if we are okay with many imperfect pairs but just want to make sure that no pair is *too* bad, then it makes sense to maximize the neutrality of the most biased pair. In the graph representation, the problem is now equivalent to finding a Hamiltonian path with maximum min-weight edge. This problem was first studied by Arkin et al. [6].



(a) First iteration **(b)** First iteration (transforming cycles to paths) **(c)** Second iteration (after transforming cycle to path)

Figure 2: (Example 4.6) ApproxCC returns the ordering $\langle t_3, t_2, t_1, t_4, t_5, t_6 \rangle$

We will refer to it as the "path maximum scatter traveling salesman problem", or PathMaxScatterTSP.

Definition 5.1 (PathMaxScatterTSP). Given a graph G, *PathMaxScatterTSP* is the problem of finding a Hamiltonian path with minimum edge weight maximized.

The cycle variant has been successfully addressed with heuristic methods [92], but there are no published results on algorithms for PathMaxScatterTSP. Unfortunately, we will have to rely on heuristic methods, as we have the following inapproximability result by Arkin et al. [6].

Theorem 5.2. *There is no polynomial-time constant-factor approximation algorithm for PathMaxScatterTSP unless* P = NP.

Corollary 5.3. *There is no polynomial-time constant-factor approximation algorithm for Neutrality*$_{\text{min}}$ *unless* P = NP.

We now adapt the BottleneckATSP heuristic algorithm of LaRusic and Punnen [64] and design the first heuristic algorithm for the PathMaxScatterTSP problem.

Given a graph $G = (V, E)$ and parameter δ, we define the graph $G' = (V, E')$ with edge set defined as follows. For each edge $(u, v) \in E$ with weight at least δ, we have an edge $(u, v) \in E'$ with weight 0. For each edge $(u, v) \in E$ with weight w less than δ, we have an edge $(u, v) \in E'$ with weight $\delta - w$. Then, there is a Hamiltonian cycle in G with minimum edge weight at least δ if and only if there is a cycle in G' with total weight 0. We use any heuristic solver for TSP to predict whether such a cycle exists in G'; we will use 2-opt [24], a simple but effective local search algorithm. One useful property of 2-opt is that it is an *anytime* algorithm. The user can choose to terminate it before convergence and obtain a slightly sub-optimal solution to save time if computational resources are scarce.

If we perform a binary search over all possible edge weights δ, we can estimate the maximum value of δ such that there is a Hamiltonian cycle in G with minimum edge weight at least δ. If we then remove the min-weight edge from that cycle, the resulting path is an approximate solution to the instance of PathMaxScatterTSP. The full pseudocode is shown in Algorithm 2.

6 EXPERIMENTS

Now that we have finished introducing all the algorithms, we present our experiments. First, we describe the data collection and generation process, then we discuss hardware and implementation details, and finally, we report the results of the experiments. Our implementations and data are freely available online [1].

Algorithm 2 Heuristic for PathMaxScatterTSP

procedure IsFeasible($G = (V, E), \delta$)
  Define the graph $G' = (V, E')$ such that for each edge $(u, v) \in E$ with weight w, we have an edge $(u, v) \in E'$ with weight $\max(\delta - w, 0)$.
  $C' \leftarrow$ 2-opt(G')
  Let C be the corresponding cycle in G and W the total weight of C'.
  if $W = 0$ **then return** C, `True`
  else return C, `False`
procedure BinarySearch(G, l, r)
  $m \leftarrow \lceil (l + r)/2 \rceil$
  $C, X \leftarrow$ IsFeasible(G, m)
  if $l = r$ **then return** C
  if X **then return** BinarySearch(G, m, r)
  return BinarySearch($G, l, m - 1$)
procedure PathMaxScatterTSP(G)
  Consider the sorted list of all edge weights.
  Let l and r be the minimum and maximum, respectively.
  $C \leftarrow$ BinarySearch(G, l, r)
  Remove the min-weight edge from C to construct a path P.
  return P

6.1 Data

To measure the empirical performance of our algorithms, we tested them on real, semi-synthetic, and synthetic data.

Real Data. We selected two news sources based in the United States: American Thinker and The Federalist. On July 24, 2022, we collected the first 11 headlines from each homepage. Within each source, every possible pair of headlines was labeled by 3 out of 6 total annotators according to whether or not they thought it may lead to significant opinion priming. The following prompt was given to all annotators.

> *Suppose an average adult residing in the United States is viewing news headlines.*
> *If the subject views headline A and headline B together, will their impression of either story likely be different from what it would have been if the subject had viewed them individually?*
> *I.e., would viewing the headline of one story influence their opinion on the veracity of the content of the other story or the causes, effects, or benefits of the events discussed within?*

Every annotator was given 55 headline pairs; for each pair, the possible answers were "yes", "no", and "maybe", corresponding to pairwise neutrality values of 0, 1, and 0.5, respectively. For 35.5%

of the pairs, all annotators agreed. For 85.5%, the majority agreed. The 15.5% of cases where there was no consensus confirm our expectation that opinion priming can be specific to each individual.

After collecting the labels, we took the average value for each pair of headlines and used these values to define the POP function. These values serve as loose estimates of average audience perceptions.

Semi-Synthetic Data. To show that our algorithms scale to larger data, we created a dataset larger than we were able to collect labels for. We considered several common probability distributions, and for each one, we computed both the parameters that best fit our data and the likelihood of the data given that distribution with those parameters. We then chose the distribution with greatest likelihood. In short, we found the distribution that best fit the labeled data. For our data, the best fit was a beta distribution.

To generate the semi-synthetic data, we create a complete graph, and for each edge, we sample a value from the chosen distribution for the weight. In this way, we are able to generate a graph corresponding to a dataset of any size that has edge weights matching the distribution of weights in the real data.

Synthetic Data. In addition to the semi-synthetic data, we also generated graphs with edge weights not drawn independently. Intuitively, if $C(u, w)$ and $C(v, w)$ are high, then $C(u, v)$ is likely to be high as well, so in this setting, we draw edge weights from the original beta distribution, but then enforce that there are no triangles in the graph where only two edges have "high" POP values. Adding a second distribution also served as a way to test the robustness of our algorithms to changes in the data distribution.

6.2 Implementation

Hardware. All experiments were run on a machine with an Intel(R) Core(TM) i5-8265U CPU @ 1.80 GHz and 16.0 GB of memory running Windows 11 Pro.

Software. All methods were implemented using Python 3.10.2. The NetworkX package [50] (version 2.6.3) was used for the graph representations and several graph algorithms. The SciPy library [93] (version 1.8.0) was used in the implementation of the statistical test and for the fitting of and sampling from probability distributions.

Implementation Details. The NetworkX method used for max-weight matching relies on a blossom-type algorithm [37]. For max-weight bipartite matching, we used the algorithm of Karp [60].

When evaluating the algorithms for maximizing neutrality, the methods were run on 4 random graphs with edge weights sampled from the same distribution, and the neutrality values obtained were averaged; this helps account for the randomness in the data. In addition, for each graph, the methods were run 3 times and the minimum execution time was recorded to account for any unrelated changes in processor utilization affecting the speed of computation.

When evaluating the algorithm for detecting cherry-picked orderings, the methods were run 5 times and the minimum execution time was recorded; the relative efficiency of the detection methods allows us to run them a higher number of times.

