

Using Clustering Strategies for Creating Authority Files

James C. French* and Allison L. Powell

Department of Computer Science, University of Virginia, Charlottesville, Virginia 22903.

E-mail: {french|alp4g}@cs.virginia.edu

Eric Schulman

Institute for Defense Analyses, 1801 N. Beauregard Street, Alexandria, VA 22311. E-mail: eschulma@ida.org

As more online databases are integrated into digital libraries, the issue of quality control of the data becomes increasingly important, especially as it relates to the effective retrieval of information. Authority work, the need to discover and reconcile variant forms of strings in bibliographic entries, will become more critical in the future. Spelling variants, misspellings, and transliteration differences will all increase the difficulty of retrieving information. We investigate a number of approximate string matching techniques that have traditionally been used to help with this problem. We then introduce the notion of approximate word matching and show how it can be used to improve detection and categorization of variant forms. We demonstrate the utility of these approaches using data from the Astrophysics Data System and show how we can reduce the human effort involved in the creation of authority files.

1. Introduction

There are increasingly more online databases in the current climate of electronic publishing. The challenge is to integrate them into coherent digital libraries that let users have unimpeded access to accurate information. As the pace of electronic publication accelerates, there will be expanding reliance on automated techniques to aid information providers as they seek to reach this goal.

For a number of years, there has been a growing emphasis on data quality in online databases (O'Neill & Vizine-Goetz, 1988; Strong et al., 1997). In this paper, we look at techniques to aid in detecting variant forms of strings in bibliographic databases. This is called authority work (Auld, 1982), and results in the creation of authority files that maintain the correspondence between all of the allow-

able forms for strings in a particular bibliographic field, for example author or journal name. In this paper, we look at techniques that aid in detecting variant forms of strings in bibliographic databases.

Taylor (1984) elucidates two principles of authority control. The first is that all variants of a name will be brought together under a single form so that once users find that form, they will be confident that they have located everything relating to the name. The second ensures that a user will find a name if the catalog has it. Although Taylor's study casts doubt on the utility of the second principle, the first is declared to be the "absolutely indispensable part of authority control" (Taylor, 1984, p. 15). It is this aspect of authority control that we are trying to support with the work described in this paper. More concretely, we are trying to automate this authority control mechanism to achieve a transparent facility having the characteristics described by Auld (1982).

A bibliographic record, together with all variant forms of each associated heading, would be entered into the system. The computer would establish linkages between the preferred forms of headings and the bibliographic record. When a user keyed in a known form of a heading, the system would follow the internal linkages and display the requested item even though the preferred form of the heading might be quite different from the form entered. [. . .] to the user it would appear that a direct linkage existed between the form of heading entered and the bibliographic record displayed. The authority control mechanism would be invisible so far as the user was concerned. (Auld, 1982, p. 327)

Although this is not a new problem (Auld, 1982), it is increasingly important because of the proliferation of online databases and the need to provide effective access to these resources. The age of digital libraries is dawning rapidly and will demand efficient solutions as more disparate online resources are aggregated into cohesive collections. High

* To whom all correspondence should be addressed.

quality data collections are the bedrock upon which advanced digital library services will be built, and these services will rely on quality for their efficacy.

Problems arise when bibliographic databases are integrated. For example, the different component databases might use different authority conventions. Users familiar with one set of conventions will expect their usual forms to retrieve relevant information from the entire collection when searching. Therefore, a necessary part of the integration will be the creation of a joint authority file in which classes of equivalent strings are maintained. These equivalence classes can be assigned a canonical form that, in principle, could be substituted for the original strings in the combined database. In practice, this will generally be impractical or impossible because of intellectual property constraints. An institution may be willing to allow searches of its database, but not be willing to give up ownership of the data or to convert the data to comply with a format used by another institution. Therefore, component databases may not be physically combined, but instead logically combined by means of a distributed search mechanism. A mapping will have to be maintained between searchers and systems that hides this heterogeneity from users. Tools are needed to automate this process. Techniques that underlie effective tools are the topic of this paper.

In the remainder of this paper, we describe a particular instance of authority work, the creation of affiliations for authors in the Astrophysics Data System (ADS). Although we describe the techniques within the framework of a particular application, it is clear that they are more broadly applicable to authority work for other bibliographic fields. The construction of authority files for fields in bibliographic records is an extremely difficult problem for which it would be unreasonable to expect a fully automated solution. Manual intervention by domain specialists will be required to help with certain aspects such as resolving abbreviations or expanding acronyms; the challenge is to minimize human interaction. Note that for some domains, it might be possible to incrementally encapsulate the necessary knowledge and thereby require increasingly less human interaction.

In the next section, we describe the particular problem to provide a framework for discussion. We then outline the approach and give an overview of the experiments reported. In Section 5, we show how approximate string matching can be used to help with the problem. We propose approximate word matching as a mechanism for overcoming some of the remaining shortcomings. We present a comparison of performance and propose a refinement to strict distance-based approaches that we have been using in a large-scale application. Finally, we conclude with an estimate of the reduction in human effort required made possible by our automated approach.

2. The ADS Database—A Case Study

For the purposes of exposition, we will consider a concrete problem where uncertainty in the data requires attention. We would like to automate the process of specifying

canonical forms for text strings occurring in a database of bibliographic records. This is a step toward automating authority control.

We are collaborating with astronomers at the Smithsonian Astrophysical Observatory who have provided us with bibliographic records from the Astrophysics Data System (ADS) (Accomazzi et al., 1997). The ADS is an extensive collection of bibliographic data, abstracts, and full text from astronomy and astrophysics journals and conference proceedings. It contains approximately 450,000 entries for articles from over 1000 journals and conference proceedings. Some of these sources are indexed beginning in the 1870s. Our experiments (French et al., 1997b) are performed on a subset of this database containing approximately 146,000 refereed articles. The experiments reported in this paper are performed on a smaller subset of approximately 85,000 refereed articles that appeared in seven of the largest astronomy and astrophysics journals.¹ This smaller subset of 85,000 articles was used in a bibliographic study of journal productivity, and is the database that we will refer to in the rest of the paper. These articles were extracted from a snapshot of the ADS taken on September 12, 1997.

We use the smaller database here because our data was derived from efforts associated with the bibliographic study mentioned previously. The fact that we have limited our consideration to seven journals has no effect on our results. These seven journals exhibit enormous variability in the structure of affiliations, and that is the only issue here.

The astronomy community collects statistics about publication in that field, tracking changes in measures such as paper length, general productivity, and institutional productivity. Traditionally, such statistics have been gathered by hand on a necessarily small subset of available documents (Abt, 1993; Trimble, 1984). We have been collaborating with astronomers interested in automatically gathering this information using the ADS database as the data source (Schulman et al., 1996, 1997a,b). Automatically gathering statistics about these electronic documents has allowed a much larger fraction of documents to be considered—it has also presented new challenges. For example, although it is relatively simple to compute statistics such as number of papers per journal for each calendar year, statistics involving authors and their affiliations are more difficult. Before we can determine how many papers were written by an individual or by someone at a given institution, we must first be able to reliably identify that individual or institution. Because the ADS does not currently have a canonical list of author affiliations, we will need to derive it from the data. Errors and inconsistencies in the data make this challenging.

