an open access journal

Citation: van Bevern, R., Komusiewicz, C., Molter, H., Niedermeier, R., Sorge, M., & Walsh, T. (2020). *h*-Index manipulation by undoing merges. *Quantitative Science Studies*, *1*(4), 1529–1552. https://doi.org/10.1162/qss_a_00093

DOI:
https://doi.org/10.1162/qss_a_00093

Received: 11 November 2019
Accepted: 11 October 2020

Corresponding Author:
Hendrik Molter
h.molter@tu-berlin.de

Handling Editor:
Ludo Waltman

The MIT Press

RESEARCH ARTICLE

h-Index manipulation by undoing merges*

René van Bevern[1], Christian Komusiewicz[2], Hendrik Molter[3], Rolf Niedermeier[3], Manuel Sorge[4], and Toby Walsh[3]

[1]Department of Mechanics and Mathematics, Novosibirsk State University, Novosibirsk, Russian Federation
[2]Fachbereich Mathematik und Informatik, Philipps-Universität Marburg, Marburg, Germany
[3]Algorithmics and Computational Complexity, Fakultät IV, TU Berlin, Germany
[4]Institute of Informatics, University of Warsaw, Poland

Keywords: article splitting, citation graph, experimental algorithmics, Google Scholar profiles, NP-hard problems, parameterized complexity

ABSTRACT

The *h*-index is an important bibliographic measure used to assess the performance of researchers. Dutiful researchers merge different versions of their articles in their Google Scholar profile even though this can decrease their *h*-index. In this article, we study the manipulation of the *h*-index by undoing such merges. In contrast to manipulation by merging articles, such manipulation is harder to detect. We present numerous results on computational complexity (from linear-time algorithms to parameterized computational hardness results) and empirically indicate that at least small improvements of the *h*-index by splitting merged articles are unfortunately easily achievable.

1. INTRODUCTION

We suppose that an author has a publication profile, for example in Google Scholar, that consists of single articles and aims to increase her or his *h*-index[1] by merging articles. This will result in a new article with a potentially higher number of citations. The merging option is provided by Google Scholar to identify different versions of the same article, for example a journal version and its archived version.

Our main points of reference are three publications dealing with the manipulation of the *h*-index, particularly motivated by Google Scholar author profile manipulation (de Keijzer & Apt, 2013; Pavlou & Elkind, 2016; van Bevern, Komusiewicz, et al., 2016b). Indeed, we will closely follow the notation and concepts introduced by van Bevern et al. (2016b) and we refer to this work for discussion of related work concerning strategic self-citations to manipulate the *h*-index (Bartneck & Kokkelmans, 2011; Delgado López-Cózar, Robinson-García, & Torres-Salinas,

[*] An extended abstract of this article appeared in the proceedings of the 22nd European Conference on Artificial Intelligence (ECAI '16; Komusiewicz, van Bevern, et al., 2016a). This full version contains additional, corrected experimental results, and strengthened hardness results (Theorem 5). The following errors in the previously performed computational experiments were corrected: (a) The algorithm (Ramsey) for generating initially merged articles was previously not described accurately. The description is now more accurate and we consider additional algorithms to avoid bias in the generated instances. (b) Two authors from the ai10-2011 and ai10-2013 data sets with incomplete data have been used in the computational experiments; these authors are now omitted. (c) There were several technical errors in the code relating to the treatment of article and cluster identifiers of the crawled articles. This led to inconsistent instances and thus erroneous possible *h*-index increases. All of these errors have been corrected.

[1] The *h*-index of a researcher is the maximum number *h* such that he or she has at least *h* articles each cited at least *h* times (Hirsch, 2005).

2014; Vinkler, 2013), other citation indices (Egghe, 2006; Pavlou & Elkind, 2016; Woeginger, 2008), and manipulation in general (Faliszewski & Procaccia, 2010; Faliszewski, Hemaspaandra, & Hemaspaandra, 2010; Oravec, 2017). The main difference between this work and previous publications is that they focus on *merging* articles for increasing the *h*-index (Bodlaender & van Kreveld, 2015; de Keijzer & Apt, 2013; Pavlou & Elkind, 2016; van Bevern et al., 2016b) or other indices, such as *g*-index and the *i*10-index (Pavlou & Elkind, 2016), while we focus on *splitting*.

In the case of splitting, we assume that, most of the time, an author will maintain a correct profile in which all necessary merges are performed. Some of these merges may decrease the *h*-index. For instance, this can be the case when the two most cited papers are the conference and archived version of the same article. A very realistic scenario is that at certain times, for example when being evaluated by their dean[2], authors may temporarily undo some of these merges to artificially increase their *h*-index. A further point that distinguishes manipulation by splitting from manipulation by merging is that for merging it is easier to detect whether someone cheats too much. This can be done by looking at the titles of merged articles (van Bevern et al., 2016b). In contrast, it is much harder to prove that someone is manipulating by splitting; the manipulator can always claim to be too busy or that he or she does not know how to operate the profile.

The main theoretical conclusion from our work is that *h*-index manipulation by splitting merged articles[3] is typically computationally easier than manipulation by merging. Hence, undoing all merges and then merging from scratch might be computationally intractable in some cases, while, in contrast, computing an optimal splitting is computationally feasible. The only good news in terms of problem complexity (and, in a way, a recommendation) is that, if one were to use the citation measure "fusionCite" as defined by van Bevern et al. (2016b), then manipulation is computationally much harder than for the "unionCite" measure used by Google Scholar. In the practical part of our work, we experimented with data from Google Scholar profiles (van Bevern et al., 2016b).

1.1. Models for Splitting Articles

We consider the publication profile of an author and denote the articles in this profile by $W \subseteq V$, where V is the set of all articles. Following previous work (van Bevern et al., 2016b), we call these articles *atomic*. Merging articles yields a partition $\mathcal{P}$ of W in which each part $P \in \mathcal{P}$ with $|P| \geq 2$ is a *merged article*.

Given a partition $\mathcal{P}$ of W, the aim of splitting merged articles is to find a refined partition $\mathcal{R}$ of $\mathcal{P}$ with a larger *h*-index, where the *h-index of a partition* $\mathcal{P}$ is the largest number h such that there are at least h parts $P \in \mathcal{P}$ whose number $\mu(P)$ of citations is at least h. Herein, we have multiple possibilities of defining the number $\mu(P)$ of citations of an article in $\mathcal{P}$ (van Bevern et al., 2016b). The first one, sumCite(P), was introduced by de Keijzer and Apt (2013), and is simply the sum of the citations of each atomic article in P. Subsequently, van Bevern et al. (2016b) introduced the citation measures unionCite (used by Google Scholar), where we take the cardinality of the union of the citing atomic articles, and fusionCite, where we additionally remove self-citations of merged articles as well as duplicate citations between merged articles. In generic definitions, we denote these measures by μ (see Figure 1 for an illustration and Section 2 for the formal definitions). Note

[2] Lesk (2015) pointed out that the *h*-index is the modern equivalent of the old saying "Deans can't read, they can only count." He also remarked that the idea of "least publishable units" by dividing one's reports into multiple (short) papers has been around since the 1970s.

[3] Google Scholar allows authors to group different versions of an article. We call the resulting grouping a *merged article*. Google Scholar author profiles typically contain many merged articles (e.g., an arXiv version with a conference version and a journal version).

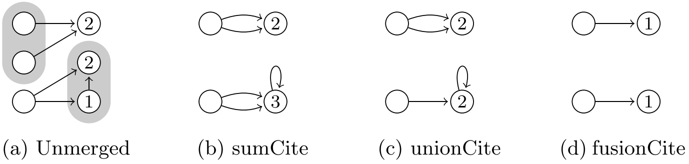

Figure 1. Vertices represent articles, arrows represent citations, and numbers are citation counts. The articles on a gray background in (a) have been merged in (b)–(d), and citation counts are given according to the measures sumCite, unionCite, and fusionCite, respectively. The arrows represent the citations counted by the corresponding measure.

that, to compute these citation measures, we need a *citation graph*: a directed graph whose vertices represent articles and in which an arc from a vertex u to a vertex v means that article u cites article v.

In this work, we introduce three different operations that may be used for undoing merges in a merged article a:

Atomizing: splitting a into all its atomic articles,
Extracting: splitting off a single atomic article from a, and
Dividing: splitting a into two parts arbitrarily.

See Figure 2 for an illustration of the three splitting operations. Note that the atomizing, extracting, and dividing operations are successively strictly more powerful in the sense that successively larger h-indices can be achieved. Google Scholar offers the extraction operation. Multiple applications of the extraction operation can, however, simulate atomizing and, together with merging, also dividing.

The three splitting operations lead to three problem variants, each taking as input a citation graph $D = (V, A)$, a set $W \subseteq V$ of articles belonging to the author, a partition $\mathcal{P}$ of W that defines already-merged articles, and a nonnegative integer h denoting the h-index to achieve. For $\mu \in$ {sumCite, unionCite, fusionCite}, we define the following problems.

Atomizing(μ)

Question: Is there a partition $\mathcal{R}$ of W such that

1. for each $R \in \mathcal{R}$ either $|R| = 1$ or there is a $P \in \mathcal{P}$ such that $R = P$,
2. the h-index of $\mathcal{R}$ with respect to μ is at least h?

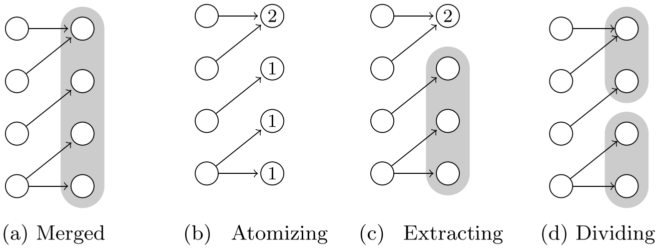

Figure 2. Vertices represent articles, arrows represent citations, and numbers are citation counts. The articles on a gray background have been merged in the initial profile (a) and correspond to remaining merged articles after applying one operation in (c) and (d). Each (merged) article has the same citation count, regardless of the used measure: sumCite, unionCite, and fusionCite.

EXTRACTING(μ)

Question: Is there a partition $\mathcal{R}$ of W such that

1. for each $R \in \mathcal{R}$ there is a $P \in \mathcal{P}$ such that $R \subseteq P$,
2. for each $P \in \mathcal{P}$ we have $|\{R \in \mathcal{R} \mid R \subset P \text{ and } |R| > 1\}| \leq 1$,
3. the h-index of $\mathcal{R}$ with respect to μ is at least h?