6.3 Results

Detection. To illustrate the process of detecting cherry-picked orderings, we run our algorithm with varying values of r. Recall that

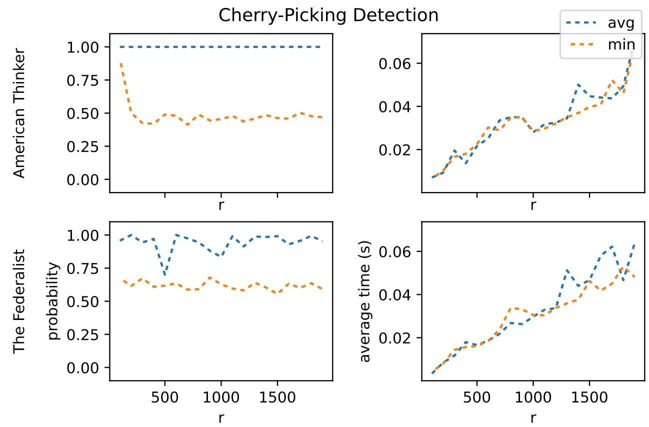


Figure 3: Left: upper bound on the probability of a random ordering having the neutrality of the given ordering; Right: average wall-clock execution time of detection method

r represents the number of random permutations sampled for the statistical test. Thus, higher values of r lead to more accurate testing but slower computation. The values of r used in the experiments are $\{100, 200, \ldots, 2000\}$. The data in Figure 3 confirms that the probabilities converge as r grows.

For a conclusive test, in order to detect evidence of cherry-picking in the real dataset, we ran the detection algorithm with a much larger value of 50000 for r to get the tightest bounds.

Under avg aggregation, we did not discover any significant evidence of cherry-picking by either source. Under min aggregation, on the other hand, *we found evidence of potential cherry-picking for both sources*. For American Thinker, we found that the probability that a random ordering would have neutrality as far from the mean as that of the true ordering is bounded above by 46.49%. Likewise, for The Federalist, we obtained an upper bound of 59.91%. This does not prove that the orderings were cherry-picked, or if they were, give proof of malicious intent, but it does give evidence that cherry-picking may have occurred, especially in the case of American Thinker.

In both cases, the computed neutrality of the ordering was *zero* under min aggregation. We computed the maximum possible neutrality via the brute force method for comparison: 0.67 for American Thinker[7] and 0.83 for The Federalist.

The following is an example of a headline pair with neutrality zero from American Thinker:

- "The Merriam-Webster's online dictionary redefines 'female'"
- "Crayola has joined the woke brigade with a vengeance"

In this case, viewing the second headline may prime the viewer to consider Merriam-Webster's actions to be "woke", a term that is often used as a pejorative. If they had viewed the first headline individually, they would be more likely to form their own (potentially less biased) opinion on the story.

Maximizing Neutrality. We used the algorithms from §4 and §5 to find orderings with high neutrality for the semi-synthetic and synthetic data. For the semi-synthetic data, the results for avg aggregation are shown in Figure 4 and the results for min aggregation in

[7] Our heuristic method also found an optimal solution — in 0.02 seconds, instead of 8 minutes.

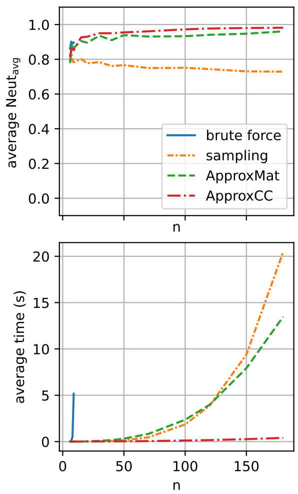


Figure 4: Neutrality and exec. time on semi-synthetic data (avg aggregation)

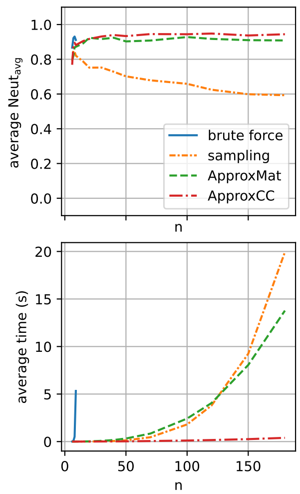


Figure 5: Neutrality and exec. time on synthetic data (avg aggregation)

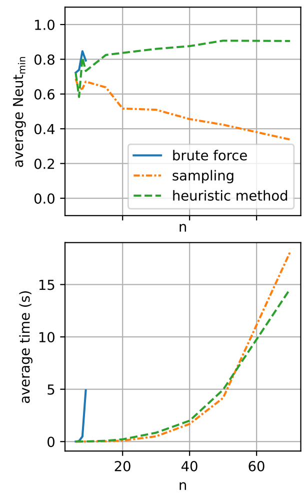


Figure 6: Neutrality and exec. time on semi-synthetic data (min aggregation)

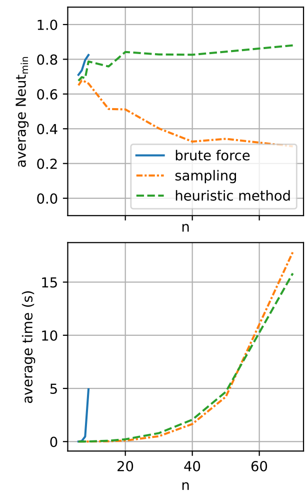


Figure 7: Neutrality and exec. time on synthetic data (min aggregation)

Figure 6. For the synthetic data, the results for avg aggregation are shown in Figure 5 and the results for min aggregation in Figure 7.

The values of n used for testing are based on the sizes expected of real-life datasets. It would be unlikely for a reader to view a contiguous list of over 200 news headlines. Furthermore, given the computational complexity of the problem, few algorithms would be able to perform well far beyond that point. The values of n used in our experiments are { 6, 7, 8, 9, 15, 20, 30, 40, 50, 70, 100, 120, 150, 180 }. For min aggregation, we stop at $n = 70$.

We also tested a "sampling" baseline that computes the neutrality of many random orderings and selects the best one. To enable a fair comparison, under avg aggregation, we set it to run for approximately the amount of time that ApproxMat takes, and under min aggregation, the amount of time that the heuristic method takes. We can see that our algorithms perform significantly better than the sampling baseline. Next, the execution times confirm that ApproxCC is significantly faster than ApproxMat ($O(n^3)$ vs. $O(n^4)$). Finally, we can see just how slow the brute-force method is — it is infeasible to run it for $n > 9$ in the experiments.

In addition to confirming the theoretical time complexities, we learn from the results that ApproxCC performs slightly better than ApproxMat. It is unclear why this is the case, but it seems to consistently compute slightly better orderings. We can also see that our algorithms are at or near optimal for small n. We cannot make any conclusions about their optimality for larger n since we are unable to compute the solution by brute force for larger n, but we expect that they are near optimal.

The experimental results also show that our algorithms are robust to changes in the data distribution. Neither the neutrality or execution time changes significantly as a result of the change in distribution (whereas the sampling baseline performs much worse on the synthetic data).

Early Stopping. Finally, we measured the effect of early stopping on the heuristic method. As mentioned earlier, the 2-opt subroutine has the "anytime property": it can be terminated at any time and still

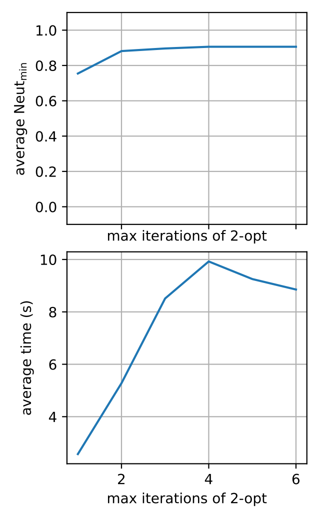


Figure 8: Early stopping in 2-opt on semi-synthetic data

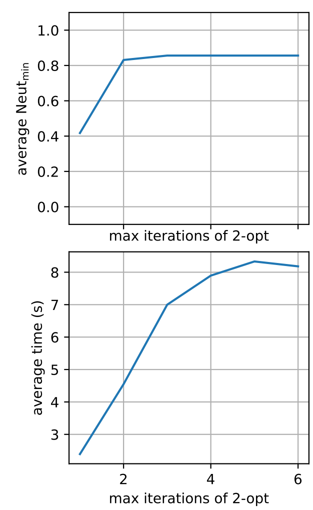


Figure 9: Early stopping in 2-opt on synthetic data

return a solution. For the other experiments, we simply ran 2-opt until it converged, but for this experiment, we ran it for a specified number of iterations (or terminated it if it converged early).