The larger problem with which we are faced is a lack of authority control in the ADS database. This problem exists

¹ The *Astrophysical Journal*, the *Astrophysical Journal Supplement Series*, the *Astronomical Journal*, *Publications of the Astronomical Society of the Pacific*, *Astronomy and Astrophysics*, the *Astronomy and Astrophysics Supplement Series*, and *Monthly Notices of the Royal Astronomical Society*.

Affiliation string	Number of occurrences
Univ. of Virginia, Charlottesville, VA, US	1
Univ. of Virginia, Charlottesville, VA, US	1
Univ. of Virginia, Charlottesville, VA, US	38
Univ. of Virginia, Charlottesville, VA, US	1
Univ. of Virginia, VA, US	1
University of Virginia, Charlottesville, VA, US	21
University of Virginia, Charlottesville, Virginia, US	1
University of Virginia, Virginia, US	1
Virginia, University, Charlottesville, VA	1
Virginia Univ.	2
Virginia Univ., Charlottesville	51
Virginia Univ., Charlottesville, VA	1
Virginia Univ., Charlottesville, VA, US	4
Virginia University, Charlottesville	1
Virginia University, Charlottesville, VA	1
Virginia, University	53
Virginia, University, Charlottesville	164
Virginia, University, Charlottesville, VA	60
Virginia, University, Charlottesville, Va.	60

FIG. 1. Raw affiliation strings for the University of Virginia.

in every bibliographic field of the ADS records—authors' full names or initials may be used, journal titles may or may not be abbreviated, and author affiliations are used very inconsistently. We do not mean to imply that the ADS is unique in this regard—this is a very common situation in many online databases. It exists primarily because the data often come from a variety of sources, and merging data consistently is so labor intensive.

The specific challenge facing us is partially illustrated by Figure 1. There are preferred naming schemes for affiliations, but the ADS database does not use any of them consistently. Institution names are recorded in a variety of formats and include a range of information—from terse, abbreviated names to full names with complete postal addresses. Figure 1 shows the different names that appear in the database for the University of Virginia and a count of the number of times each variant appears.

The example in Figure 1 shows 19 variants for what is obviously a single affiliation occurring in 463 articles. These variants illustrate a subset of the causes of the inconsistencies. These causes include misspelled words and permuted word order. In addition, common terms—such as University—may or may not have been abbreviated, and full addresses of sites were sometimes used. Some affiliations that included a full address had abbreviated state or province names, and some included postal codes and/or country names whereas others did not.

This is merely the tip of the iceberg. In addition to the causes listed here and illustrated in the example, there are multiple other potential causes. These include acronyms that were used inconsistently, affiliation names that changed over time, varying transliteration conventions from non-Latin alphabets, and many translation variants when non-English language affiliation names were keyed by native English speakers.

In this paper, we present the results of our efforts to identify the unique institutions listed as author affiliations in the ADS database. Many of these approaches are also applicable to other bibliographic fields. We must also provide a way to map variants of an institution's name to the canonical name for that institution. This is necessary so that

corrections can be made to the database and so that new entries can be verified or corrected before being added to the database. Our goal is to resolve the variants, determine the canonical set of sites and produce a mapping from the variants of the site names found in the database to the canonical set using predominately automated methods. Others have considered similar problems with variant forms in bibliographic fields, for example, author names (Siegfried & Bernstein, 1991) and titles (Williams & Lannom, 1981). Borgman and Siegfried (1992) survey many other name-matching algorithms. In addition, there is a vast literature on string matching in bioinformatics growing out of the genomic sequencing efforts. Interested readers are referred to Gusfield (1997) for a survey.

This paper is not specifically concerned with the algorithmic aspects of string matching. Rather, we are examining the application of known techniques together with those developed here to the problem of authority control. We are using the ADS as a concrete example. This data cleanup effort will make it possible for the ADS to provide services that are currently infeasible. Such services include an index of all papers with authors from a given institution and a definitive list of all papers by a particular author.

3. Approach

To attack this problem, we must do two things: (1) decide on a set of canonical affiliation strings; and (2) assign each affiliation string in the database to one of these canonical strings. A related application is to decide the set of canonical forms and then replace all variant instances with the standard form. The two subproblems outlined here still apply.

Our approach to this problem is to apply a clustering algorithm to the set of strings to bring variants together. There is absolutely no way that this can be completely automated using lexical techniques alone. Consider the following two strings for example.

Leander McCormick Observatory, Charlottesville
 Leander J. McCormick Observatory, Charlottesville

Although we might reasonably deduce that they designate the same place, there is no way to determine that McCormick Observatory is really part of the Astronomy Department at the University of Virginia based on lexical considerations alone. Clearly, additional domain knowledge would have to be introduced. Because we cannot completely automate the process, we concentrate on clustering the affiliations with the goal of minimizing the amount of manual inspection necessary to make final assignments. So our approach is to: (1) extract the strings; (2) cluster them as aggressively as possible consistent with the goal of minimizing misclassification; (3) have a domain expert review the outcome of step 2 with some automated aids and iterate until a final list has been produced; and (4) optionally, synthesize programs to (a) substitute the canonical forms for

the variant forms in the original database, and (b) screen newly entered data for conformance with the standard forms. Steps 1–3 are detailed in this paper.

4. Overview of Experiments

We have employed multiple approaches to resolving variant forms of affiliations in the database. We currently apply a combination of lexical steps, approximate string matching, manipulation of the affiliation strings, and approximate word matching. Many of our later approaches were motivated by observations about results from earlier experiments. Therefore, experiments are presented in the order in which they were performed so that they can be described in the context of the results that motivated them.

As an initial step, raw affiliations were extracted from the affiliation fields of the records in the ADS database. We maintained a list of each unique affiliation variant and the frequency with which that variant appeared. There were 13,884 unique variants in the subset of the ADS under consideration.

4.1. Evaluation

An issue that bears discussion is the evaluation of the multiple clustering approaches described in this paper. There are a number of applications for which authoritative affiliation data could be utilized. For each application, a different goal standard or ideal clustering of the raw data may be most appropriate. Two different applications may require clustering of different granularity. For example, for one application it may be appropriate to group all departments within a university together, for another it may not.

The original goal was to cluster all 13,884 strings to produce a canonical authority file for all institutions that published articles in the seven journals of interest. We discovered that it was very difficult for even a domain expert to be certain that all strings were placed correctly. As a result, we focused on 38 specific institutions. These 38 institutions were used by Abt (1993) in a previous productivity study. In our dataset, there were 1745 strings that represented variants of those 38 institutions. We produced a very thorough hand-clustering of those 1745 institutions into 38 clusters to produce both an authority file for productivity experiments and a goal standard for testing our clustering approaches.

For evaluation purposes, we assumed that the goal for each automatic clustering technique was both to locate the 1745 strings of interest among the 13,884 original strings and to place those 1745 strings correctly in the 38 clusters. More specifically, the goal was to reproduce the goal standard of 38 clusters of 1745 strings; the remaining 12,139 strings could be clustered in any way so long as they did not appear in one of the 38 standard clusters. This task is more difficult than simply clustering the 1745 strings—more clusters must be examined, and the 1745 strings of interest may be intermingled with the remaining 12,139 strings.