DIVIDING(μ)

Question: Is there a partition $\mathcal{R}$ of W such that

1. for each $R \in \mathcal{R}$ there is a $P \in \mathcal{P}$ such that $R \subseteq P$,
2. the h-index of $\mathcal{R}$ with respect to μ is at least h?

1.2. Conservative Splitting

We study for each of the problem variants an additional upper bound on the number of merged articles that are split. We call these variants *conservative*: If an insincere author would like to manipulate his or her profile temporarily, then he or she might prefer a manipulation that can be easily undone. To formally define CONSERVATIVE ATOMIZING, CONSERVATIVE EXTRACTING, and CONSERVATIVE DIVIDING, we add the following restriction to the partition $\mathcal{R}$: "the number $|\mathcal{P} \setminus \mathcal{R}|$ of changed articles is at most k."

A further motivation for the conservative variants is that, in a Google Scholar profile, an author can click on a merged article and tick a box for each atomic article that he or she wants to extract. As Google Scholar uses the unionCite measure (van Bevern et al., 2016b), Conservative Extracting(unionCite) thus corresponds closely to manipulating the Google Scholar h-index via few of the splitting operations available to the user.

1.3. Cautious Splitting

For each splitting operation, we also study an upper bound k on the number of split operations. Following our previous work (van Bevern et al., 2016a), we call this variant *cautious*. In the case of atomizing, conservativity and caution coincide, because exactly one operation is performed per changed article. Thus, we obtain two cautious problem variants: CAUTIOUS EXTRACTING and CAUTIOUS DIVIDING. For both we add the following restriction to the partition $\mathcal{R}$: "the number $|\mathcal{R}| - |\mathcal{P}|$ of extractions (or divisions, respectively) is at most k." In both variants we consider k to be part of the input.

1.4. Our Results

We investigate the parameterized computational complexity of our problem variants with respect to the parameters "the h-index h to achieve," and in the conservative case "the number k of modified merged articles," and in the cautious case "the number k of splitting operations." To put it briefly, the goal is to exploit potentially small parameter values (that is, special properties of the input instances) to gain efficient algorithms for problems that are in general computationally hard. In our context, the choice of the parameter h is motivated by the scenario that young researchers may have an incentive to increase their h-index and, because they are young, the h-index h to achieve is not very large. The conservative and cautious scenario tries to capture that the manipulation can easily be undone or is hard to detect, respectively. Hence, it is well motivated that the parameter k shall be small. Our theoretical (computational complexity classification) results are summarized in Table 1 (see Section 2 for further definitions). The measures sumCite and unionCite behave in basically the same

Table 1. Time complexity of manipulating the *h*-index by splitting operations (see Section 2 for definitions). For all FPT and W[1]-hardness results we also show NP-hardness

Problem	sumCite / unionCite	fusionCite
Atomizing	Linear (Theorem 1)	FPT† (Theorems 5 and 6)
Conservative A.	Linear (Theorem 1)	W[1]-h* (Theorem 7)
Extracting	Linear (Theorem 2)	NP-h$^{\odot}$ (Theorem 5)
Conservative E.	Linear (Theorem 2)	W[1]-h* (Corollary 1)
Cautious E.	Linear (Theorem 2)	W[1]-h* (Corollary 1)
Dividing	FPT† (Theorem 3)	NP-h$^{\odot}$ (Proposition 1)
Conservative D.	FPT†,‡ (Theorem 3)	W[1]-h* (Corollary 1)
Cautious D.	W[1]-h$^{\diamond,\odot}$ (Theorem 4)	W[1]-h* (Corollary 1)

† wrt. parameter h, the h-index to achieve.

$^{\diamond}$ wrt. parameter k, the number of operations.

* wrt. parameter $h + k + s$, where s is the largest number of articles merged into one.

‡ NP-hard even if $k = 1$ (Proposition 1).

$^{\odot}$ Parameterized complexity wrt. h open.

way. In particular, in the case of atomizing and extracting, manipulation is doable in linear time, while fusionCite mostly leads to (parameterized) intractability; that is, to high worst-case computational complexity. Moreover, the dividing operation (the most general one) seems to lead to computationally much harder problems than atomizing and extracting.

We performed computational experiments with real-world data (van Bevern et al., 2016b) and the mentioned linear-time algorithms, in particular for the case directly relevant to Google Scholar; that is, using the extraction operation and the unionCite measure. Our general findings are that increases of the *h*-index by one or two typically are easily achievable with few operations. The good news is that dramatic manipulation opportunities due to splitting are rare. They cannot be excluded, however, and they could be easily executed when relying on standard operations and measures (as used in Google Scholar). Working with fusionCite instead of the other two could substantially hamper manipulation.

2. PRELIMINARIES

Throughout this work, we use $n := |V|$ for the number of input articles and $m := |A|$ for the overall number of arcs in the input citation graph $D = (V, A)$. By $W \subseteq V$ we denote the articles in the author profile that we are manipulating. Let $\deg^{\text{in}}(v)$ denote the indegree of an article v in a citation graph $D = (V, A)$; that is, v's number of citations. Furthermore, let $N^{\text{in}}_{D}(v) := \{u \mid (u, v) \in A\}$ denote the set of articles that cite v and $N^{\text{in}}_{D-W}(v) := \{u \mid (u, v) \in A \wedge u \notin W\}$ be the set of articles outside W that cite v. For each part $P \in \mathcal{P}$, the following three measures for the number $\mu(P)$ of citations of P have been introduced (van Bevern et al., 2016b). They are illustrated in Figure 1. The measure

$$\text{sumCite}(P) := \sum_{v \in P} \deg^{\text{in}}(v)$$

defines the number of citations of a merged article P as the sum of the citations of the atomic articles it contains. This measure was proposed by de Keijzer and Apt (2013). In contrast, the measure

$$\text{unionCite}(P) := \left| \bigcup_{v \in P} N_D^{\text{in}}(v) \right|$$

defines the number of citations of a merged article P as the number of distinct atomic articles citing at least one atomic article in P. Google Scholar uses the unionCite measure (van Bevern et al., 2016b). The measure

$$\text{fusionCite}(P) := \left| \bigcup_{v \in P} N_{D-W}^{\text{in}}(v) \right| + \sum_{P' \in \mathcal{P} \setminus \{P\}} \begin{cases} 1 & \text{if } \exists v \in P' \exists w \in P : (v, w) \in A, \\ 0 & \text{otherwise} \end{cases}$$

is perhaps the most natural one: At most one citation of a part $P' \in \mathcal{P}$ to a part $P \in \mathcal{P}$ is counted; that is, we additionally remove duplicate citations between merged articles and self-citations of merged articles.

Our theoretical analysis is in the framework of parameterized complexity (Cygan, Fomin, et al., 2015; Downey & Fellows, 2013; Flum & Grohe, 2006; Niedermeier, 2006). That is, for those problems that are NP-hard, we study the influence of a *parameter*, an integer associated with the input, on the computational complexity. For a problem P, we seek to decide P using a *fixed-parameter algorithm*, an algorithm with running time $f(p) \cdot |I|^{O(1)}$, where I is the input and $f(p)$ is a computable function depending only on the parameter p. If such an algorithm exists, then P is *fixed-parameter tractable* (FPT) with respect to p. W[1]-hard parameterized problems presumably do not admit FPT algorithms. For example, the problem of finding an order-k clique in an undirected graph is known to be W[1]-hard for the parameter k. W[1]-hardness of a problem P parameterized by p can be shown via a *parameterized reduction* from a known W[1]-hard problem Q parameterized by q. That is, a reduction that runs in $f(q) \cdot n^{O(1)}$ time on input of size n with parameter q and produces instances that satisfy $p \leq f(q)$ for some function f.

3. SUMCITE AND UNIONCITE

In this section, we study the sumCite and unionCite measures. We provide linear-time algorithms for atomizing and extracting and analyze the parameterized complexity of dividing with respect to the number k of splits and the h-index h to achieve. In our results for sumCite and unionCite, we

Algorithm 1: Atomizing

```
Input: A citation graph $D = (V, A)$, a set $W \subseteq V$ of articles, a partition $\mathcal{P}$ of $W$, a nonnegative
       integer $h$ and a measure $\mu$.
Output: A partition $\mathcal{R}$ of $W$.
$\mathcal{R} \leftarrow \emptyset$
foreach $P \in \mathcal{P}$ do
    $\mathcal{A} \leftarrow$ Atomize($P$)
    if $\exists A \in \mathcal{A}$: $\mu(A) \geq h$ then $\mathcal{R} \leftarrow \mathcal{R} \cup \mathcal{A}$
    else $\mathcal{R} \leftarrow \mathcal{R} \cup \{P\}$
return $\mathcal{R}$
```

often tacitly use the observation that local changes to the merged articles do not influence the citations of other merged articles.

3.1. Manipulation by Atomizing

Recall that the atomizing operation splits a merged article into singletons and that, for the atomizing operation, the notions of *conservative* (touching few articles) and *cautious* (making few split operations) manipulation coincide and are thus both captured by Conservative Atomizing. Both Atomizing and Conservative Atomizing are solvable in linear time. Intuitively, it suffices to find the merged articles that, when atomized, increase the number of articles with at least h citations the most. This leads to Algorithms 1 and 2 for Atomizing and Conservative Atomizing. Herein, the `Atomize()` operation takes a set S as input and returns $\{\{s\} \mid s \in S\}$. The algorithms yield the following theorem.

Theorem 1. Atomizing(μ) *and* Conservative Atomizing(μ) *are solvable in linear time for* $\mu \in$ {sumCite, unionCite}.

Proof. We first consider Atomizing(μ). Let $\mathcal{R}$ be a partition created from a partition $\mathcal{P}$ by atomizing a part $P^* \in \mathcal{P}$. Observe that for all $P \in \mathcal{P}$ and $R \in \mathcal{R}$ we have that $P = R$ implies $\mu(P) = \mu(R)$, for $\mu \in$ {sumCite, unionCite}. Intuitively, this means that atomizing a single part $P^* \in \mathcal{P}$ does not alter the μ-value of any other part of the partition.