Using a fixed value of $n = 60$, for $1 \leq t \leq 6$, we ran the heuristic method but always terminated the 2-opt subroutine after at most t iterations. The results are shown in Figures 8 and 9. The neutrality increases as the number of allowed iterations increases, but plateaus after about 3 or 4 iterations. However, the neutrality is still relatively high after just 2 iterations, with the benefit of a significant reduction in computation time.

6.4 Existence of Priming (User Study)

The occurrence of priming has been well studied and has been analyzed in many related settings, including news consumption [14], social networks [2], job interviews [82], crowdsourcing [31], and annotation [25]. To verify that priming does indeed occur in our setting of viewing news headlines, we ran a user study (n=59).

We presented a test group and a control group (formed via convenience sampling) with a set of 9 fictional news headlines, and afterwards, we asked them their opinions of several people involved in the stories ("very negative", "negative", "positive", or "very positive"). The full set of headlines and images of the survey interface can be found in the appendix. The test group had the following pair of headlines placed next to each other in the ordering ($|s_j - s_i| = 1$); the control group had them separated ($|s_j - s_i| = 5$).

- "City's high school graduation rates at lowest in decades"
- "High school principal celebrates 10 years"

When surveyed at the end, after removing 6 responses that failed attention checks, 39% of the participants in the test group had formed a negative impression of the principal, compared to 16% in the control group. This difference is statistically significant (Boschloo's exact test, $\mathbf{p = 0.0337}$).

7 RELATED WORK

To the best of our knowledge, this paper is the first to study the effect of bias in news ordering. However, there are several areas of related work that we discuss next.

Media Bias. Neutrality in the *ordering* of news headlines is the focus of this paper. This is only one aspect of media bias, a large socio-technical problem with many dimensions [51]. Among many other facets, a well-recognized component of media bias is selection bias [18, 57, 63, 65]. In general, selection bias happens when the data selection process does not involve proper randomization [56, 81, 94]. A study by Bourgeois et al. [18] uses the predictability of the news coverage to measure selection bias.

Diversifying search results [3, 32] has been considered in efforts to reduce media bias [47, 61, 69, 87]. In particular, content spread in online social networks is affected by social bubbles and echo chambers [19, 21, 34, 74, 91], which significantly bias the spread of information. Diversifying news recommendations [13, 27, 46, 61, 69, 87] has been an effective technique for breaking echo chambers.

Computational Fact Checking. Whereas our work emphasizes the importance of the ordering of news stories, prior work in the literature focuses on the veracity of the content of said stories [49, 72, 99]. There have been remarkable advancements in fake news detection in the past decade [10, 22, 52–55, 95, 96]. Early fake news detection efforts include manual methods based on expert domain knowledge and crowdsourcing [52, 53]. Computational fact checking has since emerged, enabling automatic evaluation of claims [49, 72, 99]. These techniques heavily rely on natural language processing [67, 68], information retrieval [28], and graph theory [22]. Related work includes knowledge extraction from different data sources [29, 48, 75], data integration [4, 70, 84], and credibility evaluation [30, 40]. A bulk of the recent techniques used in fake news detection are based in supervised learning [59, 76, 78]. An increasing number of approaches are putting emphasis on the role of structured data [5, 11, 20, 79, 85, 89], as reflected in a special issue of the Data Engineering Bulletin [66].

Fair Ranking. Ranking news stories has been studied in the literature [26, 71, 86], but to the best of our knowledge, none of the existing work considers bias and neutrality in news ordering. Fair ranking is a recent line of work that studies ordering a set of items or individuals to satisfy some fairness constraints [9, 16, 77, 83]. At a high level, existing work is divided into score-based ranking [7–9, 97] and learning-to-rank and recommender systems [15, 43, 45, 98]. Despite the similarity in name, none of the existing work in this area can map onto our formulation of news ordering neutrality and is thus not useful in solving the problem proposed in this paper.

Traveling Salesman Problem. The (cycle) maximum traveling salesman problem has been studied since at least 1979 [41]. A survey on the maximum traveling salesman problem is given by Barvinok et al. [12], and the current state-of-the-art solution for it has a constant factor of 4/5 [35]. On the other hand, the path maximum traveling salesman problem, which we introduce in this paper to model avg aggregation, is not present in the literature to the best of our knowledge.

8 FINAL REMARKS

As this paper is opening up a new line of research in the fight against misinformation, there are numerous directions to explore.

Data Collection. A major challenge when evaluating news ordering neutrality is the collection of labels for each pair of news stories for constructing the POP function. One straightforward way is to survey the perceptions of the audience itself. However, it is not always possible to gain access to the audience's beliefs. The two main alternatives are crowdsourcing the labeling or having a domain expert provide the labels. The former is easy to scale but may result in inaccurate labels, while the latter results in accurate labels but is difficult to scale to large datasets. One promising potential approach to alleviate the difficulties of the data collection process is to train large language models to classify pairs of news stories.

Introducing Utility. While maximizing neutrality is important, a corporation's main goal is to maximize profits. It would be an interesting research direction to introduce a notion of utility and attempt simultaneous maximization of neutrality and utility.

Maximizing Neutrality. In this work, we restricted ourselves to a simple decay function to represent the problem in the form of a graph. One natural extension is to study the problem with more complex decay functions. For example, in layouts with multiple pages, it would be plausible to have a decay function with value 1 if two headlines are on the same page and 0 otherwise.

Another challenge is to try to find adversarial examples for the proposed algorithms. While we have proved approximation guarantees, we have not shown that they are tight. If adversarial examples are found, these bounds will be shown to be tight.

Finally, most of the algorithms for maximizing neutrality do not make any assumptions on the distribution of the data. It would be interesting to see if better guarantees or algorithms can be discovered for specific data distributions.

ACKNOWLEDGEMENTS

This work has been supported in parts by the UIC University Fellowship, the National Science Foundation (Grant No. 2107290), the ANR project ATTENTION (ANR-21-CE23-0037), and the Google Research Scholar Award. We would like to thank the reviewers for their constructive feedback and **Dr. H. V. Jagadish** for his invaluable contribution in problem identification.

REFERENCES

[1] Rishi Advani, Paolo Papotti, and Abolfazl Asudeh. 2023. *Maximizing Neutrality in News Ordering: Codebase*. https://doi.org/10.5281/zenodo.7955163

[2] Alexander Agadjanian, Jacob Cruger, Sydney House, Annie Huang, Noah Kanter, Celeste Kearney, Junghye Kim, Isabelle Leonaitis, Sarah Petroni, Leonardo Placeres, Morgan Quental, Henry Sanford, Cameron Skaff, Jennifer Wu, Lillian Zhao, and Brendan Nyhan. 2022. A platform penalty for news? How social media context can alter information credibility online. *Journal of Information Technology & Politics* 0, 0 (Aug. 2022), 1–11. https://doi.org/10.1080/19331681.2022.2105465

[3] Rakesh Agrawal, Sreenivas Gollapudi, Alan Halverson, and Samuel Ieong. 2009. Diversifying Search Results. In *WSDM*. ACM, New York, NY, USA, 5–14. https://doi.org/10.1145/1498759.1498766

[4] Yasser Altowim, Dmitri V. Kalashnikov, and Sharad Mehrotra. 2014. Progressive Approach to Relational Entity Resolution. *PVLDB* 7, 11 (July 2014), 999–1010. https://doi.org/10.14778/2732967.2732975

[5] Angelos-Christos Anadiotis, Oana Balalau, Théo Bouganim, Francesco Chimienti, Helena Galhardas, Mhd Yamen Haddad, Stéphane Horel, Ioana Manolescu, and Youssr Youssef. 2021. Empowering investigative journalism with graph-based heterogeneous data management. *Data Engineering Bulletin* 45, 3 (2021), 12–26.