We discovered that it is difficult to characterize the quality of the results of a clustering approach when only a subset of the strings is of interest. We wanted to capture the amount of manual work that would be required if a domain expert (a) verified the correctness of a set of clusters, (b) corrected the set of clusters by removing misplaced strings, and/or (c) placed together multiple clusters containing strings representing the same institution. We determined that a single evaluation measure was not adequate to convey all of this information. Therefore, we employed several measures intended to characterize both the completeness of clustering and cluster quality.

To convey the magnitude of the task, we first report the total number of clusters produced when the 13,884 original strings are clustered. All other measures are calculated using all clusters that contain at least one of the 1745 strings of interest. These measures include:

- **Purity of clusters.** Cluster purity is a measure of self-consistency. It is the ratio of the number of clusters containing no misclassifications to the total number of clusters produced. The denominator of this measure shows the number of clusters containing strings of interest.
- **Number of singleton clusters.** Related to the purity of clusters measure is the number of singleton clusters. This is the number of clusters containing only one string. All singleton clusters are trivially pure, so this measure should be considered in conjunction with the purity of clusters measure.
- **Number of strings incorrectly placed.** The number of strings placed into clusters when they should not have been. Specifically, assume strings *a* and *b* belong in cluster 1 and string *c* belongs in cluster 2. If a clustering approach places strings *a*, *b*, and *c* in a single cluster, string *c* will be marked as misplaced because it is in the minority. We also maintain a count of the clusters containing incorrectly placed strings.
- **Number of external strings.** For this application, there are 12,139 strings that are not of interest. This measure reports how many of these strings are placed in clusters with any one of the 1745 strings of interest. A cluster can contain both misplaced and external strings.

These measures are not perfect. For example, placing one of the 1745 strings of interest in a large cluster of external strings will produce a high value for the external measure when in fact correcting this error is relatively simple. However, when considered on a whole, these measures characterize clusters well enough to allow a comparison of the different techniques. In Section 7, we will relate these measure to the human effort involved to postprocess the clusters into a final authority file.

When examining the results, it is useful to compare across approaches at similar error levels. What constitutes an acceptable error level depends on the stage of clustering and the patience of the person examining the clusters. We foresee using these approaches in an iterative fashion. After each step, a representative string may be chosen for each cluster; the representative strings could be clustered in the following step. For example, as an initial step, one might

choose to reduce the number of strings to be considered by performing a very conservative clustering step without checking the output. Later, one might choose more aggressive clustering to suggest items for inclusion in a cluster. For an application such as the one described previously where only a subset of strings are of interest, one might exclude from further consideration clusters that contain no strings of interest.

Although operationally useful, iterative steps such as this would make it difficult to compare approaches directly. Where possible, we use different approaches to cluster the same set of strings to facilitate effectiveness comparison.

5. Experiments and Results

5.1. Lexical Cleanup

Upon examining the list of raw affiliations, we noted that many variants were caused by (a) the inconsistent use of abbreviations and acronyms, and (b) the range of information included in the postal addresses for sites. These are systematic differences between variants. A given abbreviation, acronym or address format may appear in many different affiliation variants; resolving any given difference may resolve multiple variants.

The lexical cleanup steps described below include expanding abbreviations and acronyms and removing some of the additional information included in postal addresses. An initial step is the identification of acronyms and abbreviations and the items that they represent. A few simple rules can help identify potential abbreviations and acronyms within a string. In some cases, general knowledge is sufficient to provide expansions for these abbreviations; for the specific problem reported in this paper, many of the transformations were based upon general knowledge about variants on place names (e.g., Blvd. is an abbreviation for Boulevard). However, the expertise of a domain expert may be needed to supply the correct expansion for certain abbreviations and acronyms. For example, the abbreviation NRC has several possible expansions including the National Research Council of Canada and the Nuclear Regulatory Commission of the United States. Only the former appears in the ADS.

Similarly, the identification of extraneous information may or may not require the expertise of a domain expert. For this application, a common type of extraneous information was additional postal information. General knowledge about place names and examination of the database was sufficient to identify this information. We noted that in general, detailed postal information, including mail codes and country names is not an intrinsic part of an institution's name. City names, however, tend to denote different campuses of a university or different sites for a company or institute. Consider the example below.

Institute of Astronomy, Cambridge
 Institute of Astronomy, Cambridge, England

TABLE 1. Reduction in the number of distinct affiliations using lexical cleanup approaches.

Cleanup method (applied sequentially in the order listed)	Number of distinct affiliations	Δ
None	13,884	
Remove U.S., U.S.A., etc. if occurring at the end of an affiliation	13,676	208
Remove U.S. ZIP codes from end of affiliations	13,638	38
Remove U.S. state abbrevs. occurring at the end of an affiliation	12,951	687
Expand most obvious abbreviations (University, Institute, etc.)	12,295	656
Expand other selected abbreviations and acronyms	12,171	124
Remove country names occurring at the end of an affiliation	11,382	789
Normalize case (all lowercase)	11,289	93

Institute of Astronomy, Cambridge, U.K.
 Institute of Astronomy, Cambridge, United Kingdom

Although the country information is useful and arguably should be included in the final canonical form for this site, at this stage of comparison the additional information obscures the similarities among strings.

The lexical cleanup steps were applied in the order listed in Table 1. Most steps could be combined into a single operation; they are presented here separately so that the impact of individual steps can be assessed. Other steps are simpler to perform if done sequentially, but not impossible if performed as a single step. For example, a site's postal address (if included) is found at the end of the affiliation string. Knowing this, we can strip country names, postal codes, and state abbreviations from the end of the affiliation string. By working from the end of the string, we can be more confident that we are not erroneously removing important information. For example, given an affiliation University of VA, Charlottesville, VA, US, it should be useful to remove US and the second VA, but removing the first VA would eliminate useful information. Finally, we note that acronyms must be identified and expanded before the strings are converted to lowercase.

Lexical cleanup pays two dividends. When the cleanup step is performed, some differences in affiliation strings are removed, allowing those affiliation variants to become identical. In Table 1, the Δ column shows the difference between the current number of distinct affiliations and the number in the previous step. In some steps, the apparent immediate payoff is small. However, the second payoff is seen when the clustering is performed. Consider the following affiliations.

NASA, Goddard Space Flight Center
 National Aeronautics and Space Administration, Goddard Space Flight Center

Expanding the NASA acronym would not cause the strings to become identical because of the spelling error in

“Goddard” in the second string. However, because the difference in the strings is small after the expansion, these affiliations would be clustered together using even a very conservative clustering approach.

Lexical approaches provided dramatic improvements in a few cases. However, the best immediate results were seen when common abbreviations and acronyms were tackled. This approach rapidly reaches a point of diminishing returns. Many types of affiliation variants, including spelling errors, word permutation, and affiliations that include city names, do not respond well to lexical measures. In addition, identifying misspelled words and variant transliterations of words is a time-consuming activity. It simply is not efficient to correct infrequently occurring variants in this way. Therefore, we consider clustering techniques to resolve these types of variants.

5.2. Clustering Techniques

There are many approaches to data clustering (Jain & Dubes, 1988), but they are often costly to compute and very time consuming for large datasets. We developed our own heuristic approach for this application, and applied it to the output of the lexical cleanup approaches. Our approach is a one pass algorithm that requires a distance function; we report the effect of different distance functions below.

We used the 11,289 normalized case strings unless otherwise noted. For these approaches, we have a set of unique strings $\{s_i\}$ together with the occurrence frequency of each string. The input to our clustering algorithm is a set of pairs $\{(s_i, f_i)\}$ denoting the strings and their frequencies.