Algorithm 1 computes a partition $\mathcal{R}$ that has a maximal number of parts R with $\mu(R) \geq h$ that can be created by applying atomizing operations to $\mathcal{P}$: It applies the atomizing operation to each part $P \in \mathcal{P}$ if there is at least one singleton A in the atomization of P with $\mu(A) \geq h$. By the above

Algorithm 2: Conservative Atomizing

```
Input: A citation graph $D = (V, A)$, a set $W \subseteq V$ of articles, a partition $\mathcal{P}$ of $W$, nonnegative
       integers $h$ and $k$, and a measure $\mu$.
Output: A partition $\mathcal{R}$ of $W$.
$\mathcal{R} \leftarrow \mathcal{P}$
foreach $P \in \mathcal{P}$ do
    $\ell_P \leftarrow 0$
    $\mathcal{A} \leftarrow$ Atomize($P$)
    $\ell_P \leftarrow \ell_P + |\{A \in \mathcal{A} \mid \mu(A) \geq h\}|$
    if $\mu(P) \geq h$ then $\ell_P \leftarrow \ell_P - 1$
for $i \leftarrow 1$ to $k$ do
    $P^* \leftarrow \arg\max_{P \in \mathcal{P}}\{\ell_P\}$
    if $\ell_{P^*} > 0$ then
        $\mathcal{A} \leftarrow$ Atomize($P^*$)
        $\mathcal{R} \leftarrow (\mathcal{R} \setminus \{P^*\}) \cup \mathcal{A}$
    $\ell_{P^*} \leftarrow -1$
return $\mathcal{R}$
```

observation, this cannot decrease the total number of parts in the partition that have a μ-value of at least h. Furthermore, we have that for all $R \in \mathcal{R}$, we cannot potentially increase the number of parts with μ-value at least h by atomizing R. Thus, we get the maximal number of parts R with $\mu(R) \geq h$ that can be created by applying atomizing operations to $\mathcal{P}$.

Obviously, if $\mathcal{R}$ has at least h parts R with $\mu(R) \geq h$, we face a yes-instance. Conversely, if the input is a yes-instance, then there is a number of atomizing operations that can be applied to $\mathcal{P}$ such that the resulting partition $\mathcal{R}$ has at least h parts R with $\mu(R) \geq h$ and the algorithm finds such a partition $\mathcal{R}$. Finally, it is easy to see that the algorithm runs in linear time.

The pseudocode for solving Conservative Atomizing(μ) is given in Algorithm 2. First, in Lines 2–6, for each part P, Algorithm 2 records how many singletons A with $\mu(A) \geq h$ are created when atomizing P. Then, in Lines 7–12, it repeatedly atomizes the part yielding the most such singletons. This procedure creates the maximum number of parts that have a μ-value of at least h, because the μ-value cannot be increased by exchanging one of these atomizing operations by another.

Obviously, if $\mathcal{R}$ has at least h parts R with $\mu(R) \geq h$, then we face a yes-instance. Conversely, if the input is a yes-instance, then there are k atomizing operations that can be applied to $\mathcal{P}$ to yield an h-index of at least h. Because Algorithm 2 takes successively those operations that yield the most new parts with h citations, the resulting partition $\mathcal{R}$ has at least h parts R with $\mu(R) \geq h$. It is not hard to verify that the algorithm has linear running time. □

3.2. Manipulation by Extracting

Recall that the extracting operation removes a single article from a merged article. All variants of the extraction problem are solvable in linear time. Intuitively, in the cautious case, it suffices to find k extracting operations that each increase the number of articles with h citations. In the conservative case, we determine for each merged article a set of extraction operations that increases the number of articles with h citations the most. Then we use the extraction operations for those k merged articles that yield the k largest increases in the number of articles with h citations. This leads to Algorithms 3–5 for Extracting, Cautious Extracting, and Conservative Extracting, respectively, which yield the following theorem.

Algorithm 3: Extracting

Input: A citation graph $D = (V, A)$, a set $W \subseteq V$ of articles, a partition $\mathcal{P}$ of W, a nonnegative integer h and a measure μ.

Output: A partition $\mathcal{R}$ of W.

```
$\mathcal{R} \leftarrow \emptyset$
foreach $P \in \mathcal{P}$ do
    foreach $v \in P$ do
        if $\mu(\{v\}) \geq h$ then
            $\mathcal{R} \leftarrow \mathcal{R} \cup \{\{v\}\}$
            $P \leftarrow P \setminus \{v\}$
    if $P \neq \emptyset$ then $\mathcal{R} \leftarrow \mathcal{R} \cup \{P\}$
return $\mathcal{R}$
```

Algorithm 4: Cautious Extracting

```
Input: A citation graph $D = (V, A)$, a set $W \subseteq V$ of articles, a partition $\mathcal{P}$ of $W$, nonnegative
       integers $h$ and $k$, and a measure $\mu$.
Output: A partition $\mathcal{R}$ of $W$.
$\mathcal{R} \leftarrow \emptyset$
foreach $P \in \mathcal{P}$ do
    foreach $v \in P$ do
        if $k > 0$ and $\mu(\{v\}) \geq h$ and $\mu(P \setminus \{v\}) \geq h$ then
            $\mathcal{R} \leftarrow \mathcal{R} \cup \{\{v\}\}$
            $P \leftarrow P \setminus \{v\}$
            $k \leftarrow k - 1$
    if $P \neq \emptyset$ then $\mathcal{R} \leftarrow \mathcal{R} \cup \{P\}$
return $\mathcal{R}$
```

Theorem 2. *Extracting(μ), Conservative Extracting(μ), and Cautious Extracting(μ) are solvable in linear time for $\mu \in$ {sumCite, unionCite}.*

Proof. We first consider Extracting(μ). Let $\mathcal{R}$ be a partition produced from $\mathcal{P}$ by extracting an article from a part $P^* \in \mathcal{P}$. Recall that this does not alter the μ-value of any other part (i.e., for all $P \in \mathcal{P}$ and $R \in \mathcal{R}$, we have that $P = R$ implies $\mu(P) = \mu(R)$ for $\mu \in$ {sumCite, unionCite}).

Algorithm 5: Conservative Extracting

```
Input: A citation graph $D = (V, A)$, a set $W \subseteq V$ of articles, a partition $\mathcal{P}$ of $W$, nonnegative
       integers $h$ and $k$, and a measure $\mu$.
Output: A partition $\mathcal{R}$ of $W$.
foreach $P \in \mathcal{P}$ do
    $\ell_P \leftarrow 0$
    $\mathcal{R}_P \leftarrow \emptyset$
    foreach $v \in P$ do
        if $\mu(\{v\}) \geq h$ and $\mu(P \setminus \{v\}) \geq h$ then
            $\mathcal{R}_P \leftarrow \mathcal{R}_P \cup \{\{v\}\}$
            $P \leftarrow P \setminus \{v\}$
            $\ell_P \leftarrow \ell_P + 1$
    if $P = \emptyset$ then $\mathcal{R}_P \leftarrow \mathcal{R}_P \cup \{P\}$
$\mathcal{P}^* \leftarrow$ the $k$ elements of $P \in \mathcal{P}$ with largest $\ell_P$-values
$\mathcal{R} \leftarrow \bigcup_{P \in \mathcal{P}^*} \mathcal{R}_P \cup (\mathcal{P} \setminus \mathcal{P}^*)$
return $\mathcal{R}$
```

Consider Algorithm 3. It is easy to see that the algorithm only performs extracting operations and that the running time is linear. So we have to argue that whenever there is a partition $\mathcal{R}$ that can be produced by extracting operations from $\mathcal{P}$ such that the h-index is at least h, then the algorithm finds a solution.

We show this by arguing that the algorithm produces the maximum number of articles with at least h citations possible. Extracting an article that has strictly less than h citations cannot produce an h-index of at least h unless we already have an h-index of at least h, because the number of articles with h or more citations does not increase. Extracting an article with h or more citations cannot decrease the number of articles with h or more citations. Hence, if there are no articles with at least h citations that we can extract, we cannot create more articles with h or more citations. Therefore, we have produced the maximum number of articles with h or more citations when the algorithm stops.

The pseudocode for solving Cautious Extracting(μ) is given in Algorithm 4. We perform up to k extracting operations (Line 6). Each of them increases the number of articles that have h or more citations by one. As Algorithm 4 checks each atomic article in each merged article, it finds k extraction operations that increase the number of articles with h or more citations if they exist. Thus, it produces the maximum possible number of articles that have h or more citations and that can be created by k extracting operations.

To achieve linear running time, we need to compute $\mu(P \setminus \{v\})$ efficiently in Line 4. This can be done by representing articles as integers and using an n-element array A that stores throughout the loop in Line 3, for each article $w \in N_D^{\text{in}}[P]$, the number $A[w]$ of articles in P that are cited by w. Using this array, one can compute $\mu(P \setminus \{v\})$ in $O(\deg^{\text{in}}(v))$ time in Line 4, amounting to overall linear time. The time needed to maintain array A is also linear: We initialize it once in the beginning with all zeros. Then, before entering the loop in Line 3, we can in $O(|N_D^{\text{in}}(P)|)$ total time store for each article $v \in N_D^{\text{in}}[P]$, the number $A[w]$ of articles in P that are cited by w. To update the array within the loop in Line 3, we need $O(\deg^{\text{in}}(v))$ time if Line 6 applies. In total, this is linear time.

Finally, the pseudocode for solving Conservative Extracting(μ) is given in Algorithm 5. For each merged article $P \in \mathcal{P}$, Algorithm 5 computes a set $\mathcal{R}_P$ and the number ℓ_P of additional articles v with $\mu(v) \geq h$ that can be created by extracting. Then it chooses a set $\mathcal{P}^*$ of k merged articles $P \in \mathcal{P}$ with maximum ℓ_P and, from each $P \in \mathcal{P}^*$, extracts the articles in $\mathcal{R}_P$.

This procedure creates the maximum number of articles that have a μ-value of at least h while only performing extraction operations on at most k merges.

Obviously, if the solution $\mathcal{R}$ has at least h parts R with $\mu(R) \geq h$, then we face a yes-instance. Conversely, if the input is a yes-instance, then there are k merged articles that we can apply extraction operations to, such that the resulting partition $\mathcal{R}$ has at least h parts R with $\mu(R) \geq h$. Because the algorithm produces the maximal number of parts R with $\mu(R) \geq h$, it achieves an h-index of at least h.

The linear running time follows by implementing the check in Line 5 in $O(\deg^{\text{in}}(v))$ time as described for Algorithm 4 and by using counting sort to find the k parts to extract from in Line 10. □

3.3. Manipulation by Dividing

Recall that the dividing operation splits a merged article into two arbitrary parts. First we consider the basic and conservative cases and show that they are FPT when parameterized by the h-index h. Then we show that the cautious variant is W[1]-hard when parameterized by k. Dividing(μ) is closely related to h-index Manipulation(μ) (van Bevern et al., 2016b; de Keijzer

& Apt, 2013) which is, given a citation graph $D = (V, A)$, a subset of articles $W \subseteq V$, and a nonnegative integer h, to decide whether there is a partition $\mathcal{P}$ of W such that $\mathcal{P}$ has h-index h with respect to μ. de Keijzer and Apt (2013) showed that h-index Manipulation(sumCite) is NP-hard, even if merges are unconstrained. The NP-hardness of h-index Manipulation for $\mu \in$ {unionCite, fusionCite} follows. We can reduce h-index Manipulation to Conservative Dividing by defining the partition $\mathcal{P} = \{W\}$; hence we get the following.

