Enhanced Graph Based Genealogical Record Linkage

Cary Sweet[1], Tansel Özyer[2], and Reda Alhajj[1,3]

[1] Department of Computer Science, University of Calgary, Calgary, Alberta
[2] TOBB Ekonomi ve Teknoloji Üniversitesi, Ankara, Turkey
[3] Department of Computer Science, Global University, Beirut, Lebanon
alhajj@cpsc.ucalgary.ca

Abstract. Record linkage is an essential social problem that has received considerable attention for over two centuries. With the popularity of computers, automated systems started to be realized for better record linkage systems. In this paper, we look at improving the latest techniques in record linkage, from the perspective of genealogy, to obtain results that require decreased human intervention. The increased benefit will be measured against the mainstream string and entity comparison techniques.

1 Introduction

With the Internet being utilized in more homes around the world, information on the web is becoming more accessible. As a result, the new generation of genealogy software, which allows the user to load multiple genealogies, search and merge duplicate individuals, is becoming popular in the current market; most of it is likely to be on-line and be used interactively. However, the complex queries for searching are still restricted to the boundaries defined by the software developer and all have GUI interfaces that run the queries; much of the searching in genealogy has suffered from the limitation of a GUI interface. So, one of the objectives of this research work is to allow users to type in a command that will resemble the English language structure, and return the result from genealogical database. The major reason for developing this system is to give users the added flexibility of potential relationships (cousins, ancestries) in the database.

Family history is generally represented by a genetic family tree (also called a "pedigree"), which shows the past and present members of the family joined together by a series of links that help in ascertaining their relationship to each other, and the location, documentation and recording of a family history. An example of a genetic family tree is shown in Figure 1.

The main idea in using genetic family tree is that it is easy to view and understand by users. By using chart diagram, users can easily track and transfer the relationships including sound, video, and text files - along with a wide variety of graphic formats in the trees. When the multiple records identified reside within the same family tree, a smaller tree with no duplicates will be produced. When

R. Alhajj et al. (Eds.): ADMA 2007, LNAI 4632, pp. 476–487, 2007.

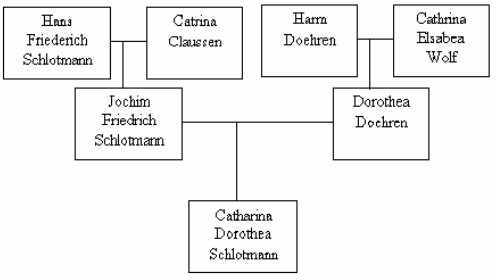


Fig. 1. Example genetic family tree

the identified records reside in two or more disjoint trees, a larger tree will result. This process is also referred to as duplicate reduction or merge/purge.

Graph-matching [19] is an essential technique in genealogical record linkage that takes advantage of the hierarchical nature of genealogical data. In the abstract form, trees represent genealogical data whose nodes are the real world entities and edges are relationships between the entities. A graph-matching algorithm then uses the identified records as focal points and extends to all nodes that overlap between the trees. The record linkage evaluation will be based on the extended coverage of the trees.

A set of potentially matching records goes through a ranking stage. At the record level, attributes of records are compared for corresponding accuracy. The aggregate ranking of a set of identified records is determined by both these attribute matches and the subsequent node matches. Once all the identified records have been sorted by ranking, they are grouped into three categories: Linked, Unlinked and Unknown. Any record that falls into the "Unknown" category will require manual research to determine which category the record finally belongs. The decision of which ranking will act as a cutoff is also made manually.

This process seems to be able to produce effective results. Addressed in this research work are several areas that could be improved upon that would produce a more user-friendly set of final data. In particular, the following issues are handled.

- The algorithms used to evaluate the record attributes (names, dates, etc.)
- The calculation of both the entity and aggregate rankings
- Knowing the structure of the overlapping tree
- Using inferred dates in replacement of missing ones
- A smaller and better-organized "Unknown" category would reduce the amount of time necessary for manual evaluation of the records.

The conducted analysis demonstrate the power of the developed process.