Given some distance metric $d(x, y)$, our heuristic clustering algorithm proceeds as follows.

Step 1. Sort the strings in descending order by frequency of occurrence.

Step 2. Starting with the first (next) string, compute the distance $d(s_i, s_j)$ for all strings $s_j, j > i$. When $d(s_i, s_j) < \delta$, where δ is a threshold parameter, (1) insert s_j into cluster C_i , and (2) remove s_j from further consideration. Note: By construction, $f_i \geq f_k \forall s_k \in C_i$, that is, s_i is the most frequently occurring string in the cluster C_i so it is the algorithm’s nominee for the canonical label for that cluster.

Step 3. Increment i until s_i is a string still under consideration for clustering and repeat step 2.

The performance of single-pass clustering algorithms often depends on the order of the data. Step 1 of our approach places the data in a canonical order and is based on the following hypothesis: strings occurring frequently have a greater chance of being accurate. So our approach is to process the most frequently occurring strings first, finding all similar strings and removing them from further consideration.

The choice of the distance metric, $d(x, y)$, plays a crucial role in our clustering approach. We are trying to coalesce similar strings while accounting for misspellings and other variant forms. The next section discusses the well-known edit distance metric (Lowrance & Wagner,

$$e(\text{“university”}, \text{“institut”}) = 8$$

		university
1	S(i,u)	in iversity
2	D(i)	in versity
3	D(v)	in ersity
4	D(e)	in rsity
5	D(r)	in sity
6	I(t)	in stity
7	S(u,y)	in stitu
8	I(t)	in stitut
	S(x,y)	substitute x for y
	D(x)	delete x
	I(x)	insert x

FIG. 2. Transformation of the string “university” to the string “institut” by a minimal sequence of simple edit operations. The prefix of the final string is shown in boldface after each edit operation.

1975; Wagner & Fischer, 1974) and our proposed metric in more detail.

5.3. Approximate String-Matching Techniques

Our first experiments considered each affiliation string as a whole. The edit distance, $e(u, v)$, of two strings is a measure of dissimilarity that is given by the minimal number of simple edit operations needed to transform u into v or *vice versa*. The four simple edit operations considered here are insertion of one character, deletion of one character, substitution of one character by another, and transposing two adjacent characters. An example of the concept of edit distance is shown in Figure 2.

In information retrieval, edit distance has traditionally been used in approximate string matching (Hall & Dowling, 1980), spelling error detection and correction (Kukich, 1992), and more recently has been shown to be more effective than Soundex for phonetic string matching (Zobel & Dart, 1996). The edit distance measure is robust to spelling variants such as could occur when non-Latin alphabets are transliterated (e.g., Chebyshev and Tchebysheff have an edit distance of 3) or when misspellings occur.

There is the issue of how to set the threshold, δ , in step 2 when using edit distance. We have investigated both absolute and relative thresholds and discuss this issue next.

5.3.1. Fixed threshold edit distance clustering

We first investigated the straightforward fixed threshold edit distance clustering. The results of this approach are listed in Table 2.

Note that an edit distance of one provides significant improvement with no misplacement errors for the 1745 strings of interest. A visual examination of all 9430 clusters also revealed no errors. We hypothesize that all of these variants are simple typing errors.

Increasing the edit distance threshold allows more affiliations to be clustered together and decreases the number of clusters. This approach makes no errors at edit distance 1, and a small number of errors up to edit distance 3. However,

TABLE 2. Fixed threshold edit distance clustering using lexically cleaned up affiliation set (11,289 affiliation strings).

Edit distance	Total clusters	Purity of clusters	Singleton clusters	Incorrectly placed		External	
				Strings	In x clusters	Strings	In x clusters
1	9430	952/952	674	0	0	0	0
2	8970	922/923	642	0	0	3	1
3	8521	870/875	582	2	2	17	4
4	8007	809/826	525	4	3	72	15
5	7610	768/796	500	13	8	261	25
6	7116	714/757	466	27	14	447	40
7	6644	648/695	409	61	21	587	43
8	6132	577/636	359	77	21	724	53

the approach breaks down for larger thresholds. It is apparent that fixed threshold edit distance clustering alone will not be enough to group affiliation variants correctly. However, it is reasonable to assume that multiple typing errors or systematic transliteration variants could occur in a string. Our next approach is intended to handle this more effectively than fixed threshold clustering does.

5.3.2. Variable threshold edit distance clustering

An absolute threshold is problematic. Especially for longer affiliation names, it is reasonable to assume that more than one typing error can occur in a string. Minor variants in affiliation names, for example liberal use of commas, could also manifest themselves as a small edit distance greater than one. However, using an edit distance threshold large enough to cluster strings having these common differences can be problematic. As shown in Table 2, even a moderate fixed edit distance threshold can be inappropriate in many cases, especially when short affiliation strings are involved. For example, consider the strings

University of Virginia
University of Victoria

and

California Institute of Technology, Jet Propulsion
Laboratory, Pasadena

California Institute of Technology and Jet Propulsion
Laboratory, Pasadena

Both pairs differ by an edit distance of 4; however, for the first two strings, that difference represents a greater fraction of the characters found in the strings.

Variable threshold edit distance clustering is a way to keep stricter control over these short affiliation strings whereas allowing slightly more leeway for the longer ones. Assuming that two affiliation strings differ by an edit distance of 5, the two strings are more likely to be variants of the same affiliation if they both have more than 90 characters than if they both have fewer than 20 characters.

We suggest a relative threshold of the form $\delta = \alpha \min(|u|, |v|)$, that is, the threshold is some fraction, α , of the length of the shorter string. This definition protects shorter strings from being clustered indiscriminately. The results of variable threshold edit distance clustering are shown in Table 3.

Examining Tables 2 and 3, it is apparent that variable threshold edit distance clustering performs better than fixed threshold clustering. For example, compare the performance of fixed threshold edit distance 4 and variable threshold $\alpha = 0.15$. The numbers of incorrectly placed strings are similar; however, the variable threshold approach produces many fewer clusters at this error level and includes fewer external strings.

Table 3 also shows that the problem of incorrectly clustering affiliations still exists at larger relative distance values. Examining the clusters that were produced, we noticed

TABLE 3. Variable threshold edit distance clustering using lexically cleaned up affiliation set (11,289 affiliation strings).

Relative distance (α)	Total clusters	Purity of clusters	Singleton clusters	Incorrectly placed		External	
				Strings	In x clusters	Strings	In x clusters
0.05	8908	903/903	613	0	0	0	0
0.10	8120	813/814	523	1	1	0	0
0.15	7404	723/733	452	6	4	10	6
0.20	6553	617/648	360	17	8	65	27
0.25	5807	505/561	288	49	15	163	47
0.30	5089	404/491	217	80	27	397	79
0.35	4462	325/427	170	114	34	761	94

TABLE 4. Variable threshold edit distance clustering of unique-word affiliations (9472 affiliation strings).

Relative distance (α)	Total clusters	Purity of clusters	Singleton clusters	Incorrectly placed		External	
				Strings	In x clusters	Strings	In x clusters
0.05	8064	824/824	535	0	0	0	0
0.10	7269	719/719	444	0	0	0	0
0.15	6529	631/633	368	1	1	3	1
0.20	5842	540/556	302	4	3	39	14
0.25	5210	449/494	248	16	7	211	39
0.30	4660	374/445	203	50	18	493	61
0.35	4103	309/384	154	85	27	848	66

that variants of the same affiliation still differed significantly. Raising the edit distance threshold to a level that would cluster these variants together would also incorrectly cluster a significant portion of the affiliation set. We therefore considered other representations of the affiliation strings.