Proposition 1. Dividing and Conservative Dividing are NP-hard for $\mu \in$ {sumCite, unionCite, fusionCite}.

As to computational tractability, Dividing and Conservative Dividing are FPT when parameterized by h—the h-index to achieve.

Theorem 3. *Dividing and Conservative Dividing(μ) can be solved in $2^{O(h^4 \log h)} \cdot n^{O(1)}$ time, where h is the h-index to achieve and $\mu \in$* {sumCite, unionCite}.

Proof. The pseudocode is given in Algorithm 6. Herein, `Merge`(D, W, h, μ) decides h-index Manipulation(μ); that is, it returns `true` if there is a partition $\mathcal{Q}$ of W such that has h-index h and `false` otherwise. It follows from van Bevern et al. (2016b, Theorem 7) that `Merge` can be carried out in $2^{O(h^4 \log h)} \cdot n^{O(1)}$ time.

Algorithm 6 first finds, using `Merge`, the maximum number ℓ_P of (merged) articles with at least h citations that we can create in each part $P \in \mathcal{P}$. For this, we first prepare an instance (D', W', h, μ) of h-index Manipulation(μ) in Lines 2 and 3. In the resulting instance, we ask whether there is a partition of P with h-index h. If this is the case, then we set ℓ_P to h. Otherwise, we add one artificial article with h citations to W' in Line 9. Intuitively, this causes `Merge` to check whether there is a partition of P into $h - 1$ (more generally, one less than in the current iteration) merged articles with h citations each in the next iteration. We iterate this process until `Merge` returns `true`, or we find that there is not even one merged article contained in P with h citations. Clearly, this process

Algorithm 6: Conservative Dividing

```
Input: A citation graph $D = (V, A)$, a set $W \subseteq V$ of articles, a partition $\mathcal{P}$ of $W$, nonnegative
       integers $h$ and $k$, and a measure $\mu$.
Output: true if $k$ dividing operations can be applied to $\mathcal{P}$ to yield $h$-index $h$ and false
        otherwise.
1  foreach $P \in \mathcal{P}$ do
2      $D' \leftarrow$ The graph obtained from $D$ by removing all citations $(u, v)$ such that
       $v \notin P$ and adding $h + 1$ articles $r_1, \ldots, r_{h+1}$
3      $W' \leftarrow P$, $\ell_P \leftarrow 0$
4      for $i \leftarrow 0$ to $h$ do
5          if Merge$(D', W', h, \mu)$ then
6              $\ell_P \leftarrow h - i$
7              Break
8          else
9              Add $r_i$ to $W'$ and add each citation $(r_j, r_i)$, $j \in \{1, \ldots, h + 1\} \setminus \{i\}$ to $D'$
10 return $\exists \mathcal{P}' \subseteq \mathcal{P}$ s.t. $|\mathcal{P}'| \leq k$ and $\Sigma_{P \in \mathcal{P}'} \ell_P \geq h$
```

correctly computes ℓ_P. Thus, the algorithm is correct. The running time is clearly dominated by the calls to `Merge`. As `Merge` runs in $2^{O(h^4 \log h)} \cdot n^{O(1)}$ time (van Bevern et al., 2016b, Theorem 7), the running time bound follows. □

We note that `Merge` can be modified so that it outputs the desired partition. Hence, we can modify Algorithm 6 to output the actual solution. Furthermore, for $k = n$, Algorithm 6 solves the nonconservative variant, which is therefore also fixed-parameter tractable parameterized by h.

In contrast, for the cautious variant we show W[1]-hardness when parameterized by k, the number of allowed operations.

Theorem 4. Cautious Dividing(μ) *is NP-hard and W[1]-hard when parameterized by k for* $\mu \in$ {sumCite, unionCite, fusionCite}*, even if the citation graph is acyclic.*

Proof. We reduce from the Unary Bin Packing problem: given a set S of n items with integer sizes s_i, $i \in \{1, \dots, n\}$, ℓ bins and a maximum bin capacity B, can we distribute all items into the ℓ bins? Herein, all sizes are encoded in unary. Unary Bin Packing parameterized by ℓ is W[1]-hard (Jansen, Kratsch, et al., 2013).

Given an instance (S, ℓ, B) of Unary Bin Packing, we produce an instance $(D, W, \mathcal{P}, h, \ell - 1)$ of Cautious Dividing(sumCite). Let $s^* = \Sigma_i s_i$ be the sum of all item sizes. We assume that $B < s^*$ and $\ell \cdot B \geq s^*$ as, otherwise, the problem is trivial, because all items fit into one bin or they collectively cannot fit into all bins, respectively. Furthermore, we assume that $\ell < B$ because, otherwise, the instance size is upper bounded by a function of ℓ and, hence, is trivially FPT with respect to ℓ. We construct the instance of Cautious Dividing(sumCite) in polynomial time as follows.

- Add s^* articles $x_1, \dots, x_{s^*}$ to D. These are only used to increase the citation count of other articles.
- Add one article a_i to D and W for each s_i.
- For each article a_i, add citations (x_j, a_i) for all $1 \leq j \leq s_i$ to G. Note that, after adding these citations, each article a_i has citation count s_i.
- Add $\Delta := \ell \cdot B - s^*$ articles $u_1, \dots, u_\Delta$ to D and W.
- For each article u_i with $i \in \{1, \dots, \Delta\}$, add a citation (x_1, u_i) to D. Note that each article u_i has citation count 1.
- Add $B - \ell$ articles $h_1, \dots, h_{B-\ell}$ to D and W.
- For each article h_i with $i \in \{1, \dots, B - \ell\}$, add citations (x_j, h_i) for all $1 \leq j \leq B$ to D. Note that each article h_i has citation count B.
- Add $P^* = \{a_1, \dots, a_n, u_1, \dots, u_\Delta\}$ to $\mathcal{P}$, for each article h_i with $i \in \{1, \dots, B - \ell\}$, add $\{h_i\}$ to $\mathcal{P}$, and set $h = B$.

Now we show that (S, ℓ, B) is a yes-instance if and only if $(D, W, \mathcal{P}, h, \ell - 1)$ is a yes-instance.

($\Rightarrow$) Assume that (S, ℓ, B) is a yes-instance and let $S_1, \dots, S_\ell$ be a partition of S such that items in S_i are placed in bin i. Now we split P^* into ℓ parts $R_1, \dots, R_\ell$ in the following way. Note that for each S_i, we have that $\Sigma_{s_j \in S_i} s_j = B - \delta_i$ for some $\delta_i \geq 0$. Furthermore, $\Sigma_i \delta_i = \Delta$. Recall that there are Δ articles $u_1, \dots, u_\Delta$ in P^*. Let $\delta_{<i} = \Sigma_{j<i} \delta_j$ and $U_i = \{u_{\delta_{<i}}+1, \dots, u_{\delta_{<i}}+\delta_i\}$, with $\delta_0 = 0$ and if $\delta_i > 0$, let $U_i = \emptyset$ for $\delta_i = 0$. We set $R_i = \{a_j \mid s_j \in S_i\} \cup U_i$. Then for each R_i, we have that sumCite(R_i) = sumCite($\{a_j \mid s_j \in S_i\}$) + sumCite(U_i), which simplifies to sumCite(R_i) = $\Sigma_{s_j \in S_i} s_j + \delta_i = B$. For each i, $1 \leq i \leq n$, we have sumCite($\{h_i\}$) = B. Hence, $\mathcal{R} = \{R_1, \dots, R_\ell, \{h_1\}, \dots, \{h_{B-\ell}\}\}$ has h-index B.

(⇐) Assume that $(D, W, \mathcal{P}, h, \ell - 1)$ is a yes-instance and let $\mathcal{R}$ be a partition with h-index h. Recall that $\mathcal{P}$ consists of P^* and $B - \ell$ singletons $\{h_1\}, \ldots, \{h_{B-\ell}\}$, which are hence also contained in $\mathcal{R}$. Furthermore, $\text{sumCite}(\{h_i\}) = B$ for each h_i and, by the definition of the h-index, there are ℓ parts $R_1, \ldots, R_\ell$ with $R_i \subset P^*$ and $\text{sumCite}(R_i) \geq B$ for each i. Because, by definition, $\text{sumCite}(P^*) = \ell \cdot B$ and $\text{sumCite}(P^*) = \Sigma_{1 \leq i \leq \ell}\, \text{sumCite}(R_i)$ we have that $\text{sumCite}(R_i) = B$ for all i. It follows that $\text{sumCite}(R_i \setminus \{u_1, \ldots, u_\Delta\}) \leq B$ for all i. This implies that packing into bin i each item in $\{s_j \mid a_j \in R_i\}$ solves the instance (S, ℓ, B). □

Note that this proof can be modified to cover also the unionCite and fusionCite cases by adding $\ell \cdot s^*$ extra x-articles and ensuring that no two articles in W are cited by the same x-article.

4. FUSIONCITE

We now consider the fusionCite measure, which makes manipulation considerably harder than the other measures. In particular, we obtain that, even in the most basic case, the manipulation problem is NP-hard.

Theorem 5. Atomizing(fusionCite) *and* Extracting(fusionCite) *are NP-hard, even if the citation graph is acyclic and* $s = 3$, *where* s *is the largest number of articles merged into one.*

Proof. We reduce from the NP-hard 3-Sat problem: Given a 3-CNF formula F with n variables and m clauses, decide whether F has a satisfying truth assignment to its variables. Without loss of generality, we assume $n + m > 3$ and that each clause contains three literals over mutually distinct variables. Given a formula F with variables $x_1, \ldots, x_n$ and clauses $c_1, \ldots, c_m$ such that $n + m > 3$, we produce an instance $(D, W, \mathcal{P}, m + n)$ of Atomizing(fusionCite) or Extracting(fusionCite) in polynomial time as follows. The construction is illustrated in Figure 3.