[6] Esther M Arkin, Yi-Jen Chiang, Joseph SB Mitchell, Steven S Skiena, and Tae-Cheon Yang. 1999. On the maximum scatter traveling salesperson problem. *SIAM J. Comput.* 29, 2 (1999), 515–544.

[7] Abolfazl Asudeh and H. V. Jagadish. 2020. Fairly Evaluating and Scoring Items in a Data Set. *PVLDB* 13, 12 (Aug. 2020), 3445–3448. https://doi.org/10.14778/3415478.3415566

[8] Abolfazl Asudeh, H. V. Jagadish, Gerome Miklau, and Julia Stoyanovich. 2018. On obtaining stable rankings. *PVLDB* 12, 3 (2018), 237–250.

[9] Abolfazl Asudeh, H. V. Jagadish, Julia Stoyanovich, and Gautam Das. 2019. Designing Fair Ranking Schemes. In *SIGMOD* (Amsterdam, Netherlands) *(SIGMOD '19)*. ACM, New York, NY, USA, 1259–1276. https://doi.org/10.1145/3299869.3300079

[10] Abolfazl Asudeh, H. V. Jagadish, You (Will) Wu, and Cong Yu. 2020. On Detecting Cherry-Picked Trendlines. *PVLDB* 13, 6 (Feb. 2020), 939–952. https://doi.org/10.14778/3380750.3380762

[11] Abolfazl Asudeh, You Will Wu, Cong Yu, and H. V. Jagadish. 2021. Perturbation-based Detection and Resolution of Cherry-picking. *Data Engineering Bulletin* 45, 3 (2021), 52–65.

[12] Alexander Barvinok, Edward Kh. Gimadi, and Anatoliy I. Serdyukov. 2007. The Maximum TSP. In *The Traveling Salesman Problem and Its Variations*, Gregory Gutin and Abraham P. Punnen (Eds.). Springer US, Boston, MA, 585–607. https://doi.org/10.1007/0-306-48213-4_12

[13] Michael Bauer. 2021. *Diversification in News Recommendation*. Ph. D. Dissertation. Wien.

[14] Susanne E. Baumgartner and Werner Wirth. 2012. Affective Priming During the Processing of News Articles. *Media Psychology* 15, 1 (2012), 1–18. https://doi.org/10.1080/15213269.2011.648535

[15] Alex Beutel, Jilin Chen, Tulsee Doshi, Hai Qian, Li Wei, Yi Wu, Lukasz Heldt, Zhe Zhao, Lichan Hong, Ed H. Chi, and Cristos Goodrow. 2019. Fairness in Recommendation Ranking through Pairwise Comparisons. In *SIGKDD* (Anchorage, AK, USA) *(KDD '19)*. ACM, New York, NY, USA, 2212–2220. https://doi.org/10.1145/3292500.3330745

[16] Asia J. Biega, Krishna P. Gummadi, and Gerhard Weikum. 2018. Equity of Attention: Amortizing Individual Fairness in Rankings. In *SIGIR* (Ann Arbor, MI, USA) *(SIGIR '18)*. ACM, New York, NY, USA, 405–414. https://doi.org/10.1145/3209978.3210063

[17] Markus Bläser and Bodo Manthey. 2005. Approximating Maximum Weight Cycle Covers in Directed Graphs with Weights Zero and One. *Algorithmica* 42, 2 (2005), 121–139.

[18] Dylan Bourgeois, Jérémie Rappaz, and Karl Aberer. 2018. Selection Bias in News Coverage: Learning It, Fighting It. In *WWW* (Lyon, France) *(WWW '18)*. WWW, Republic and Canton of Geneva, CHE, 535–543. https://doi.org/10.1145/3184558.3188724

[19] Axel Bruns. 2019. *Are Filter Bubbles Real?* John Wiley & Sons.

[20] Mingke Chai, Zihui Gu, Xiaoman Zhao, Ju Fan, and Xiaoyong Du. 2021. TFV: A Framework for Table-Based Fact Verification. *Data Engineering Bulletin* 45, 3 (2021), 39–51.

[21] Matteo Cinelli, Gianmarco De Francisci Morales, Alessandro Galeazzi, Walter Quattrociocchi, and Michele Starnini. 2021. The echo chamber effect on social media. *PNAS* 118, 9 (2021), e2023301118. https://doi.org/10.1073/pnas.2023301118

[22] Sarah Cohen, James T. Hamilton, and Fred Turner. 2011. Computational Journalism. *Commun. ACM* 54, 10 (Oct. 2011), 66–71. https://doi.org/10.1145/2001269.2001288

[23] Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein. 2001. *Introduction To Algorithms*. MIT Press.

[24] G. A. Croes. 1958. A Method for Solving Traveling-Salesman Problems. *Operations Research* 6, 6 (1958), 791–812. http://www.jstor.org/stable/167074

[25] Tadele T. Damessie, J. Shane Culpepper, Jaewon Kim, and Falk Scholer. 2018. Presentation Ordering Effects On Assessor Agreement. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management* (Torino, Italy) *(CIKM '18)*. ACM, New York, NY, USA, 723–732. https://doi.org/10.1145/3269206.3271750

[26] Gianna M. Del Corso, Antonio Gullí, and Francesco Romani. 2005. Ranking a Stream of News. In *WWW* (Chiba, Japan) *(WWW '05)*. ACM, New York, NY, USA, 97–106. https://doi.org/10.1145/1060745.1060764

[27] Maunendra Sankar Desarkar and Neha Shinde. 2014. Diversification in news recommendation for privacy concerned users. In *DSAA*. IEEE, Shanghai, China, 135–141. https://doi.org/10.1109/DSAA.2014.7058064

[28] AnHai Doan, Alon Halevy, and Zachary Ives. 2012. *Principles of data integration*. Elsevier, Waltham, MA. https://doi.org/10.1016/C2011-0-06130-6

[29] Xin Dong, Evgeniy Gabrilovich, Geremy Heitz, Wilko Horn, Ni Lao, Kevin Murphy, Thomas Strohmann, Shaohua Sun, and Wei Zhang. 2014. Knowledge Vault: A Web-Scale Approach to Probabilistic Knowledge Fusion. In *SIGKDD* (New York, New York, USA) *(KDD '14)*. ACM, New York, NY, USA, 601–610. https://doi.org/10.1145/2623330.2623623

[30] Xin Luna Dong, Evgeniy Gabrilovich, Kevin Murphy, Van Dang, Wilko Horn, Camillo Lugaresi, Shaohua Sun, and Wei Zhang. 2015. Knowledge-Based Trust: Estimating the Trustworthiness of Web Sources. *PVLDB* 8, 9 (May 2015), 938–949. https://doi.org/10.14778/2777598.2777603