5.4. String Manipulation

We noted that a general edit distance comparison of affiliation strings could not accurately capture the similarity of two strings if the words in one string were a permuted order of the words in the other. In addition, we also noted that some strings contained duplicate words. The duplicate words rarely added useful information. We therefore used an alternate internal representation of the affiliation string—the unique words of the string listed in lexicographically sorted order. Words can occur in simple strings or as part of quoted phrases and parenthetical expressions. They can be delimited by spaces, commas and slashes or extraneous punctuation combined with any of the above. Our word-extraction procedure took this into account. We also removed numeric “words” (which were typically mail stops and postal codes).

This approach works well for affiliation strings and may also be appropriate for other bibliographic fields where the order of words in a string is not vital to the meaning of a string. We note that this (and following) approaches may not be appropriate for all bibliographic fields.

Note that this approach provides a double payoff similar to that noted in Section 5.1. Multiple affiliation strings can have the same representation, reducing the number of representations to consider. In addition, the sorted word order and lack of duplicates can reduce the edit distance between variants of the same affiliation. Table 4 shows the results of this approach. Note that at the same relative distance (α) this approach produces fewer clusters than variable threshold edit distance clustering of the lexically cleaned up affiliation set (see Table 3), although at slightly higher error rates.

Figure 3 illustrates some of the benefits and remaining weaknesses of using the string manipulation technique. Figure 3(a) shows two original affiliation strings with an approximate string matching edit distance of 36. Figure 3(b) shows an alternate representation of these strings formed by sorting the constituent words; using the alternate representation yields an edit distance of 22. However, in this case, even the sorted representation does not fully capture the similarity between the two strings. Most of the difference between the strings is found in one word per string, Institut and University.² Measuring the difference between those two words (plus the other minor differences in the strings) would produce a smaller and more representative edit distance. However, these words do not begin with the same character and therefore do not align properly to be compared using approximate string matching.

Our next approach worked to resolve this difficulty. In this approach, we seek to match words in each string to achieve a minimum sum of edit distances. An example is shown in Figure 3(c). This method is described more fully in the next section.

Moskovskii Gosudarstvennyi Pedagogicheskii Institut, Moscow
 Moskovskij Pedagogicheskij Gosudarstvennyj University, Moscow
 (a) Approximate string matching, edit distance 36

Gosudarstvennyi Institut Moscow Moskovskii Pedagogicheskii
 Gosudarstvennyj Moscow Moskovskij Pedagogicheskij University
 (b) String manipulation, edit distance 22

Moskovskii Gosudarstvennyi Pedagogicheskii Institut, Moscow
 | | | | |
 Moskovskij Pedagogicheskij Gosudarstvennyj University, Moscow
 | | | | |
 (c) Approximate word matching, edit distance 11

5.5. Word-Based Approaches

5.5.1. Approximate word matching

Although edit distance is robust to spelling variants, it can be completely defeated by permutations of words. Consider the five strings in Figure 4(a). They clearly belong in three classes, $\{s_1, s_2\}$, $\{s_3, s_4\}$, and $\{s_5\}$. Each of these classes represents a separate institution, and we would like to judge them as “similar” in the clustering algorithm. Figure 4(b) shows the pairwise edit distances for the strings

FIG. 3. Edit distances using different string representations.

² Figure 2 shows that the edit distance between these two words is 8.

s_1 : Moskovskii Gosudarstvennyi Pedagogicheskii Institut, Moscow
 s_2 : Moskovskij Pedagogicheskij Gosudarstvennyj Universitj, Moscow
 s_3 : Virginia, University
 s_4 : University of Virginia
 s_5 : University of Vermont

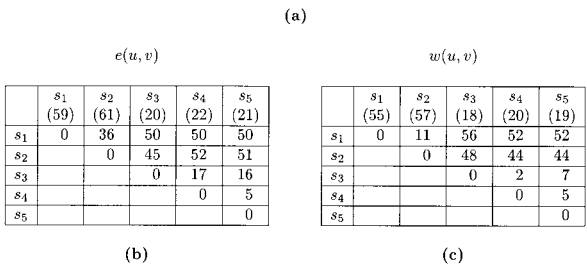


FIG. 4. A comparative example of $e(u, v)$ and $w(u, v)$.

of Figure 4(a). The length of each string is shown in parentheses. Note that $e(s_4, s_5) \ll e(s_3, s_4)$ so that University of Vermont is much more similar to University of Virginia than is Virginia, University. Also, in general, the edit distances are over one-half the string length. Any threshold large enough to accept these strings as similar would admit a large number of unrelated strings.

The problem here is mainly because of word permutations, although spelling variants still play a role. The conventional representation of a string s is a sequence of characters, $s = \langle c_i \rangle$. A more useful representation is to think of s as a set of words, $s = \{w_i\}$, where each “word,” $w_i = \langle c_{ij} \rangle$, is a sequence of characters. This representation provides more flexibility in the choice of comparators. We should also note that this set representation does not allow duplicate words. This has been advantageous in our work but there may be applications where a multiset is more appropriate.

The decision as to which tokens denote words is domain dependent. For the present purposes, we use a blank as the word separator and exclude punctuation. In general, punctuation will require more care in handling. This is discussed further in French et al. (1997b).

Our approximate word-matching approach is to find a minimum distance matching of the words in u and v . The idea is to pair up the words so that the sum of the edit distances is minimized. Note that if u and v contain a different number of words, then the cost of each excess word, w , is the edit distance between it and the null string, $e(w, \lambda) = |w|$, which is simply the length of the word. Our proposed word edit distance $w(u, v)$ is the sum of the edit distances resulting from a minimal matching. Figure 4(c) shows $w(u, v)$ for the strings of Figure 4(a). As with edit distance, we use a relative threshold for $w(u, v)$. However, we adjust the “length” of the new string representation to reflect the loss of blanks and perhaps other punctuation. The revised lengths are also shown in parentheses in Figure 4(c). Using a relatively conservative threshold of 0.2 of the length of the shorter string, it can be seen from Figure 4(c) that the strings of Figure 4(a) will be assigned to the correct clusters. Moreover, such a conservative threshold will allow relatively few unrelated strings to be grouped together.

Although it is not generally true that $w(u, v) < e(u, v)$, when word permutations are considered, it is almost always

the case that $w(u, v) \ll e(u, v)$. In addition, when there are no word permutations involved, $w(u, v) \approx e(u, v)$ for strings with the same number of words. For these reasons, $w(u, v)$ will often characterize similarity better than $e(u, v)$ for particular applications. The practical implication is that a lower threshold can often be used in the clustering algorithm, and this will result in fewer misclassifications.

Table 5 shows the results when approximate word matching is applied to our set of 9472 strings. This table should be compared to Table 4. The comparison of these approaches is less straightforward than the comparison of fixed-threshold and variable-threshold edit distance clustering. When comparing the results produced at a given value of α , approximate word matching outperforms clustering of the alternate string representations. However, when results at similar error levels are compared, the alternate string representations sometimes outperform approximate word matching.