For each variable x_i of F, add to D and W sets $\mathcal{X}_i^F := \{X_{i,1}^F, X_{i,2}^F, X_{i,3}^F\}$ and $\mathcal{X}_i^T := \{X_{i,1}^T, X_{i,2}^T, X_{i,3}^T\}$ of *variable articles*. Add $\mathcal{X}_i^F$ and $\mathcal{X}_i^T$ to $\mathcal{P}$. Let $h := m + n$. For each variable x_i, add

1. $h - 2$ citations from (newly introduced) distinct atomic articles to $X_{i,1}^T$ and $X_{i,1}^F$,
2. citations from $X_{i,1}^F$ to $X_{i,2}^T$ and from $X_{i,2}^T$ to $X_{i,3}^F$ and
3. citations from $X_{i,1}^T$ to $X_{i,2}^F$ and from $X_{i,2}^F$ to $X_{i,3}^T$.

Next, for each clause c_j of F, add a *clause article* C_j with $h - 4$ incoming citations to D, to W, and add $\{C_j\}$ to $\mathcal{P}$. Finally, if a positive literal x_i occurs in a clause c_j, then add citations $(X_{i,\ell}^T, C_j)$ to D for $\ell \in \{2, 3\}$. If a negative literal $\neg x_i$ occurs in a clause c_j, then add citations $(X_{i,\ell}^F, C_j)$ to D

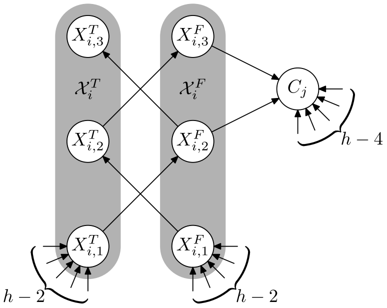

Figure 3. Illustration of the construction in the proof of Theorem 5 for a literal $\neg x_i$ contained in a clause c_j.

for $\ell \in \{2, 3\}$. This concludes the construction. Observe that D is acyclic, as all citations go from variable articles to clause articles or to variable articles with a higher index. It remains to show that F is satisfiable if and only if $(D, W, \mathcal{P}, h)$ is a yes-instance.

($\Rightarrow$) If F is satisfiable, then a solution $\mathcal{R}$ for $(D, W, \mathcal{P}, h)$ looks as follows: for each $i \in \{1, \ldots, n\}$, if x_i is true, then we put $X_i^F \in \mathcal{R}$ and we put $\mathcal{X}_i^T \in \mathcal{R}$ otherwise. All other articles of D are added to $\mathcal{R}$ as singletons. We count the citations that every part of $\mathcal{R}$ gets from other parts of $\mathcal{R}$. If x_i is true, then $\mathcal{X}_i^F$ gets two citations from $\{X_{i,\ell}^T\}$ for $\ell \in \{1, 2\}$ and the $h - 2$ initially added citations. Moreover, for the clause c_j containing the literal x_i, $\{C_j\}$ gets two citations from $\{X_{i,\ell}^T\}$ for $\ell \in \{2, 3\}$, at least two citations from variable articles for two other literals it contains, and the $h - 4$ initially added citations. Symmetrically, if x_i is false, then $\{\mathcal{X}_i^T\}$ gets h citations and so does every $\{C_j\}$ for each clause c_j containing the literal $\neg x_i$. As every clause is satisfied and every variable is either true or false, it follows that each of the m clause articles gets h citations and that, for each of the n variables x_i, either $\mathcal{X}_i^F$ or $\mathcal{X}_i^T$ gets h citations. It follows that $h = m + n$ parts of $\mathcal{R}$ get at least h citations and thus, that $\mathcal{R}$ has h-index at least h.

($\Leftarrow$) Let $\mathcal{R}$ be a solution for $(D, W, \mathcal{P}, m + n)$. We first show that, for each variable x_i, we have either $\mathcal{X}_i^T \in \mathcal{R}$ or $\mathcal{X}_i^F \in \mathcal{R}$. To this end, it is important to note two facts:

1. For each variable x_i, $\mathcal{X}_i^T$ contains two atomic articles with one incoming arc in D and one with $h - 2$ incoming arcs. Thus, no subset of $\mathcal{X}_i^T$ can get h citations. The same holds for $\mathcal{X}_i^F$.
2. If, for some variable x_i, the part $\mathcal{X}_i^T \in \mathcal{R}$ gets h citations, then $\mathcal{X}_i^F \notin \mathcal{R}$ and vice versa.

Thus, as there are at most m clause articles and $\mathcal{R}$ contains $h = m + n$ parts with h citations, $\mathcal{R}$ contains exactly one of the parts $\mathcal{X}_i^T$, $\mathcal{X}_i^F$ of each variable x_i. It follows that, in $\mathcal{R}$, all singleton clause articles have to receive h citations. Each such article gets at most $h - 4$ initially added citations and citations from at most three sets $\mathcal{X}_i^T$ or $\mathcal{X}_i^F$ for some variable x_i. Thus, for each clause c_j, there is a literal x_i in c_j or a literal $\neg x_i$ in c_j such that $\mathcal{X}_i^T \notin \mathcal{R}$ or $\mathcal{X}_i^F \notin \mathcal{R}$, respectively. It follows that setting each x_i to true if and only if $\mathcal{X}_i^T \notin \mathcal{R}$ gives a satisfying truth assignment to the variables of F. □

This NP-hardness result motivates the search for fixed-parameter tractability.

Theorem 6. Atomizing(fusionCite) *can be solved in* $O(4^{h^2}(n + m))$ *time, where h is the h-index to achieve.*

Proof. We use the following procedure to solve an instance $(D, W, \mathcal{P}, h)$ of Atomizing (fusionCite).

Let $\mathcal{P}_{\geq h}$ be the set of merged articles $P \in \mathcal{P}$ with fusionCite $(P) \geq h$. If $|P_{\geq h}| \geq h$, then we face a yes-instance and output "yes." We can determine whether this is the case in linear time because we can compute fusionCite (P) in linear time for all $P \in \mathcal{P}$. Below we assume that $|\mathcal{P}_{\geq h}| < h$.

First, we atomize all $P \in \mathcal{P}$ that cannot have h or more citations; that is, for which, even if we atomize all merged articles except for P, we have fusionCite$(P) < h$. Formally, we atomize P if $\Sigma_{v \in P} |N_{D-P}^{\text{in}}(v)| < h$. Let $\mathcal{P}'$ be the partition obtained from $\mathcal{P}$ after these atomizing operations; note that $\mathcal{P}'$ can be computed in linear time.

The basic idea is now to look at all remaining merged articles that receive at least h citations from atomic articles; they form the set $\mathcal{P}_{<h}$ below. They are cited by at most $h - 1$ other merged articles. Hence, if the size of $\mathcal{P}_{<h}$ exceeds some function $f(h)$, then, among the contained merged articles, we find a large number of merged articles that do not cite each other. If we have such a set,

then we can atomize all other articles, obtaining h-index h. If the size of $\mathcal{P}_{<h}$ is smaller than $f(h)$, then we can determine by brute force whether there is a solution.

Consider all merged articles $P \in \mathcal{P}'$ that have fewer than h citations but can obtain h or more citations by applying atomizing operations to merged articles in $\mathcal{P}'$. Let us call the set of these merged articles $\mathcal{P}_{<h}$. Formally, $P \in \mathcal{P}_{<h}$ if $\Sigma_{v \in P} |N^{\text{in}}_{D-P}(v)| \geq h$ and fusionCite$(P) < h$. Again, $\mathcal{P}_{<h}$ can be computed in linear time. Note that $\mathcal{P}' \setminus (\mathcal{P}_{\geq h} \cup \mathcal{P}_{<h})$ consists only of singletons.

Now, we observe the following. If there is a set $\mathcal{P}^* \subseteq \mathcal{P}_{<h}$ of at least h merged articles such that, for all $P_i, P_j \in \mathcal{P}^*$, neither P_i cites P_j nor P_j cites P_i, then we can atomize all merged articles in $\mathcal{P}' \setminus \mathcal{P}^*$ to reach an h-index of at least h. We finish the proof by showing that we can conclude the existence of the set $\mathcal{P}^*$ if $\mathcal{P}_{<h}$ is sufficiently large and solve the problem using brute force otherwise.

Consider the undirected graph G that has a vertex v_P for each $P \in \mathcal{P}_{<h}$ and an edge between v_{Pi} and v_{Pj} if P_i cites P_j or P_j cites P_i. Note that $\{v_P \mid P \in \mathcal{P}^*\}$ forms an independent set in G. Furthermore, let I be an independent set in G that has size at least h. Let $\mathcal{P}^{**} = \{P \in \mathcal{P}_{<h} \mid v_P \in I\}$. Then, we can atomize all merged articles in $\mathcal{P}' \setminus \mathcal{P}^{**}$ to reach an h-index of at least h.

We claim that the number of edges in G is at most $(h - 1) \cdot |\mathcal{P}_{<h}|$. This is because the edge set of G can be obtained by enumerating for every vertex v_P the edges incident with v_P that result from a citation of P from another $P' \in \mathcal{P}_{<h}$. The citations for each P are less than h as, otherwise, we would have $P \in \mathcal{P}_{\geq h}$. Now, we can make use of Turán's Theorem, which can be stated as follows: If a graph with ℓ vertices has at most $\ell k/2$ edges, then it admits an independent set of size at least $\ell/(k + 1)$ (Jukna, 2001, Exercise 4.8). Hence, if $|\mathcal{P}_{<h}| \geq 2h^2 - h$, then we face a yes-instance because G contains an independent set of size at least h. Consequently, we can find a solution by taking an arbitrary subset $\mathcal{P}'_{<h}$ of $\mathcal{P}_{<h}$ with $|\mathcal{P}'_{<h}| = 2h^2 - h$, atomizing every merged article outside of $\mathcal{P}'_{<h}$, and guessing which merged articles we need to atomize inside of $\mathcal{P}'_{<h}$. If $|\mathcal{P}_{<h}| < 2h^2 - h$, then we guess which merged articles in $\mathcal{P}_{<h} \cup \mathcal{P}_{\geq h}$ we need to atomize to obtain a solution if it exists. In both cases, for each guess we need linear time to determine whether we have found a solution, giving the overall running time of $O(4^{h^2} \cdot (m + n))$. □

For the conservative variant, however, we cannot achieve FPT, even if we add the number of atomization operations and the maximum size of a merged article to the parameter.

Theorem 7. Conservative Atomizing(fusionCite) *is NP-hard and W[1]-hard when parameterized by* $h + k + s$, *where* $s := \max_{P \in \mathcal{P}} |P|$, *even if the citation graph is acyclic*.

Proof. We reduce from the Clique problem: Given a graph G and an integer k, decide whether G contains a clique on at least k vertices. Clique parameterized by k is known to be W[1]-hard.