[31] Tim Draws, Alisa Rieger, Oana Inel, Ujwal Gadiraju, and Nava Tintarev. 2021. A Checklist to Combat Cognitive Biases in Crowdsourcing. *Proceedings of the AAAI Conference on Human Computation and Crowdsourcing* 9, 1 (Oct. 2021), 48–59. https://doi.org/10.1609/hcomp.v9i1.18939

[32] Marina Drosou, H. V. Jagadish, Evaggelia Pitoura, and Julia Stoyanovich. 2017. Diversity in Big Data: A Review. *Big Data* 5, 2 (2017), 73–84. https://doi.org/10.1089/big.2016.0054

[33] Ran Duan and Seth Pettie. 2014. Linear-Time Approximation for Maximum Weight Matching. *J. ACM* 61, 1, Article 1 (Jan. 2014), 23 pages. https://doi.org/10.1145/2529989

[34] Elizabeth Dubois and Grant Blank. 2018. The echo chamber is overstated: the moderating effect of political interest and diverse media. *Information, Communication & Society* 21, 5 (2018), 729–745. https://doi.org/10.1080/1369118X.2018.1428656

[35] Szymon Dudycz, Jan Marcinkowski, Katarzyna Paluch, and Bartosz Rybicki. 2017. A 4/5 - Approximation Algorithm for the Maximum Traveling Salesman Problem. In *Integer Programming and Combinatorial Optimization*, Friedrich Eisenbrand and Jochen Koenemann (Eds.). Springer International Publishing, Cham, 173–185.

[36] Richard Durstenfeld. 1964. Algorithm 235: Random Permutation. *Commun. ACM* 7, 7 (July 1964), 420. https://doi.org/10.1145/364520.364540

[37] Jack Edmonds. 1965. Paths, Trees, and Flowers. *Canadian Journal of Mathematics* 17 (1965), 449–467. https://doi.org/10.4153/CJM-1965-045-4

[38] Jack Edmonds and Richard M. Karp. 1972. Theoretical Improvements in Algorithmic Efficiency for Network Flow Problems. *J. ACM* 19, 2 (April 1972), 248–264. https://doi.org/10.1145/321694.321699

[39] David Eppstein. 2013. Partition a graph into node-disjoint cycles. Theoretical Computer Science Stack Exchange. https://cstheory.stackexchange.com/q/8570 (version: 2013-11-19).

[40] Diego Esteves, Aniketh Janardhan Reddy, Piyush Chawla, and Jens Lehmann. 2018. Belittling the Source: Trustworthiness Indicators to Obfuscate Fake News on the Web. In *FEVER*. Association for Computational Linguistics, Brussels, Belgium, 50–59. https://doi.org/10.18653/v1/W18-5508

[41] M. L. Fisher, G. L. Nemhauser, and L. A. Wolsey. 1979. An Analysis of Approximations for Finding a Maximum Weight Hamiltonian Circuit. *Operations Research* 27, 4 (Aug. 1979), 799–809. https://doi.org/10.1287/opre.27.4.799

[42] Ronald Aylmer Fisher and Frank Yates. 1953. *Statistical tables for biological, agricultural and medical research*. Hafner Publishing Company, New York.

[43] Ruoyuan Gao and Chirag Shah. 2020. Toward creating a fairer ranking in search engine results. *Information Processing & Management* 57, 1 (2020), 102138. https://doi.org/10.1016/j.ipm.2019.102138

[44] Michael R. Garey and David S. Johnson. 1979. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman & Co., New York, NY, USA.

[45] Sahin Cem Geyik, Stuart Ambler, and Krishnaram Kenthapadi. 2019. Fairness-Aware Ranking in Search & Recommendation Systems with Application to LinkedIn Talent Search. In *SIGKDD* (Anchorage, AK, USA) *(KDD '19)*. ACM, New York, NY, USA, 2221–2231. https://doi.org/10.1145/3292500.3330691

[46] Alireza Gharahighehi and Celine Vens. 2021. Diversification in session-based news recommender systems. *Personal and Ubiquitous Computing* 27, 1 (2021), 1–11.

[47] Giorgos Giannopoulos, Marios Koniaris, Ingmar Weber, Alejandro Jaimes, and Timos Sellis. 2015. Algorithms and criteria for diversification of news article comments. *Journal of Intelligent Information Systems* 44 (2015), 1–47. https://doi.org/10.1007/s10844-014-0328-1

[48] Ralph Grishman. 2015. Information extraction. *IEEE Intelligent Systems* 30, 5 (2015), 8–15.

[49] Zhijiang Guo, Michael Schlichtkrull, and Andreas Vlachos. 2022. A Survey on Automated Fact-Checking. *Transactions of the ACL* 10 (Feb. 2022), 178–206. https://doi.org/10.1162/tacl_a_00454

[50] Aric A. Hagberg, Daniel A. Schult, and Pieter J. Swart. 2008. Exploring Network

Structure, Dynamics, and Function using NetworkX. In *Proceedings of the 7th Python in Science Conference*, Gaël Varoquaux, Travis Vaught, and Jarrod Millman (Eds.). SciPy, Pasadena, CA, USA, 11–15.

[51] Felix Hamborg, Karsten Donnay, and Bela Gipp. 2019. Automated identification of media bias in news articles: an interdisciplinary literature review. *International Journal on Digital Libraries* 20, 4 (2019), 391–415.

[52] Naeemul Hassan, Fatma Arslan, Chengkai Li, and Mark Tremayne. 2017. Toward Automated Fact-Checking: Detecting Check-Worthy Factual Claims by ClaimBuster. In *SIGKDD* (Halifax, NS, Canada) *(KDD '17)*. ACM, New York, NY, USA, 1803–1812. https://doi.org/10.1145/3097983.3098131

[53] Naeemul Hassan, Chengkai Li, and Mark Tremayne. 2015. Detecting Check-Worthy Factual Claims in Presidential Debates. In *CIKM* (Melbourne, Australia) *(CIKM '15)*. ACM, New York, NY, USA, 1835–1838. https://doi.org/10.1145/2806416.2806652

[54] Naeemul Hassan, Afroza Sultana, You Wu, Gensheng Zhang, Chengkai Li, Jun Yang, and Cong Yu. 2014. Data in, Fact out: Automated Monitoring of Facts by FactWatcher. *PVLDB* 7, 13 (Aug. 2014), 1557–1560. https://doi.org/10.14778/2733004.2733029

[55] Naeemul Hassan, Gensheng Zhang, Fatma Arslan, Josue Caraballo, Damian Jimenez, Siddhant Gawsane, Shohedul Hasan, Minumol Joseph, Aaditya Kulkarni, Anil Kumar Nayak, Vikas Sable, Chengkai Li, and Mark Tremayne. 2017. ClaimBuster: The First-Ever End-to-End Fact-Checking System. *PVLDB* 10, 12 (Aug. 2017), 1945–1948. https://doi.org/10.14778/3137765.3137815

[56] James Heckman. 1990. Varieties of Selection Bias. *The American Economic Review* 80, 2 (1990), 313–318. http://www.jstor.org/stable/2006591

[57] Peter Hocke. 1996. *Determining the selection bias in local and national newspaper reports on protest events*. WZB, Berlin, Germany.

[58] Ata Kabán. 2012. Non-parametric detection of meaningless distances in high dimensional data. *Statistics and Computing* 22, 2 (01 March 2012), 375–385. https://doi.org/10.1007/s11222-011-9229-0

[59] Rohit Kumar Kaliyar, Anurag Goswami, Pratik Narang, and Soumendu Sinha. 2020. FNDNet – A deep convolutional neural network for fake news detection. *Cognitive Systems Research* 61 (2020), 32–44. https://doi.org/10.1016/j.cogsys.2019.12.005

[60] Richard M. Karp. 1980. An algorithm to solve the m × n assignment problem in expected time O(mn log n). *Networks* 10, 2 (1980), 143–152. https://doi.org/10.1002/net.3230100205

[61] Brent Kitchens, Steven L Johnson, and Peter Gray. 2020. Understanding Echo Chambers and Filter Bubbles: The Impact of Social Media on Diversification and Partisan Shifts in News Consumption. *MIS Quarterly* 44, 4 (2020), 1619–1649.