5.5.2. Overlap

There are still situations that can cause problems when using $w(u, v)$. Because the measure is the sum of the component edit distances, all of the difference may be because of one component. For example, consider the following strings.

University of California, Davis
 University of California, Irvine

Here we have $w(u, v) = 4$, but all of the distance is because of $e(\text{“Davis,” “Irvine”})$.³ These two strings would be judged similar at a threshold $\alpha = .2$, but they are clearly two different institutions.

We can address this problem by constraining the allowable edit distance between two components. Damerau (1964) has noted that over 80% of all spelling errors are because of one of the four simple edit operations (insertion, deletion, substitution, and transposition). This has also been confirmed by Morgan (1970). So whenever $e(u, v) = 1$, it is almost certainly a spelling variant or misspelling. This observation leads to the following approach. First, as before, we find a minimal cost matching of the components in the string. Next, we determine what components match sufficiently well. This is simply a bound on the allowable edit distance for a component. If we constrain the component difference to be ≤ 1 , then we will almost certainly be dealing exclusively with spelling variants (e.g., center, centre, color, colour) or misspellings; a larger threshold will allow more variability and may be appropriate for some applications. The threshold is just a parameter of the algorithm and could be set to other values. After we determine how many words match sufficiently well, we compute a Jaccard coefficient, the ratio of the matching words in u and

³ It is also the case here that $e(u, v) = 4$ because there are no word permutations.

TABLE 5. Approximate word matching clustering of unique word affiliations (9472 strings).

Relative distance (α)	Total clusters	Purity of clusters	Singleton clusters	Incorrectly placed		External	
				Strings	In x clusters	Strings	In x clusters
0.05	7789	794/794	502	0	0	0	0
0.10	6875	673/674	402	1	1	0	0
0.15	6128	569/576	327	6	3	7	5
0.20	5347	473/503	255	9	6	66	26
0.25	4561	367/445	192	23	12	416	71
0.30	3740	256/381	138	110	38	1353	117

v to all of the words in u and v , to use as a similarity function for clustering.

The results of applying this approach to our set of 9472 strings are shown in Table 6. Although the threshold parameters in Tables 5 and 6 are not directly comparable, we can compare the approaches at similar error levels. For example, using the Jaccard approach with a similarity coefficient of 0.65 incorrectly places the same number of strings as approximate word matching at $\alpha = 0.20$. The Jaccard approach produces fewer clusters and places more strings but does include a few more external strings.

6. Additional Considerations

6.1. Applying the Approaches Iteratively

Where possible, to facilitate comparison, we have applied the approaches evaluated previously to the same set of strings. However, in an operational setting, these approaches might be applied iteratively. For example, assuming simple typographical errors, strings might first be clustered at a fixed edit distance of 1 but not verified. The cluster representatives could be used for later clustering steps. This would reduce the number of strings to be clustered with little chance of error. Using cluster representatives at each stage reduces the number of strings whose placement must be verified. Later, clustering stages might employ aggressive thresholds in an attempt to locate additional variants.

We have examined the impact of applying these approaches iteratively. We found that applying a given threshold (e.g., $\alpha = 0.20$) to representatives produced slightly different clusters than applying it to the original strings. This phenomenon is discussed in more detail in our previous work (French et al., 1997b).

6.2. Computation Time

Another consideration is that because $e(u, v)$ is calculated in $\Theta(mn)$ time by a dynamic programming algorithm, it is a fairly expensive distance metric to compute. Next, we show when the computation can be avoided, but first we need the following lemma.

Lemma. $||u| - |v|| \leq e(u, v) \leq \max(|u|, |v|)$.

Proof. The lower bound follows when one string is a substring of the other. In that case, it is only necessary to delete (insert) the excess characters from (into) the longer (shorter) string. The upper bound follows from the fact that if u and v are disjoint, it will be necessary to edit every character in the longer string. \square

To simplify the notation somewhat, let m denote the length of the shorter string and n denote the length of the longer. Recall that we have suggested the decision rule $e(u, v) \leq \alpha m$ to protect shorter strings. We state the following result using this simplified notation.

Theorem. $e(u, v) \leq \alpha m$ only if $n - m \leq \alpha m$.

Proof. From the lemma, $n - m \leq e(u, v) \leq \alpha m$ and the result is immediate. \square

TABLE 6. Clustering using the Jaccard Coefficient (9472 strings).

Similarity coefficient	Total clusters	Purity of clusters	Singleton clusters	Incorrectly placed		External	
				Strings	In x clusters	Strings	In x clusters
0.85	7365	731/736	446	0	0	6	5
0.80	6563	610/618	354	1	1	23	7
0.75	5897	529/544	290	1	1	47	14
0.70	5626	494/509	266	1	1	47	14
0.65	4935	414/435	212	9	5	109	19
0.60	4491	353/392	175	21	10	159	34
0.55	4275	322/363	158	18	8	221	38
0.50	3613	241/302	111	25	11	382	57
0.45	3542	230/297	105	39	12	491	63
0.40	3028	168/254	80	135	25	1000	80

Because $n - m \leq \alpha m$ is a necessary condition for $e(u, v) \leq \alpha m$, the expensive computation of $e(u, v)$ can be avoided when $n - m > \alpha m$. The following is an example of the benefit. In one of our subprocesses in French et al. (1997b), we computed the upper triangular matrix of pairwise edit distances for an 8380 word vocabulary. On a Sun Ultra, this took 938 seconds to compute. That time was reduced to 716 seconds after applying the test above. This was a net reduction in CPU time of 222 seconds or a 24% reduction in processing time.

We have also worked with other researchers to investigate an alternative clustering technique, RED (Ganti et al., 1999), based on incorporating relative edit distance into the BIRCH⁴ clustering algorithm (Zhang et al., 1996). Although this clustering technique still employs edit distance as the measure of difference between two strings, it creates a “distance space” for the purposes of clustering. We found that using the BIRCH system resulted in much faster clustering times but provided substantially less control over the number and type of errors allowed and was not well suited to this application.

6.3. Other Bibliographic Fields

It is reasonable to ask how these techniques might work on other data. We have also studied the application of these approaches to the journal title field of the ADS records. These results are reported in French et al. (1997a). Different abbreviations (e.g., J. and Jour. for Journal) and acronyms (e.g., PASP is *Publications of the Astronomical Society of the Pacific*) were appropriate for this data, but we found that the approximate word matching approach worked best for this field.

7. Human Effort Involved

So far, our evaluation of the different clustering approaches has focused on the number of clusters produced and the number of placement errors made. An open question is what impact these clustering approaches have on the amount of human effort required to complete some clustering task.

The amount of human effort required will depend on a number of factors, including the number of strings being clustered and the number of clusters into which the strings are to be placed. We have identified a set of costs involved in clustering a group of strings by hand and in verifying or correcting the output of an automated clustering approach.

As an example, we compare the human effort required for our specific application area using manual clustering vs. using our automated approximate word matching approach at thresholds 0.05, 0.10, 0.15, 0.20, 0.25, and 0.30 (see Table 5).

Details of our application (institutional productivity) are given in Section 4.1. Recall that there are 13,884 strings, 1745 of which are variants of the 38 institution names for which we wish to determine productivity.

Assume that we have some sort of user interface that allows strings or clusters to be examined in one window, maintains the set of destination clusters in another window, and allows a user to move a string or cluster from the examination window to the desired destination cluster.