Given an instance (G, k) of Clique, we produce an instance $(D, W, \mathcal{P}, h, k)$ of Conservative Atomizing (fusionCite) in polynomial time as follows. Without loss of generality, we assume $k \geq 4$ so that $\binom{k}{2} \geq 4$. For each vertex v of G, introduce a set R_v of $\left\lceil \binom{k}{2}/2 \right\rceil$ vertices to D and W and add R_v as a part to $\mathcal{P}$. For an edge $\{v, w\}$ of G, add to D and W a vertex $e_{\{v,w\}}$ and add $\{e_{\{v,w\}}\}$ to $\mathcal{P}$. Moreover, add a citation from each vertex in $R_v \cup R_w$ to $e_{\{v,w\}}$. Finally, set $h := \binom{k}{2}$. Each of h, k, and s in our constructed instance of Conservative Atomizing(fusionCite) depends only on k in the input Clique instance. It remains to show that (G, k) is a yes-instance for Clique if and only if $(D, W, \mathcal{P}, h, k)$ is.

($\Rightarrow$) Assume that (G, k) is a yes-instance and let S be a clique in G. Then, atomizing R_v for each $v \in S$ yields $\binom{k}{2}$ articles with at least $\binom{k}{2}$ citations in D: For each of the $\binom{k}{2}$ pairs of vertices $v, w \in S$, the vertex $e_{\{v,w\}}$ gets $\left\lceil \binom{k}{2}/2 \right\rceil$ citations from the vertices in R_v and the same number of citations from the vertices in R_w and, thus, at least $\binom{k}{2}$ citations in total.

($\Leftarrow$) Assume that $(D, W, \mathcal{P}, h, k)$ is a yes-instance and let $\mathcal{R}$ be a solution. We construct a subgraph $S = (V_S, E_S)$ of G that is a clique of size k. Let $V_S := \{v \in V(G) \mid R_v \in \mathcal{P} \setminus \mathcal{R}\}$ and $E_S := \{\{v, w\} \in E(G) \mid \{v, w\} \subseteq V_S\}$; that is, $S = G[V_S]$. Obviously, $|V_S| \leq k$. It remains to show $|E_S| \geq \binom{k}{2}$, which implies both that $|V_S| = k$ and that S is a clique. To this end, observe that the only vertices with incoming citations in D are the vertices $e_{\{v,w\}}$ for the edges $\{v, w\}$ of G. The only citations of a vertex $e_{\{v,w\}}$ are from the parts R_v and R_w in $\mathcal{P}$. That is, with respect to the partition $\mathcal{P}$, each vertex $e_{\{v,w\}}$ has two citations. As the h-index h to reach is $\binom{k}{2}$, at least $\binom{k}{2}$ vertices $e_{\{v,w\}}$ have to receive $\binom{k}{2} \geq 4$ citations, which is only possible by atomizing both R_v and R_w. That is, for at least $\binom{k}{2}$ vertices $e_{\{v,w\}}$, we have $\{R_v, R_w\} \subseteq \mathcal{P} \setminus \mathcal{R}$ and, thus, $v, w \subseteq V_S$ and $\{v, w\} \in E_S$. It follows that $|E_S| \geq \binom{k}{2}$. □

The reduction given above easily yields the same hardness result for most other problem variants: A vertex $e_{\{v,w\}}$ receives a sufficient number of citations only if R_v and R_w are atomized. Hence, even if we allow extractions or divisions on R_v, it helps only if we extract or split off all articles in R_v. The only difference is that the number of allowed operations is set to $k \cdot \left\lceil \binom{k}{2}/2 - 1 \right\rceil$ for these two problem variants. By the same argument, we obtain hardness for the conservative variants.

Corollary 1. For μ = fusionCite, Conservative Extracting(μ), Cautious Extracting(μ), Conservative Dividing(μ), and Cautious Dividing(μ) are NP-hard and W[1]-hard when parameterized by $h + k + s$, where $s := \max_{P \in \mathcal{P}} |P|$, even if the citation graph is acyclic.

5. COMPUTATIONAL EXPERIMENTS

To assess how much the h-index of a researcher can be manipulated by splitting articles in practice, we performed computational experiments with data extracted from Google Scholar.

5.1. Description of the Data

We use three data sets collected by van Bevern et al. (2016b). One data set consists of 22 selected authors of the conference IJCAI'13. The selection of these authors was biased to obtain profiles of authors in their early career. More precisely, the selected authors have a Google Scholar profile, an h-index between 8 and 20, between 100 and 1,000 citations, and activity between 5 and 10 years when the data was collected. Below we refer to this data set as *ijcai-2013*. The other two data sets contain Google Scholar data of "AI's 10 to Watch," a list of young accomplished researchers in AI compiled by *IEEE Intelligent Systems*. One data set contains five profiles from the 2011 edition (ai10-2011), the other eight profiles from the 2013 edition of the list (ai10-2013). In comparison with van Bevern et al. (2016b) we removed one author from the ai10-2013 data set because the data were inconsistent. All data were gathered between November 2014 and January 2015. For an overview of the data see Table 2.

Due to difficulties in obtaining the data from Google Scholar, van Bevern et al. (2016b) did not gather the concrete set of citations for articles that are cited a large number of times. These were articles that will always be part of the articles counted in the h-index. They subsequently ignored these articles as it is never beneficial to merge them with other articles to increase the h-index. In our case, although such articles may be merged initially, they will also always be counted in the h-index and hence their concrete set of citations is not relevant for us as well. The information about whether such articles could be merged is indeed contained in the data sets.

Table 2. Properties of the three data sets. Here, p is the number of profiles for each data set, $\overline{|W|}$ is the average number of atomic articles in the profile, $\bar{c}$ is the average number of citations, $\bar{h}$ is the average h-index in the data set (without merges), and h/a is the average h-index increase per year; the 'max' subscript denotes the maximum of these values.

	p	$\overline{\lvert W\rvert}$	$\lvert W\rvert_{\max}$	$\bar{c}$	$c_{\max}$	$\bar{h}$	$h_{\max}$	h/a
ai10-2011	5	170.2	234	1614.2	3725	34.8	46	2.53
ai10-2013	7	58.7	144	557.5	1646	14.7	26	1.57
ijcai-2013	22	45.9	98	251.5	547	10.4	16	1.24

5.2. Generation of Profiles with Merged Articles

In our setting, the input consists of a profile that already contains some merged articles. The merges should be performed in a way which reflects the purpose of merging in the Google Scholar interface. That is, the merged articles should roughly correspond to different versions of the same work. To find different versions of the same work, we used the compatibility graphs for each profile provided by van Bevern et al. (2016b) which they generated as follows. The set of vertices is the set of articles in the profile. For each article u let $T(u)$ denote the set of words in its title. There is an edge between articles u and v if $|T(u) \cup T(v)| \geq t \cdot |T(u) \cup T(v)|$, where $t \in [0, 1]$ is the *compatibility threshold*. For $t = 0$, the compatibility graph is a clique; for $t = 1$ only articles with the same words in the title are adjacent. For $t \leq 0.3$, very dissimilar articles are still considered compatible (van Bevern et al., 2016b). Hence, we usually focus on $t \geq 0.4$ below.

We then generated the merged articles as follows. We used four different methods so that we can avoid artifacts that could be introduced by one specific method. Each method iteratively computes an inclusion-wise maximal clique C in the compatibility graph D, adds it as a merged article to the profile, and then removes C from D. The clique C herein is computed as follows.

GreedyMax	Recursively include into C a largest-degree vertex that is adjacent to all vertices already included until no such vertex exists anymore.
GreedyMin	Recursively include into C a smallest-degree vertex that is adjacent to all vertices already included until no such vertex exists anymore.
Maximum	A maximum-size clique.
Ramsey	A recursive search of a maximal clique in the neighborhood of a vertex v and the remaining graph. See algorithm Clique Removal by Boppana and Halldórsson (1992) for details.

If the compatibility graph has no edge any more, then each method adds all remaining articles as atomic articles of the profile.

Figure 4 shows the distributions of the h-indices of the generated profiles with merged articles and those where no article has been merged. The lower edge of a box is the first quartile, the upper edge the third quartile, and the thick bar is the median; the remaining data points are shown by dots. Note that when no article is merged—and no atomic article cites itself—all three citation measures coincide. Often, merging compatible articles leads to a decline in h-index in our data sets and this effect is most pronounced for the more senior authors (in ai10-2011). In contrast, merging very closely related articles (compatibility threshold $t = 0.9$) for authors in ai10-2013 led to increased h-indices. The initial h-indices are very weakly affected by the different methods for generating initially merged articles.

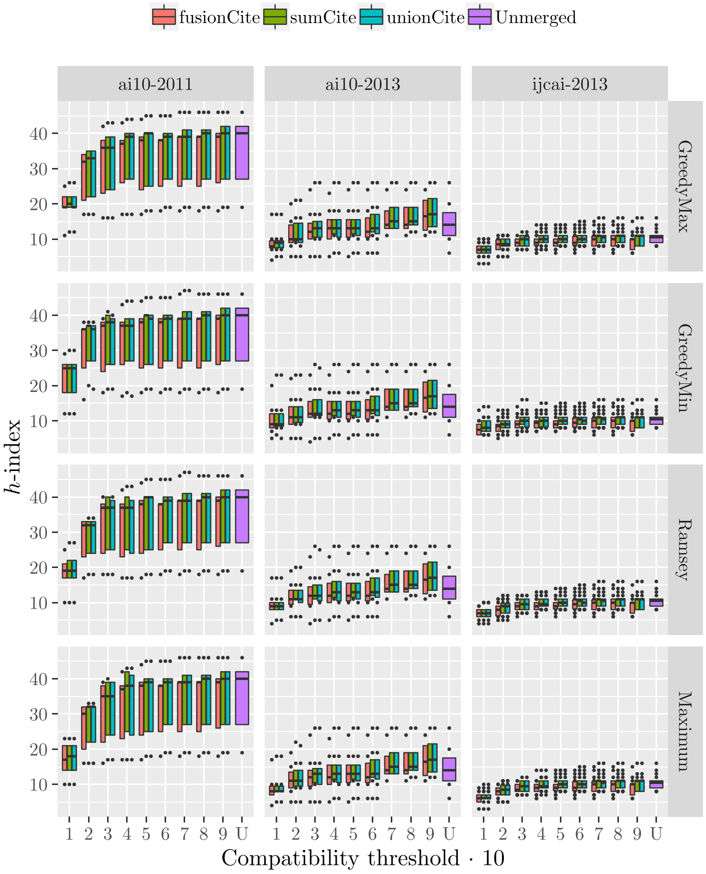

Figure 4. *h*-index distributions of the profiles with generated merged articles in comparison to the profiles without any merged articles.