[62] Jon Kleinberg and Eva Tardos. 2005. *Algorithm Design*. Addison-Wesley Longman Publishing Co., Inc., USA.

[63] Claudia Kozman and Jad Melki. 2022. Selection Bias of News on Social Media: The Role of Selective Sharing and Avoidance During the Lebanon Uprising. *International Journal of Communication* 16 (2022), 21. https://ijoc.org/index.php/ijoc/article/view/18811

[64] John LaRusic and Abraham P. Punnen. 2014. The asymmetric bottleneck traveling salesman problem: Algorithms, complexity and empirical analysis. *Computers & Operations Research* 43 (2014), 20–35. https://doi.org/10.1016/j.cor.2013.08.005

[65] Konstantina Lazaridou and Ralf Krestel. 2016. Identifying political bias in news articles. *Bulletin of the IEEE TCDL* 12 (Nov. 2016), 12 pages. Issue 2.

[66] Chengkai Li and Jun Yang (Eds.). 2021. *Special Issue on Data Engineering Challenges in Combating Misinformation*. Data Engineering Bulletin, Vol. 45, 3. TCDE.

[67] Yunyao Li, Ishan Chaudhuri, Huahai Yang, Satinder Singh, and H. V. Jagadish. 2007. DaNaLIX: A Domain-Adaptive Natural Language Interface for Querying XML. In *SIGMOD* (Beijing, China) *(SIGMOD '07)*. ACM, New York, NY, USA, 1165–1168. https://doi.org/10.1145/1247480.1247643

[68] Yunyao Li, Huahai Yang, and H. V. Jagadish. 2006. Constructing a Generic Natural Language Interface for an XML Database. In *EDBT*, Yannis Ioannidis, Marc H. Scholl, Joachim W. Schmidt, Florian Matthes, Mike Hatzopoulos, Klemens Boehm, Alfons Kemper, Torsten Grust, and Christian Boehm (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 737–754.

[69] Gabriel Machado Lunardi. 2019. Representing the Filter Bubble: Towards a Model to Diversification in News. In *Advances in Conceptual Modeling*, Giancarlo Guizzardi, Frederik Gailly, and Rita Suzana Pitangueira Maciel (Eds.). Springer International Publishing, Cham, 239–246.

[70] Amr Magdy and Nayer Wanas. 2010. Web-Based Statistical Fact Checking of Textual Documents. In *SMUC* (Toronto, ON, Canada) *(SMUC '10)*. ACM, New York, NY, USA, 103–110. https://doi.org/10.1145/1871985.1872002

[71] Richard M. C. McCreadie, Craig Macdonald, and Iadh Ounis. 2010. News Article Ranking: Leveraging the Wisdom of Bloggers. In *Adaptivity, Personalization and Fusion of Heterogeneous Information* (Paris, France) *(RIAO '10)*. LE CENTRE DE HAUTES ETUDES INTERNATIONALES D'INFORMATIQUE DOCUMENTAIRE, Paris, France, 40–48.

[72] Preslav Nakov, David Corney, Maram Hasanain, Firoj Alam, Tamer Elsayed, Alberto Barrón-Cedeño, Paolo Papotti, Shaden Shaar, and Giovanni Da San Martino. 2021. Automated Fact-Checking for Assisting Human Fact-Checkers. In *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI-21*, Zhi-Hua Zhou (Ed.). International Joint Conferences on Artificial Intelligence Organization, Montreal, Canada, 4551–4558. https://doi.org/10.24963/ijcai.2021/619 Survey Track.

[73] Andrei Nikolaev and Anna Kozlova. 2021. Hamiltonian decomposition and verifying vertex adjacency in 1-skeleton of the traveling salesperson polytope by variable neighborhood search. *Journal of Combinatorial Optimization* 42, 2 (01 Aug. 2021), 212–230. https://doi.org/10.1007/s10878-020-00652-7

[74] Dimitar Nikolov, Diego FM Oliveira, Alessandro Flammini, and Filippo Menczer. 2015. Measuring online social bubbles. *PeerJ computer science* 1 (Dec. 2015), e38. https://doi.org/10.7717/peerj-cs.38

[75] Sachin Pawar, Girish K. Palshikar, and Pushpak Bhattacharyya. 2017. Relation Extraction : A Survey. https://doi.org/10.48550/ARXIV.1712.05191

[76] Verónica Pérez-Rosas, Bennett Kleinberg, Alexandra Lefevre, and Rada Mihalcea. 2018. Automatic Detection of Fake News. In *ACL*. Association for Computational Linguistics, Santa Fe, New Mexico, USA, 3391–3401. https://aclanthology.org/C18-1287

[77] Evaggelia Pitoura, Kostas Stefanidis, and Georgia Koutrika. 2022. Fairness in rankings and recommendations: an overview. *The VLDB Journal* 31 (2022), 431–458. https://doi.org/10.1007/s00778-021-00697-y

[78] Julio C. S. Reis, André Correia, Fabrício Murai, Adriano Veloso, and Fabrício Benevenuto. 2019. Supervised Learning for Fake News Detection. *IEEE Intelligent Systems* 34, 2 (March 2019), 76–81. https://doi.org/10.1109/MIS.2019.2899143

[79] Mohammed Saeed and Paolo Papotti. 2021. Factchecking statistical claims with tables. *Data Engineering Bulletin* 45, 3 (2021), 27–38.

[80] John G. Saw, Mark C. K. Yang, and Tse Chin Mo. 1984. Chebyshev Inequality with Estimated Mean and Variance. *The American Statistician* 38, 2 (1984), 130–132. http://www.jstor.org/stable/2683249

[81] Nima Shahbazi, Yin Lin, Abolfazl Asudeh, and H. V. Jagadish. 2022. A Survey on Techniques for Identifying and Resolving Representation Bias in Data. https://doi.org/10.48550/ARXIV.2203.11852

[82] Uri Simonsohn and Francesca Gino. 2013. Daily Horizons: Evidence of Narrow Bracketing in Judgment From 10 Years of M.B.A. Admissions Interviews. *Psychological Science* 24, 2 (2013), 219–224. https://doi.org/10.1177/0956797612459762

[83] Ashudeep Singh and Thorsten Joachims. 2018. Fairness of Exposure in Rankings. In *SIGKDD* (London, United Kingdom) *(KDD '18)*. ACM, New York, NY, USA, 2219–2228. https://doi.org/10.1145/3219819.3220088

[84] Rebecca C. Steorts, Rob Hall, and Stephen E. Fienberg. 2016. A Bayesian Approach to Graphical Record Linkage and Deduplication. *J. Amer. Statist. Assoc.* 111, 516 (2016), 1660–1672. https://doi.org/10.1080/01621459.2015.1105807

[85] Matthew Sumpter and Giovanni Luca Ciampaglia. 2021. Preserving the Integrity and Credibility of the Online Information Ecosystem. *Data Engineering Bulletin* 45, 3 (2021), 4–11.