We have identified different types of human effort, and assume that they each have a cost. c_{setup} is the cost to identify/select destination clusters, c_u is the cost of an unplaced string, c_m is the cost of a misplaced string, c_e is the cost to determine if a string is correctly placed in a cluster, and c_{merge} is the cost to merge two clusters.

To simplify the comparison between manual and assisted clustering, we assume that c_u is our basic measure of cost and that other costs can be expressed in terms of that cost. For example, for a given application, we might assume that it is twice as difficult to correct a misplaced string than an unplaced one. We express c_m as αc_u and c_e as βc_u .

Next, assume that a cluster can be moved as a unit. Therefore, the cost of moving a cluster (that contains a subset of the strings representing a site) to the correct destination cluster is the same as handling a single unplaced string. So

$$c_{\text{merge}} = c_u \quad (1)$$

If we perform the clustering manually, all 13,884 strings are essentially unplaced, so our cost is

$$\text{cost}_{\text{manual}} = 38 \cdot c_{\text{setup}} + 13,884 \cdot c_u \quad (2)$$

The cost of approximate word matching is a little more complex. At the threshold of 0.20, there are 5347 clusters. 3029 of those clusters contain only one string (not shown in Table 5), so the placement of those strings does not need to be verified; we will treat these 3029 strings as unplaced strings. 503 of the 5347 clusters contain strings of interest. There are 75 total misplaced strings (nine incorrectly placed strings of interest plus 66 external strings).

Assume that $\alpha = 2$ and $\beta = 0.2$. That is, misplaced strings are twice as difficult to handle as unplaced strings and it is five times more time-consuming to place an unplaced string than to verify the placement of a string in a cluster.⁵ With these assumptions we compute the following cost for the manual steps after approximate word matching has been employed to cluster the data at a threshold of 0.2.

$$\begin{aligned} \text{cost}_{\text{AWM}(0.20)} &= 38 \cdot c_{\text{setup}} + 10,855 \cdot c_e + 75 \cdot c_m \\ &\quad + 2318 \cdot c_{\text{merge}} + 3029 \cdot c_u \\ &= 38 \cdot c_{\text{setup}} + 10,855 \cdot \beta c_u + 75 \cdot \alpha c_u \\ &\quad + 2318 \cdot c_u + 3029 \cdot c_u \\ &= 38 \cdot c_{\text{setup}} + 7668 \cdot c_u \end{aligned} \quad (3)$$

⁵ Our choice of α and β reflects our estimation of the difficulty of the task as we have outlined it. The choice of values for these parameters may be influenced by many factors, for example, the tools available, the number of strings being clustered, and the number of clusters desired.

⁴ Balanced Iterative Reducing and Clustering Using Hierarchies.

Different degrees of clustering will produce different numbers of clusters and different numbers of errors. For example, for threshold 0.05 we have 7789 clusters (5173 containing only one string), threshold 0.10 produces 6875 clusters (4301 one-string), threshold 0.15 produces 6128 clusters (3662 one-string), threshold 0.25 produces 4561 clusters (2445 one-string), and threshold 0.30 produces 3740 clusters (1870 one-string). These, plus the errors made lead to the following costs:

$$\begin{aligned}
\text{cost}_{\text{AWM}(0.05)} &= 38 \cdot c_{\text{setup}} + 8211 \cdot c_e + 0 \cdot c_m \\
&\quad + 2616 \cdot c_{\text{merge}} + 5173 \cdot c_u \\
&= 38 \cdot c_{\text{setup}} + 8211 \cdot \beta c_u + 2616 \cdot c_u \\
&\quad + 5173 \cdot c_u \\
&= 38 \cdot c_{\text{setup}} + 9431.2 \cdot c_u \\
\text{cost}_{\text{AWM}(0.10)} &= 38 \cdot c_{\text{setup}} + 9583 \cdot c_e + 1 \cdot c_m \\
&\quad + 2574 \cdot c_{\text{merge}} + 4301 \cdot c_u \\
&= 38 \cdot c_{\text{setup}} + 9583 \cdot \beta c_u + 1 \cdot \alpha c_u \\
&\quad + 2574 \cdot c_u + 4301 \cdot c_u \\
&= 38 \cdot c_{\text{setup}} + 8793.6 \cdot c_u \\
\text{cost}_{\text{AWM}(0.15)} &= 38 \cdot c_{\text{setup}} + 10,222 \cdot c_e + 13 \cdot c_m \\
&\quad + 2466 \cdot c_{\text{merge}} + 3662 \cdot c_u \\
&= 38 \cdot c_{\text{setup}} + 10,222 \cdot \beta c_u + 13 \cdot \alpha c_u \\
&\quad + 2466 \cdot c_u + 3662 \cdot c_u \\
&= 38 \cdot c_{\text{setup}} + 8198.4 \cdot c_u \\
\text{cost}_{\text{AWM}(0.25)} &= 38 \cdot c_{\text{setup}} + 11,439 \cdot c_e + 439 \cdot c_m \\
&\quad + 2116 \cdot c_{\text{merge}} + 2445 \cdot c_u \\
&= 38 \cdot c_{\text{setup}} + 11,439 \cdot \beta c_u + 439 \cdot \alpha c_u \\
&\quad + 2116 \cdot c_u + 2445 \cdot c_u \\
&= 38 \cdot c_{\text{setup}} + 7726.8 \cdot c_u \\
\text{cost}_{\text{AWM}(0.30)} &= 38 \cdot c_{\text{setup}} + 12,014 \cdot c_e + 1463 \cdot c_m \\
&\quad + 1870 \cdot c_{\text{merge}} + 1870 \cdot c_u \\
&= 38 \cdot c_{\text{setup}} + 12,014 \cdot \beta c_u + 1463 \cdot \alpha c_u \\
&\quad + 1870 \cdot c_u + 1870 \cdot c_u \\
&= 38 \cdot c_{\text{setup}} + 9068.8 \cdot c_u \tag{4}
\end{aligned}$$

The setup costs are the same and small relative to the other costs, so for approximate word matching we have the following.

$$\begin{aligned}
\text{cost}_{\text{AWM}(0.05)} &= 0.68 \cdot \text{cost}_{\text{manual}} \\
\text{cost}_{\text{AWM}(0.10)} &= 0.63 \cdot \text{cost}_{\text{manual}} \\
\text{cost}_{\text{AWM}(0.15)} &= 0.59 \cdot \text{cost}_{\text{manual}} \\
\text{cost}_{\text{AWM}(0.20)} &= 0.55 \cdot \text{cost}_{\text{manual}} \\
\text{cost}_{\text{AWM}(0.25)} &= 0.56 \cdot \text{cost}_{\text{manual}} \\
\text{cost}_{\text{AWM}(0.30)} &= 0.65 \cdot \text{cost}_{\text{manual}} \tag{5}
\end{aligned}$$

The cost is lowest for thresholds 0.20 and 0.25 and is just a little over half the effort of a strictly manual approach.

We have shown the expanded calculations because inspection of the coefficients of each term helps explain which factors are improving and which are worsening as the threshold is varied. The dominant factor is the number of misplaced strings as the threshold is increased. This forces more human effort to clean up the clusters. As can be seen, for this set of assumptions the human effort required after clustering is about half that when clustering is not employed.