The rest of this paper is organized as follows. Section 2 is a brief overview of the related work. Section 3 highlights the aspects of the developed data miner for handling record linkage information. Section 4 reports on the analysis of the ranking process. Section 5 is summary and conclusions.

2 Related Work

Record linkage is one of the most essential social problems studied in history, and several attempts tried to handle the problem from different perspectives, e.g., [1,3,9,12,16,20] . Heuristics based on matching Soundex codes on names and exact matching on date and place fields for some event, including birth, marriage, or death, were the drivers for the first initiatives in record linkage algorithms. Once a pair of duplicates was linked, additional, less-stringent rules were used to link related family members. The next milestone in record linkage could be identified as statistical based on a Fellegi-Sunter algorithm [7,13]. The weighted features used in the statistical algorithm were based on name and event information for various world regions.

In general, there are two main areas of interest dealing with genealogical databases. They are family reconstruction and the work of projects like those undertaken by the GRANDMA project [8] of the California Mennonite Historical Society. The Minnesota Population Center [15] is Family reconstruction organization that is trying to link the records from 5 different censuses ranging from 1860 through 1910. Considering these crosses over approximately 3 generations, record linkage would be occurring with the new families of the children as well as within the same family.

Taking multiple genealogical databases and merging them together help produce a larger, more complete database. The algorithms utilized in this task have been progressing in their level of complexity. The early attempts involved only comparing two records [11]. Post-processing rules were used to handle merging the relationships of the selected records [14]. Current work in this area by the Family History department involves graph-matching [19,14].

While graph-matching generates accurate results, Wilson's algorithm [19] focused on the resultant tree only as a benefit to the merging process. Wilson's algorithm also focused on finding only high-ranking matches while building the tree. This meant that the content of the tree could not be utilized in determining the ranking of the individuals within the tree. This has been the basic motivation to conduct the work described in this paper.

3 Developing the Data Miner

The DataMiner project involved the creation of a query language that would be used to assist in the determination for candidate objects to be merged within a hierarchical data model. Specifically, a genealogical database usually contains one or more entries of the same person. In this data model, only the individual, parents and spouse are used for comparison and ranking purposes. It covers several stages of the record linkage process.

3.1 Find All the Matching Individuals Within the Database

This involved taking a list of groups where the only grouping criterion was the first and last names had to match. This list was then manually verified as a

possibility match with secondary and tertiary sources [17,4,5,8,18]. This has been tested on a database of 22696 persons. While this procedure identified 362 duplicates and 42 potential duplicates as not a match, it did not guarantee that *all* matches were found. Great pains were made, however, to ensure that most reasonable combinations of the 22696 person database were evaluated.

3.2 Query Language

DataMiner has the capability to describe when a field does not contain null. Previously, when an attribute was placed within the query, the resultant grouping allowed all the nulls to match one another. To help reduce the level of undesirable information, the attribute modifier not_null was added to the query language.

Date(not_null:birth:indiv)
Father_modified_given(not_null)

3.3 Analyze the Ranking Function

It was important to determine the maximum efficiency of the implemented algorithm. This will allow the comparisons with the following enhancements to be better understood. A hill-climbing algorithm was used to drive the search for the best values for each attribute that was used within the ranking function. When analyzing the values for the attributes three interests were being considered:

- The highest level of accuracy for the current algorithm
- List of attributes that had no impact on the outcome of the results
- List of attributes that had the highest impact on the outcome

Attributes of the ranking function. Three relationships were exploited and the attributes of each person involved were combined to reach a single ranking value. The relationships were: Current individual, Spouse, Mother and Father. For each of these the aggregate attributes of name, birth date, baptismal date and death date were selected. In total, there are 98 attributes (23 for each individual, plus the 6 for only the spouse, mother and father). New attributes were added in to determine how they would impact the results. These included: Entity Comparison results in zero ranking; The individual does not exist; Sex; Location; First and last name both match; First and last name both do not match; Year, month and day all match; and Fuzzy logic date matching.

It was previously assumed that Sex and Location would have zero impact on the results and were removed, for this reason, from the ranking function. Since a full analysis of the ranking function was identified essential, the basic attributes of Sex and Location were added to ensure completeness.