[86] Alexandru Tatar, Panayotis Antoniadis, Marcelo Dias de Amorim, and Serge Fdida. 2014. From popularity prediction to ranking online news. *Social Network Analysis and Mining* 4, 1 (2014), 12 pages. https://doi.org/10.1007/s13278-014-0174-8

[87] Nava Tintarev, Emily Sullivan, Dror Guldin, Sihang Qiu, and Daan Odjik. 2018. Same, Same, but Different: Algorithmic Diversification of Viewpoints in News. In *UMAP* (Singapore, Singapore) *(UMAP '18)*. ACM, New York, NY, USA, 7–13. https://doi.org/10.1145/3213586.3226203

[88] N. Tomizawa. 1971. On some techniques useful for solution of transportation network problems. *Networks* 1, 2 (1971), 173–194. https://doi.org/10.1002/net.3230010206

[89] Immanuel Trummer. 2021. WebChecker: Towards an Infrastructure for Efficient Misinformation Detection at Web Scale. *Data Engineering Bulletin* 45, 3 (2021), 66–77.

[90] W. T. Tutte. 1954. A Short Proof of the Factor Theorem for Finite Graphs. *Canadian Journal of Mathematics* 6 (1954), 347–352. https://doi.org/10.4153/CJM-1954-033-3

[91] Cristian Vaccari and Augusto Valeriani. 2021. *Outside the Bubble: Social Media and Political Participation in Western Democracies*. Oxford University Press. https://doi.org/10.1093/oso/9780190858476.001.0001

[92] Pandiri Venkatesh, Alok Singh, and Rammohan Mallipeddi. 2019. A Multi-Start Iterated Local Search Algorithm for the Maximum Scatter Traveling Salesman Problem. In *2019 IEEE Congress on Evolutionary Computation (CEC)*. IEEE, Wellington, New Zealand, 1390–1397. https://doi.org/10.1109/CEC.2019.8790018

[93] Pauli Virtanen, Ralf Gommers, Travis E. Oliphant, Matt Haberland, Tyler Reddy, David Cournapeau, Evgeni Burovski, Pearu Peterson, Warren Weckesser, Jonathan Bright, Stéfan J. van der Walt, Matthew Brett, Joshua Wilson, K. Jarrod Millman, Nikolay Mayorov, Andrew R. J. Nelson, Eric Jones, Robert Kern, Eric Larson, C J Carey, İlhan Polat, Yu Feng, Eric W. Moore, Jake VanderPlas, Denis Laxalde, Josef Perktold, Robert Cimrman, Ian Henriksen, E. A. Quintero, Charles R. Harris, Anne M. Archibald, Antônio H. Ribeiro, Fabian Pedregosa, Paul van Mulbregt, and SciPy 1.0 Contributors. 2020. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods* 17 (2020), 261–272. https://doi.org/10.1038/s41592-019-0686-2

[94] Christopher Winship and Robert D. Mare. 1992. Models for Sample Selection Bias. *Annual Review of Sociology* 18, 1 (1992), 327–350. https://doi.org/10.1146/annurev.so.18.080192.001551

[95] You Wu, Pankaj K. Agarwal, Chengkai Li, Jun Yang, and Cong Yu. 2014. Toward Computational Fact-Checking. *PVLDB* 7, 7 (2014), 589–600. https://doi.org/10.14778/2732286.2732295
[96] You Wu, Pankaj K. Agarwal, Chengkai Li, Jun Yang, and Cong Yu. 2017. Computational Fact Checking through Query Perturbations. *TODS* 42, 1 (2017), 41 pages.
[97] Meike Zehlike, Ke Yang, and Julia Stoyanovich. 2022. Fairness in Ranking, Part I: Score-Based Ranking. *ACM Comput. Surv.* 55, 6, Article 118 (Dec. 2022), 36 pages. https://doi.org/10.1145/3533379
[98] Meike Zehlike, Ke Yang, and Julia Stoyanovich. 2022. Fairness in Ranking, Part II: Learning-to-Rank and Recommender Systems. *ACM Comput. Surv.* 55, 6, Article 117 (Dec. 2022), 41 pages. https://doi.org/10.1145/3533380
[99] Xinyi Zhou and Reza Zafarani. 2020. A Survey of Fake News: Fundamental Theories, Detection Methods, and Opportunities. *ACM Comput. Surv.* 53, 5, Article 109 (Sept. 2020), 40 pages. https://doi.org/10.1145/3395046

A DIRECTIONALITY

If we model the POP function as being sensitive to the order of the two stories taken as input, all but one of the results in this paper still hold. Approx3CC cannot be adapted for the directed setting because it relies on finding a max-weight 3-cycle cover and doing this is NP-hard in a directed graph [44, GT13]. Fortunately, this algorithm is of theoretical interest only, and all other algorithms can be easily adapted, as-is, for the directed setting.

B PSEUDOCODE

The pseudocode for approximately solving PathMaxTSP via iterated matching is shown below.

procedure ApproxMat($G = (V, E)$)
    $k \leftarrow 0$; $G_k \leftarrow G$; $size \leftarrow n$
    while $size > 1$ **do**
        Compute a max-weight matching M in G_k.
        Construct a graph $G_{k+1} = (V_{k+1}, E_{k+1})$ as follows.
        for all $((u_1, \ldots, u_a), (v_1, \ldots, v_b)) \in M$ **do**
            Consider the edge e in E with highest weight from the set $\{ (u_1, v_1), (u_1, v_b), (u_a, v_1), (u_a, v_b) \}$.
            if $e = (u_1, v_1)$: Add $(u_a, \ldots, u_1, v_1, \ldots, v_b)$ to V_{k+1}.
            if $e = (u_1, v_b)$: Add $(u_a, \ldots, u_1, v_b, \ldots, v_1)$ to V_{k+1}.
            if $e = (u_a, v_1)$: Add $(u_1, \ldots, u_a, v_1, \ldots, v_b)$ to V_{k+1}.
            if $e = (u_a, v_b)$: Add $(u_1, \ldots, u_a, v_b, \ldots, v_1)$ to V_{k+1}.
        If $|V_k|$ is odd, add the remaining vertex in V_k to V_{k+1}.
        for all pairs $(u_1, \ldots, u_a), (v_1, \ldots, v_b)$ in V_{k+1} **do**
            Let w be the weight of the edge in E with highest weight from the set $\{ (u_1, v_1), (u_1, v_b), (u_a, v_1), (u_a, v_b) \}$.
            Add the edge $((u_1, \ldots, u_a), (v_1, \ldots, v_b))$ to E_{k+1} with weight w.
        $size \leftarrow |V_{k+1}|$; $k \leftarrow k + 1$
    return the Hamiltonian path $v_1, \ldots, v_n$ in G where $(v_1, \ldots, v_n)$ is the sole vertex in V_k.

The pseudocode for approximately solving PathMaxTSP via 3-cycle cover is shown below.

procedure Approx3CC($G = (V, E)$)
    Construct a graph $G' = (V', E')$ as follows.
    Let $w_{\max}$ be the weight of the edge with max weight in E.
    For each vertex $v \in V$, add a complete bipartite graph $G_v = K_{n-1,n-3}$ to G' with each edge having weight $w_{\max}$.
    Let $G_{v,R}$ denote the side of the bipartition with $n-1$ vertices.
    For each edge $(u, v) \in E$ with weight w, add an edge with weight w between a pair of vertices from $G_{u,R}$ and $G_{v,R}$ such that for all $x \in V$, each vertex in $G_{x,R}$ has degree exactly $n - 2$.
    Compute a max-weight matching in G'. Let C be the corresponding 3-cycle cover in G.
    For each cycle $c \in C$, remove the min-weight edge.
    Arbitrarily join the paths together.
    return the resulting Hamiltonian path.

C BINARY AVERAGE AGGREGATION

If we add the condition that the POP function, C, takes values in $\{ 0, 1 \}$, we can prove additional results. This can be interpreted as assuming that any pair of stories has either zero risk of giving rise to opinion priming or is absolutely certain to do so.