It should be noted that two factors are not captured by this cost analysis. First, the initial development of heuristics is not counted. This cost may be substantial, but we assume that it is not repeated after the development is complete and can, therefore, be amortized over all subsequent uses. Second, we do not account for iteration. As we have noted, this is an iterative process. One might try several thresholds, note the error rate and rerun with a tighter or looser threshold setting. We have not factored this into the analysis and have assumed that fixed threshold values would be employed. Nevertheless, the analysis is useful in demonstrating where the major costs are incurred. It also demonstrates that cost savings are achievable.

8. Conclusions

There are many opportunities to exploit existing online databases as new techniques are developed in the field of information retrieval. Before this evolution can take place, we need to find ways to improve the quality of the data and to make it easier for data providers to integrate new information into their systems. This paper has discussed techniques for improving data quality and data access by detecting variable forms of strings and collecting them together under a standard form. This can be thought of as the semi-automatic generation of an authority file. After we generate authority files, we can use the information to reduce the burden of subsequent data entry by automating the detection of variant forms as new data is acquired by a system.

We have evaluated some approximate string matching approaches for clustering variant forms of strings. We have also proposed a new way to evaluate string distance—approximate word matching—and shown how it can im-

prove string clustering in this application. Finally, we have given a method for speeding up the evaluation of edit distance in our one-pass algorithm when a threshold is being employed for clustering.

The techniques described here can be used very effectively for automating the construction of authority files. For the application discussed here, the human effort involved can be reduced by half. However, there are still situations that require manual intervention to resolve, for example, when two completely dissimilar strings denote the same entity. Further improvement in the performance of these techniques can only be achieved by the introduction of specific domain knowledge into the process. In this way, we can eliminate the manual intervention once the domain knowledge has been captured.

Acknowledgment

This work was supported in part by Department of Energy Grant no. DE-FG05-95ER25254, NSF grant CDA-9529253, DARPA contract N66001-97-C-8542, and NASA Graduate Student Researchers Program fellowship NGT5-50062. This work was performed while Eric Schulman was at the National Radio Astronomy Observatory in Charlottesville, VA. The National Radio Astronomy Observatory is a facility of the National Science Foundation operated under cooperative agreement by Associated Universities, Inc. We would like to thank Stephen S. Murray, Guenther Eichhorn, and Michael J. Kurtz of the Smithsonian Astrophysical Observatory for providing us access to the data in the Astrophysics Data System (ADS). We would also like to acknowledge the thoughtful comments of the reviewers.

References

Abt, H.A. (1993). Institutional productivities. *Publications of the Astronomical Society of the Pacific*, 105, 794–798.

Accomazzi, A., Eichhorn, G., Kurtz, M.J., Grant, C.S., & Murray, S.S. (1997). Astronomical information discovery and access: Design and implementation of the ADS bibliographic services. In *Astronomical Data Analysis Software and Systems VI*, Vol. 125 of the *Astronomical Society of the Pacific Conference Series* (pp. 357–360).

Auld, L. (1982). Authority control: An eight-year review. *Library Resources & Technical Services*, 26, 319–330.

Borgman, C.L., & Siegfried, S.L. (1992). Getty's synonyme and its cousins: A survey of applications of personal name-matching algorithms. *Journal of the American Society for Information Science*, 43, 459–476.

Damerau, F.J. (1964). A technique for computer detection and correction of spelling errors. *Communications of the ACM*, 7, 171–176.

French, J.C., Powell, A.L., & Schulman, E. (1997a). Applications of approximate word matching in information retrieval. In *6th International Conference on Information and Knowledge Management (CIKM'97)* (pp. 9–15). Las Vegas, Nevada.

French, J.C., Powell, A.L., Schulman, E., & Pfaltz, J.L. (1997b). Automating the construction of authority files in digital libraries: A case study. In C. Peters & C. Thanos (Eds.), *First European Conference on Research and Advanced Technology for Digital Libraries*, Vol. 1324 of *Lecture Notes in Computer Science* (pp. 55–71). Pisa: Springer-Verlag.

Ganti, V., Ramakrishnan, R., Gehrke, J., Powell, A., & French, J. (1999). Clustering large datasets in arbitrary metric spaces. In *15th International Conference on Data Engineering (ICDE'99)* (pp. 502–511). Sydney.

Gusfield, D. (1997). *Algorithms on strings, trees, and sequences*. Cambridge University Press.

Hall, P.A.V., & Dowling, G.R. (1980). Approximate string matching. *Computing Surveys*, 12, 381–402.

Jain, A.K., & Dubes, R.C. (1988). *Algorithms for clustering data*. Prentice Hall.

Kukich, K. (1992). Techniques for automatically correcting words in text. *Computing Surveys*, 24, 377–440.

Lowrance, R., & Wagner, R.A. (1975). An extension of the string-to-string correction problem. *Journal of the ACM*, 22, 177–183.

Morgan, H.L. (1970). Spelling correction in systems programs. *Communications of the ACM*, 13, 90–94.

O'Neill, E.T., & Vizin-Goetz, D. (1988). Quality control in online databases. *Annual Review of Information Science and Technology*, 23, 125–156.

Schulman, E., Powell, A.L., French, J.C., Eichhorn, G., Kurtz, M.J., & Murray, S.S. (1996). Using the ADS database to study trends in astronomical publication. *Bulletin of the American Astronomical Society*, 28, 1281.

Schulman, E., French, J.C., Powell, A.L., Eichhorn, G., Kurtz, M.J., & Murray, S.S. (1997a). Trends in astronomical publication between 1975 and 1996. *Publications of the Astronomical Society of the Pacific*, 109, 1278–1284.

Schulman, E., French, J.C., Powell, A.L., Murray, S.S., Eichhorn, G., & Kurtz, M.J. (1997b). The sociology of astronomical publication using ADS and ADAMS. In G. Hunt & H. Payne (Eds.), *Astronomical data analysis software and systems VI*, Vol. 125 of the *Astronomical Society of the Pacific Conference Series* (pp. 361–364).

Siegfried, S.L., & Bernstein, J. (1991). Synonyme: The Getty's new approach to pattern matching for personal names. *Computers and the Humanities*, 25, 211–226.

Strong, D.M., Lee, Y.W., & Wang, R.Y. (1997). Data quality in context. *Communications of the ACM*, 40, 103–110.

Taylor, A.G. (1984). Authority files in online catalogs: An investigation of their value. *Cataloging & Classification Quarterly*, 4, 1–17.

Trimble, V. (1984). Postwar growth in the length of astronomical and other scientific papers. *Publications of the Astronomical Society of the Pacific*, 96, 1007–1016.

Wagner, R.A., & Fischer, M.J. (1974). The string-to-string correction problem. *Journal of the ACM*, 21, 168–173.

Williams, M.E., & Lannom, L. (1981). Lack of standardization of the journal title data element in databases. *Journal of the American Society for Information Science*, 32, 229–233.

Zhang, T., Ramakrishnan, R., & Livny, M. (1996). BIRCH: An efficient data clustering method for very large databases. In *Proceedings of the 1996 ACM SIGMOD International Conference on Management of Data* (pp. 103–114). Montreal.

Zobel, J., & Dart, P. (1996). Phonetic string matching: Lessons from information retrieval. In *Proceedings of the 19th International Conference on Research and Development in Information Retrieval (SIGIR'96)* (pp. 166–172).