5.3. Implementation

We implemented Algorithms 2, 4 and 5—the exact, linear-time algorithms from Section 3 for Conservative Atomizing, Conservative Extracting, and Cautious Extracting, respectively, each for all three citation measures, sumCite, unionCite, and fusionCite. The algorithms for sumCite and unionCite were implemented directly as described. For fusionCite, we implemented minor modifications of the described algorithms—to make the computation of μ well defined, we need to additionally specify which articles are currently merged, but otherwise the basic algorithms are unchanged. More precisely, recall that for Algorithm 2 we greedily perform atomizing operations if they increase the *h*-index. Thus, in the adaption to fusionCite, the partition $\mathcal{P}$ is continuously updated whenever the check in Line 6 is positive, and the application of μ = fusionCite in that line uses the partition $\mathcal{P}$ which is current at the time of application. Similarly, the partitions are updated after positive checks in Algorithm 4, Line 4, and in Algorithm 5, Line 5, and the application of μ = fusionCite in that line uses the current partition $\mathcal{P}$.

Using the algorithms, we computed *h*-index increases under the respective restrictions. For sumCite and unionCite these algorithms yield the maximum-possible *h*-index increases by Theorems 1 and 2. For fusionCite, we obtain only a lower bound.

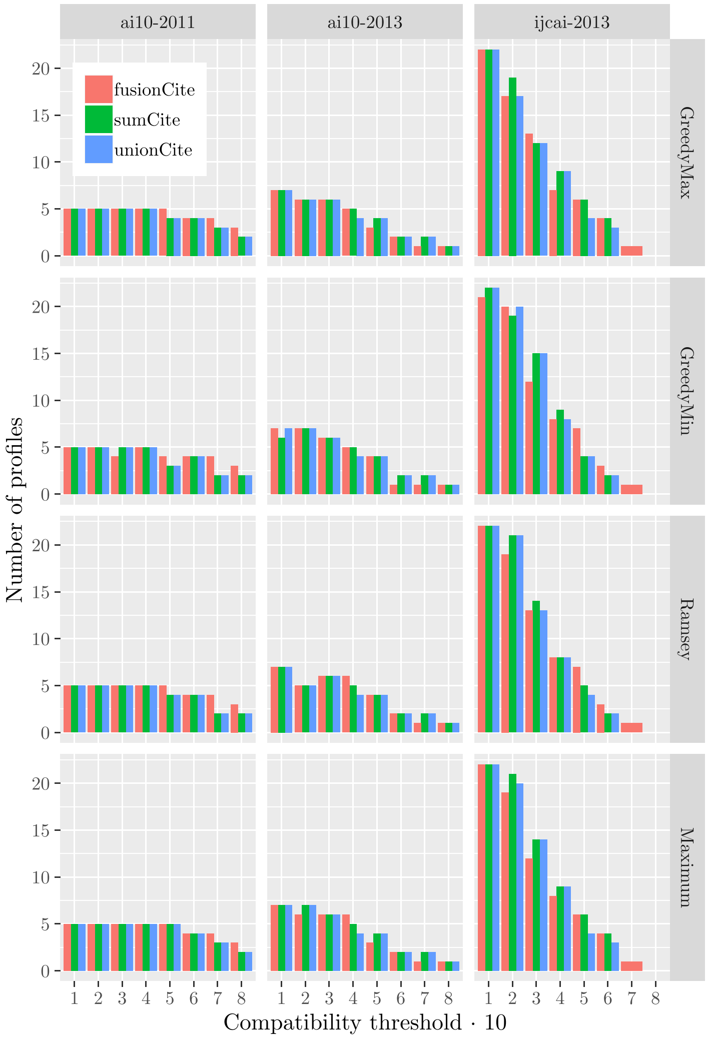

Figure 5. Number of profiles whose *h*-indices may be increased by unmerging.

The implementation is in Python 3.6.7 under Ubuntu Linux 18.04 and the source code is freely available[4]. In total, 137,626 instances of the decision problems were generated. Using a 2.5 GHz Intel Core i5-7200U CPU and 8 GB RAM, the instances could be solved within 14 hours altogether (ca. 350 ms average time per instance).

5.4. Authors with Potential for Manipulation

Figure 5 gives the number of profiles in which the *h*-index can be increased by unmerging articles. We say the profiles or the corresponding authors have *potential*. Concordant with intuition, for each threshold value, the methods for creating initial merges are roughly ordered according to the number of authors with potential as follows: Maximum > Ramsey > GreedyMax >

[4] See http://gitlab.com/rvb/split-index.

GreedyMin. GreedyMax and GreedyMin are surprisingly close. However, the differences between the methods in general are rather small, indicating that the property of having potential is inherent to the profile rather than the method for generating initial merges. As GreedyMax is one of the most straightforward of the four, we will focus only on GreedyMax below.

At first glance, we could expect that the number of authors with potential would decrease monotonically with increasing compatibility threshold: Note that, for increasing compatibility threshold the edge sets in compatibility graphs are decreasing in the subset order. Hence each maximal clique in the compatibility graph can only increase in size. However, because we employ heuristics to find the set of initial merges (in the case of Ramsey, GreedyMax, and GreedyMin) and because there may be multiple choices for a maximum-size clique (for Maximum), different possible partitionings into initial merges may result. This can lead to the fact that the authors with potential do not decrease monotonically with increasing compatibility threshold.

Furthermore, with the same initial merges it can happen that an increase in the *h*-index value through unmerging with respect to sumCite is possible and no increase is possible with respect to unionCite and vice versa. The first may happen, for example, if two articles *v*, *w* are merged such that sumCite({*v*, *w*}) is above but unionCite({*v*, *w*}) is below the *h*-index threshold. The second may happen if the *h*-index of the merged profile is lower for unionCite compared to that for sumCite. Then, unmerging articles may yield atomic articles that are still above the *h*-index threshold for unionCite but not for sumCite. As can be seen from Figure 5, both options occur in our data set.

The fraction of authors with potential differs clearly between the three data sets. The authors in ai10-2011 have already accumulated so many citations that almost all have potential for each threshold up to 0.6. Meanwhile, the authors with potential in ai10-2013 continually drop for increasing threshold and this drop is even more pronounced for ijcai-2013. This may reflect the three levels of seniority represented by the data sets.

There is no clear difference between the achievable *h*-indices when comparing fusionCite with unionCite and sumCite: While there are generally more authors with potential for each threshold for fusionCite in the ai10-2011 data set, there are fewer authors with potential for the ai10-2013 data set, and a similar number of authors with potential for the ijcai-2013 data set.

Focusing on the most relevant threshold, 0.4, and the unionCite measure, which is used by Google Scholar (van Bevern et al., 2016a), we see that all authors (100%) in ai10-2011 could potentially increase their *h*-indices by unmerging, four authors (57%) could do so in ai10-2013, and seven (31%) in ijcai-2013. We next focus only on these authors with potential and gauge to that extent manipulation is possible.

5.5. Extent and Cost of Possible Manipulation

Figure 6 shows the largest achievable *h*-index increases for the authors with potential in the three data sets: Again, the lower edge of a box is the first quartile, the upper edge the third quartile, and the thick bar is the median; the remaining data points are shown by dots.

In the majority of cases, drastic increases can only be achieved when the compatibility threshold is lower than 0.4. Generally, the increases achieved for the fusionCite measure are slightly lower than for the other two, but the median is at most smaller by one. Because of the heuristic nature of our algorithms for fusionCite, we cannot exclude the possibility that the largest possible increases for fusionCite are comparable to the other two measures. In the most relevant regime of unionCite and compatibility threshold $t = 0.4$, the median *h*-index increases are 4 for the ai10-2011 authors, 1 for the ai10-2013 authors, and 2 for the ijcai-2013 authors. Notably, there is an outlier in ijcai-2013 who can achieve an increase of 5.

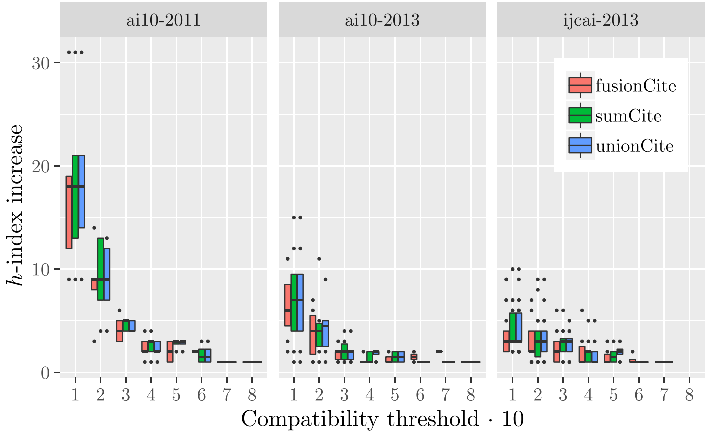

Figure 6. *h*-index increases for each compatibility threshold for authors with potential (note that these authors may be different for different threshold values). The increases are largest-possible for sumCite and unionCite.

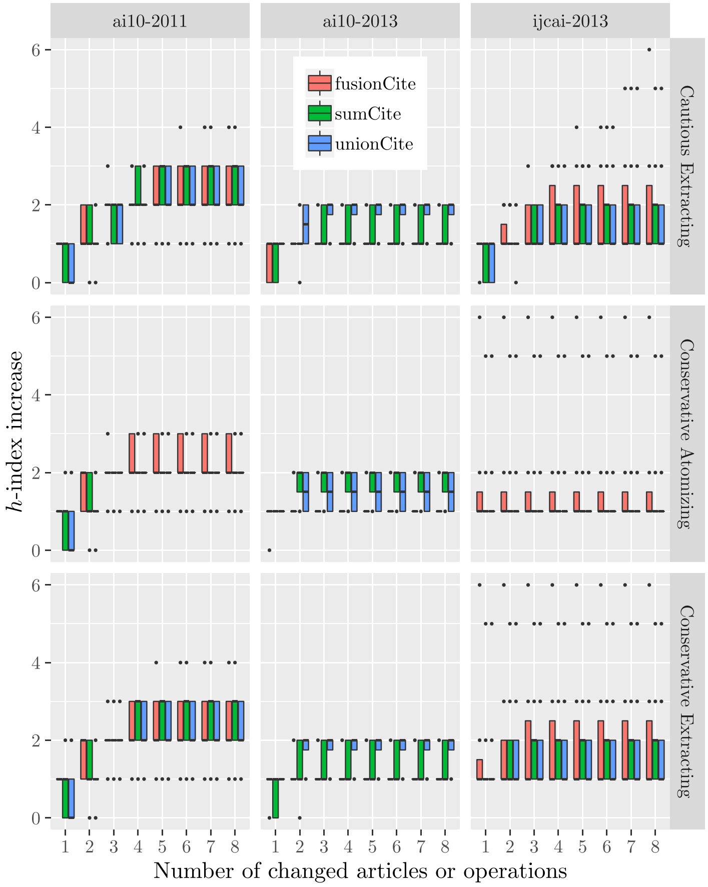

Figure 7. *h*-index increases versus number of changed articles or allowed operations for authors with potential and compatibility threshold 0.4. The increases are largest-possible for sumCite and unionCite.