Definition C.1 (PathMaxTSP(0,1)). Given a graph G such that all edges have binary weight, *PathMaxTSP(0,1)* is the problem of finding a Hamiltonian path with maximum total weight.

We also introduce the cycle variant, a problem not yet addressed in the literature:

Definition C.2 (MaxTSP(0,1)). Given a graph G such that all edges have binary weight, *MaxTSP(0,1)* is the problem of finding a Hamiltonian cycle with maximum total weight.

Lemma C.3. *PathMaxTSP(0,1) is NP-hard.*

Proof. Consider a graph $G = (V, E)$ with the property that all edges have binary weight. Then, define $G' = (V, E')$ to be the unweighted graph with the same vertex set and an edge between any two vertices u and v if and only if (u, v) has weight 1 in E.

If a solution to PathMaxTSP(0,1) in G has weight $n - 1$, then there must exist a Hamiltonian path in G'. If a solution to PathMaxTSP(0,1) in G has weight less than $n-1$, then there cannot exist a Hamiltonian path in G'. Given a solution to PathMaxTSP(0,1), we can decide HamPath in constant time.

Suppose for contradiction that PathMaxTSP(0,1) is not NP-hard. Then, by our previous claim, HamPath is not NP-hard. This is a contradiction, since HamPath is known to be NP-hard. Therefore, PathMaxTSP(0,1) must be NP-hard. □

Proof of Theorem 4.2. Path MaxTSP(0,1) is a special case of PathMaxTSP. Thus, PathMaxTSP is also NP-hard. □

Since PathMaxTSP(0,1) is NP-hard, we seek an approximation algorithm for the problem. The simplest approach is to reduce it to another problem that already has a solution. The cycle variant of this problem, MaxTSP(0,1), which we introduced in this paper, has not yet been studied, but its generalization, MaxTSP, has been addressed in the literature; the current state-of-the-art algorithm has an approximation factor of 4/5 [35].

Theorem C.4. *Given an α-approximation for MaxTSP(0,1), we can compute an α-approximation for PathMaxTSP(0,1) in $O(n)$ time.*

Proof. Given a cycle that is an α-approximation for MaxTSP(0,1), remove the min-weight edge. If it has weight 1, the cycle has weight n and the resulting path has weight $n - 1$, which is the maximum weight possible for a path of length $n - 1$, so it must be an optimal solution. Henceforth, assume that the min-weight edge has weight 0. Let C be the weight of the cycle and $P = C$ the weight of the resulting path.

Let C^* be the weight of an optimal solution to MaxTSP(0,1). If we remove an edge, we have a path of weight at most C^*. If there was a path with greater weight, we could join the endpoints to form a cycle with weight greater than C^*, so the first path must be optimal. Let P^* be its weight. Then, we have $\frac{P}{P^*} = \frac{C}{P^*} >= \frac{C}{C^*} >= \alpha$.

Thus, the path we have constructed gives an α-approximation for PathMaxTSP(0,1).

It takes $O(n)$ time to find and remove the min-weight edge, so we have constructed the approximation in $O(n)$ time. □

Corollary C.5. *There is a 4/5-approximation algorithm for PathMaxTSP(0,1).*

Proof. There is a 4/5-approximation algorithm for MaxTSP(0,1). Thus, by Theorem C.4, we can construct a 4/5-approximation for PathMaxTSP(0,1). □

D AVERAGE AGGREGATION PROOFS

Proof of Theorem 4.5. We will show that the first iteration of ApproxMat gives us a 1/2-approximation. All future iterations cannot worsen the approximation, so we do not have to consider their effects. Let $e_1, e_2, \ldots, e_{n-1}$ be the sequence of edges in some solution to an instance of PathMaxTSP. Let W be the total weight of the path. The following sets of edges are matchings in the graph:

$$\{ e_1, e_3, \ldots \}, \{ e_2, e_4, \ldots \}$$

Consider the matching of greater total weight (breaking a tie arbitrarily). It must have weight at least $W/2$. Thus, the max-weight matching in the graph must have weight at least $W/2$. Finally, if we arbitrarily patch the edges together to form a Hamiltonian path, the resulting path also has weight at least $W/2$. This gives us a 1/2-approximation. □

Proof of Theorem 4.7. We show that the first iteration of ApproxCC gives us a 1/2-approximation. All future iterations cannot worsen the approximation, so we do not have to consider their effects. Let W be the weight of a max-weight Hamiltonian path in G. First, note that a max-weight Hamiltonian cycle must have weight at least W. Next, a Hamiltonian cycle is a cycle cover with 1 cycle, so the max-weight cycle cover constructed in the algorithm must have weight at least W.

Each cycle, by definition, has at least 2 edges. If we remove the min-weight edge from each cycle, we are removing at most half of the weight of each cycle. The resulting set of paths has total weight at least $W/2$. Accordingly, the constructed Hamiltonian path has weight at least $W/2$. This gives us a 1/2-approximation. □

Proof of Theorem 4.8. Assume $n \geq 3$ (it is trivial to solve $\textsc{Neutrality}_{\text{avg}}$ if $n < 3$). Let W be the weight of a max-weight Hamiltonian path in G. First, note that a max-weight Hamiltonian cycle must have weight at least W. Next, a Hamiltonian cycle is a 3-cycle cover with 1 cycle, so the max-weight cycle cover constructed in Approx3CC must have weight at least W.

Each cycle, by construction, has at least 3 edges. If we remove the min-weight edge from each cycle, we are removing at most one third of the weight of each cycle. The resulting set of paths has total weight at least $2W/3$. Accordingly, the constructed Hamiltonian path has weight at least $2W/3$. This gives us a 2/3-approximation. □

E USER STUDY HEADLINES

- "Mayor accused of mishandling city funds"
- "Library temporarily closing doors for renovations"
- "Navy veteran receives Silver Star Medal"
- "City's high school graduation rates at lowest in decades"
- "Residents urged to stay in their homes as temperatures reach -20s"
- "Local artist's work featured in popular downtown bar"
- "New bus routes added to accommodate increase in commuters"
- "Man charged with DUI after crashing into restaurant"
- "High school principal celebrates 10 years"

F USER STUDY INTERFACE

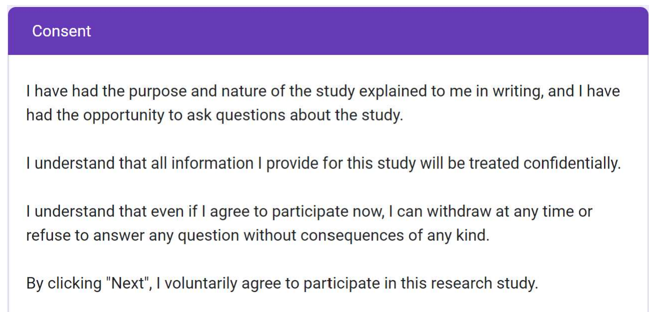


Figure 10: Consent page of the user study

Figure 11: Instructions for the user study

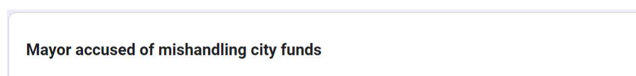


Figure 12: Example of a headline from the user study

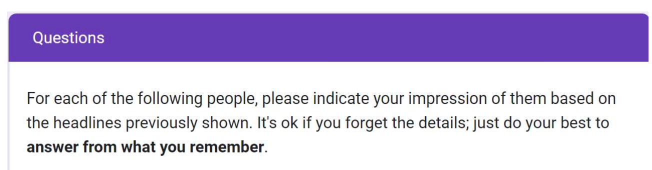


Figure 13: Prompt from the user study