Each attribute is counted only once. For instance, if year, month and day all match, only the full date match attribute is applied to the ranking function. The individual component evaluations are ignored in this case.

Individual level attributes: These attributes are applied to only the spouse, mother and father. The individual is assumed to always be involved in the equation and thus exists with at least a partial match of the name. This assumption

makes these attributes unnecessary for the ranking of the primary individual of interest.

Comparison results in a rank of zero. None of the following attributes match between the individuals of interest. This is useful when a person has married more than once. The first individual could represent the spouse of the first marriage, while the second individual could represent the individual of the second marriage.

The individual does not exist. If an individual were missing both parents, this situation would be more probable for a match than if both parents existed and had a very low individual level ranking.

Name and sex attributes

First name and last name. These are the basic constructs for comparing an individual. These rankings only apply if only one and both attributes are matched.

Comparison results in a rank of zero. Both the first and last names did not match. This attribute was added to see what the hill-climbing algorithm would do with it.

Both match. This attribute gives the capability of enhancing the ranking for a better than partial match.

Sex. Some names are unisexual, such as Kim. This attribute may play a role in obtaining better results. This attribute was added to test its significance within the ranking function. The hill-climbing algorithm was designed with this in mind.

Birth, baptismal and death events

Location. Most of the data does not have locations and some have mixed level of detail for the same location: Swift Current, Canada; Swift Current, Saskatchewan. This attribute will help test some of the string comparison functions.

Year, month and day. This set of attributes are processed the same as the name attributes. Their ranking is applied only if some, but not all, of the attributes match.

Year, month and day all match. This attribute gives the capability of enhancing the ranking for a better than partial match.

Fuzzy logic matching. While looking at the data it was determined that there were several focal points surrounding the perfect date match. There were peaks at 0, 7, 14, and 21 days as the absolute difference between two entities that were determined by research to be an exact match.

Calculation of the ranking function. The ranking function is created in several phases. First, each of the individuals - primary individual, spouse, mother and father - are evaluated separately by adding the attribute values together. Each individual is then identified as a percentage of the maximum rank for that

individual. These percentages get one added to them and then are multiplied together. This gives us an overall ranking for the individual.

All the relationships for that individual are then evaluated. Moving recursively along the parents, children and spouse lines, the tree is built up with individual matches. These may be good or bad matches. The only criterion is that both individuals have the same movement (e.g. both have a father). The ranking for the tree is the average of all the individuals within the tree. The final ranking for the individual is obtained by finding out what percentage of the maximum individual ranking the current individual and multiplying that to the tree ranking (e.g. 45 % $\times$ 450).

Hill-climbing Algorithm. The hill-climbing algorithm looks at marginal changes to the original ranking and decides which values to keep for all the different attributes. Each attribute is evaluated over a range of values (0 thru 128). The rank that achieves the highest level of accuracy is kept and the next attribute is evaluated. Once all the attributes have been evaluated, the algorithm continues with the first one. This is repeated until the level of accuracy at the end of evaluating the 98^{th} attribute remains the same.

Evaluating the points 0 thru 128 results in either a convex, concave or linear curve. Since all of these only have at most one maximum point, a more optimized algorithm can be employed by using a bisection line search algorithm. Using this algorithm results in an average of 5 evaluations instead of the original 129.

Determining the accuracy of the current algorithm. Before any modification could be made to the algorithms, a baseline level of accuracy had to be defined.

When the program is run, a list of groups is obtained. Each group of two individuals is then ranked using the ranking function and the set of ranks configured by the user, or the hill-climbing algorithm. This list of groups will then be sorted with the highest ranking being first in the list. This sorted list was then compared against the list of known matches. This comparison had to consider three things:

1. The first group of 2 should have a higher score than the last group of 2. This will imply that an incorrect mismatch occurring at the highest likely duplicate will have a more significant impact than the least likely duplicate.
2. A score of 100% should be theoretically attainable
3. The scoring mechanism should be consistent across differing data sets