Figure 7 shows the *h*-index increases that can be achieved by changing a certain number of articles (in the rows containing the conservative problem variants) or with a certain number of operations (in the row containing the cautious problem variant) for compatibility threshold 0.4. For the majority of ai10-2013 and ijcai-2013 authors we can see that, if manipulation is possible, then the maximum *h*-index increase can be reached already by manipulating at most two articles and performing at most two unmerges. The more senior authors in the ai10-2011 data set can still gain increased *h*-indices by manipulating four articles and performing four unmerges. For the outlier in ijcai-2013 with an *h*-index increase of 5, we see that there is one merged article that contains many atomic articles with citations above her or his unmanipulated *h*-index: With respect to an increasing number of operations, we see a continuously increasing *h*-index for Cautious Extracting compared to a constant high increase for Conservative Atomizing.

Summarizing, our findings indicate that realistic profiles from academically young authors cannot in the majority of cases be manipulated by unmerging articles. If they can, then in most cases the achievable increase in *h*-index is at most two. Furthermore, our findings indicate that the increase can be obtained by tampering with a small number of merged articles (at most two in the majority of cases).

6. CONCLUSION

In summary, our theoretical results suggest that using fusionCite as a citation measure for merged articles makes manipulation by undoing merges harder. From a practical point of view, our experimental results indicate that author profiles with surprisingly large *h*-index may be worth inspecting concerning potential manipulation.

Regarding theory, we leave three main open questions concerning the computational complexity of Extracting(fusionCite), the parameterized complexity of Dividing(fusionCite), and the parameterized complexity of Cautious Dividing (sumCite / unionCite) with respect to *h* (see Table 1), as the most immediate challenges for future work. Also, finding hardness reductions that produce more realistic instances would be desirable. From the experimental side, evaluating the potentially possible *h*-index increase by splitting on real merged profiles would be interesting as well as computational experiments using fusionCite as a measure. Moreover, it makes sense to consider the manipulation of the *h*-index also in context with the simultaneous manipulation of other indices (e.g., Google's i10-index; see also Pavlou and Elkind [2016]) and to look for Pareto-optimal solutions. We suspect that our algorithms easily adapt to other indices. In addition, it is natural to consider combining merging and splitting in manipulation of author profiles.

ACKNOWLEDGMENTS

The authors thank Clemens Hoffmann and Kolja Stahl for their excellent support in implementing our algorithms and performing and analyzing the computational experiments.

Manuel Sorge's work was carried out while affiliated with Technische Universität Berlin, Berlin, Germany; Ben-Gurion University of the Negev, Beer-Sheva, Israel; and University of Warsaw, Warsaw, Poland.

AUTHOR CONTRIBUTIONS

René van Bevern: Conceptualization, Formal Analysis, Writing—original draft, Writing—review & editing, Visualization. Christian Komusiewicz: Conceptualization, Formal Analysis, Writing—original draft, Writing—review & editing, Data curation, Software. Hendrik Molter: Conceptualization, Formal Analysis, Writing—original draft, Writing—review & editing,

Software. Rolf Niedermeier: Conceptualization, Formal Analysis, Writing—original draft, Writing—review & editing. Manuel Sorge: Conceptualization, Formal Analysis, Writing—original draft, Writing—review & editing, Data curation, Visualization, Software. Toby Walsh: Conceptualization, Formal Analysis, Writing—original draft, Writing—review & editing.

COMPETING INTERESTS

The authors have no competing interests.

FUNDING INFORMATION

We acknowledge support by the Open Access Publication Fund of TU Berlin. René van Bevern was supported by Mathematical Center in Akademgorodok, agreement No. 075-15-2019-1675 with the Ministry of Science and Higher Education of the Russian Federation. Hendrik Molter was supported by the DFG, projects DAPA (NI 369/12) and MATE (NI 369/17). Manuel Sorge was supported by the DFG, project DAPA (NI 369/12), by the People Programme (Marie Curie Actions) of the European Union's Seventh Framework Programme (FP7/2007-2013) under REA grant agreement number 631163.11, the Israel Science Foundation (grant no. 551145/14), and the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme under grant agreement number 714704. Toby Walsh was supported by the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme under grant agreement number 670077.

DATA AVAILABILITY

The data sets used in this paper were originally collected by van Bevern et al. (2016b) and are available under http://gitlab.com/rvb/split-index.

REFERENCES

Bartneck, C., & Kokkelmans, S. (2011). Detecting *h*-index manipulation through self-citation analysis. *Scientometrics*, *87*(1), 85–98. **DOI:** https://doi.org/10.1007/s11192-010-0306-5, **PMID:** 21472020, **PMCID:** PMC3043246

Bodlaender, H. L., & Kreveld, M. van. (2015). Google Scholar makes it hard–the complexity of organizing one's publications. *Information Processing Letters*, *115*(12), 965–968. **DOI:** https://doi.org/10.1016/j.ipl.2015.07.003

Boppana, R., & Halldórsson, M. M. (1992). Approximating maximum independent sets by excluding subgraphs. *BIT Numerical Mathematics*, *32*(2), 180–196. **DOI:** https://doi.org/10.1007/BF01994876

Cygan, M., Fomin, F. V., Kowalik, L., Lokshtanov, D., Marx, D., Pilipczuk, M., Pilipczuk, M., & Saurabh, S. (2015). *Parameterized Algorithms*. Heidelberg: Springer. **DOI:** https://doi.org/10.1007/978-3-319-21275-3

de Keijzer, B., & Apt, K. R. (2013). The *h*-index can be easily manipulated. *Bulletin of the EATCS*, *110*, 79–85.

Delgado López-Cózar, E., Robinson-García, N., & Torres-Salinas, D. (2014). The Google Scholar experiment: How to index false papers and manipulate bibliometric indicators. *Journal of the Association for Information Science and Technology*, *65*(3), 446–454. **DOI:** https://doi.org/10.1002/asi.23056

Downey, R. G., & Fellows, M. R. (2013). *Fundamentals of Parameterized Complexity*. Heidelberg: Springer. **DOI:** https://doi.org/10.1007/978-1-4471-5559-1

Egghe, L. (2006). Theory and practise of the *g*-index. *Scientometrics*, *69*(1), 131–152. **DOI:** https://doi.org/10.1007/s11192-006-0144-7

Faliszewski, P., & Procaccia, A. D. (2010). AI's war on manipulation: Are we winning? *AI Magazine*, *31*(4), 53–64. **DOI:** https://doi.org/10.1609/aimag.v31i4.2314

Faliszewski, P., Hemaspaandra, E., & Hemaspaandra, L. A. (2010). Using complexity to protect elections. *Communications of the ACM*, *53*(11), 74–82. **DOI:** https://doi.org/10.1145/1839676.1839696

Flum, J., & Grohe, M. (2006). *Parameterized Complexity Theory*. Heidelberg: Springer.

Hirsch, J. E. (2005). An index to quantify an individual's scientific research output. *Proceedings of the National Academy of Sciences of the United States of America*, *102*(46), 16569–16572. **DOI:** https://doi.org/10.1073/pnas.0507655102, **PMID:** 16275915, **PMCID:** PMC1283832

Jansen, K., Kratsch, S., Marx, D., & Schlotter, I. (2013). Bin packing with fixed number of bins revisited. *Journal of Computer and System Sciences*, *79*(1), 39–49. **DOI:** https://doi.org/10.1016/j.jcss.2012.04.004

Jukna, S. (2001). *Extremal Combinatorics – with Applications in Computer Science*. Texts in Theoretical Computer Science. Heidelberg: Springer. **DOI:** https://doi.org/10.1007/978-3-662-04650-0

Lesk, M. (2015). How many scientific papers are not original? *Proceedings of the National Academy of Sciences of the United States of America*, *112*(1), 6–7. **DOI:** https://doi.org/10.1073/pnas.1422282112, **PMID:** 25538304, **PMCID:** PMC4291619

Niedermeier, R. (2006). *Invitation to Fixed-Parameter Algorithms*. Oxford Lecture Series in Mathematics and Its Applications.

Oxford: Oxford University Press. **DOI:** https://doi.org/10.1093/acprof:oso/9780198566076.001.0001

Oravec, J. A. (2017). The manipulation of scholarly rating and measurement systems: Constructing excellence in an era of academic stardom. *Teaching in Higher Education*, *22*(4), 423–436. **DOI:** https://doi.org/10.1080/13562517.2017.1301909

Pavlou, C., & Elkind, E. (2016). Manipulating citation indices in a social context. In *Proceedings of the 15th International Conference on Autonomous Agents and Multiagent Systems (AAMAS '16)* (pp. 32–40).

van Bevern, R., Komusiewicz, C., Molter, H., Niedermeier, R., Sorge, M., & Walsh, T. (2016a). *h*-Index manipulation by undoing merges. In *Proceedings of the 22nd European Conference on Artificial Intelligence (ECAI '16)*, Frontiers in Artificial Intelligence and Applications, (pp. 895–903).

van Bevern, R., Komusiewicz, C., Niedermeier, R., Sorge, M., & Walsh, T. (2016b). *h*-Index manipulation by merging articles: Models, theory, and experiments. *Artificial Intelligence*, *240*, 19–35. **DOI:** https://doi.org/10.1016/j.artint.2016.08.001

Vinkler, P. (2013). Would it be possible to increase the Hirsch-index, π-index or CDS-index by increasing the number of publications or citations only by unity? *Journal of Informetrics*, *7*(1), 72–83. **DOI:** https://doi.org/10.1016/j.joi.2012.08.001

Woeginger, G. J. (2008). An axiomatic analysis of Egghe's *g*-index. *Journal of Informetrics*, *2*(4), 364–368. **DOI:** https://doi.org/10.1016/j.joi.2008.05.002