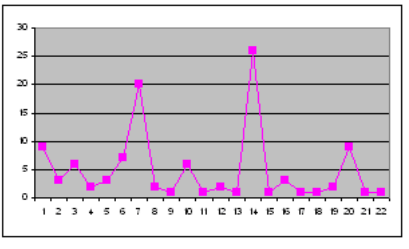
The algorithm decided upon has the following attributes:

- The first group has a rank of n. Where n is the total number of matched records, which were manually identified earlier. In this case: 362
- The last group has a rank of 1
- The total for all the groups is $\frac{n \times (n+1)}{2}$
- For each group that exists within the matched record list, add that group's rank to the total

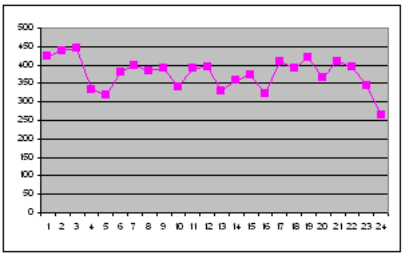
– Only evaluate the first n groups
– Divide the final total by the maximum total to obtain a percentage of accuracy

3.4 Date Comparisons

People who are entering dates into data entry system can always make mistakes. Another discrepancy in dates could arise when changing from the Julian to the Gregorian calendars. This discrepancy could be anywhere from 10 to 13 days depending on the country that the data originated from.



(a) Differences within matching data



(b) Differences within non-matching data of almost the same number of records

Fig. 2. Differences within matching and non-matching

Figure 2(a) shows the results of differences between the dates on the records that were manually identified as matches (see Section 3.1). Several peaks can be identified, the tallest peaks occurring at a difference of 7 and 14 and lower peaks at 1 and 20. Looking at the unmatched data (Figure 2(b)), we can see that no identifiable peaks exist. For this set of data, another unique characteristic of the data can be used as a focus for the ranking function. Under normal date comparisons, these differences could not be utilized. Fuzzy logic along multiple user-defined focus points helps solve these issues. There are three aspects for each focal point: 1) Offset from the date of interest (e.g. 14 days); 2) Range that the fuzzy logic will be applied (e.g. +/-3 days); 3) Type of curve that will be used: concave, convex, or linear.

Table 1. Example focal points identified from Figure 2(a)

Offset	*Range*	*Curve*
7	*2*	*Concave*
14	*2*	*Concave*
20	*2*	*Concave*

3.5 Entity Comparisons

Entity comparisons always involve weighting the individual evaluations and obtaining an aggregate ranking value. Part of this project is to develop an automated evaluation of the different attributes in the data and obtain ideal weightings for each of the attributes. In this way, we can evaluate the importance of a death date over a birth date. An important factor for this analysis will be the availability of the attribute within the data.

Entity comparisons will utilize the graph-matching algorithm [14]. Modifications to this algorithm will involve Identifying potential matches within the main database, Fully evaluating all the nodes within the overlapping sub tree, and Using the final tree ranking to assist in the individual ranking.

Table 2. Highest and lowest Standard Deviations of the attributes

Standard Deviation	***Preferred Value***	***Attribute***
8.27	*0*	*Father No name given*
7.26	*0*	*Father Last Name*
5.89	*0*	*Mother No name given*
2.68	*1*	*Individual Birth Month*
1.97	*4*	*Mother First Name*
1.76	*0*	*Father sex*
1.62	*0 - 15*	*Father Birth Location*
1.48	*14*	*Mother Last Name*
1.45	*14*	*Father Both first and last name*
1.43	*0*	*Spouse No name given*
1.36	*0*	*Father Birth Month*
1.20	*1*	*Mother Sex*
1.06	*6*	*Individual Birth Year*
0	*0 – 99*	*All information regarding Baptism and Death for all individuals*
0	*0 – 99*	*The overall rank of the Mother when zero attributes match*
0	*0 – 99*	*Individual Both first name and last name match*
0	*0 – 99*	*Spouse Birth Location*

3.6 Query Analysis

The blocking technique allows the user to define the minimum requirement of how records will be grouped together for further evaluation and ranking. Sometimes the user will create a blocking definition that creates groups that are larger than necessary (10 or more records within a single group). Evaluating the group could result in suggestions being made to the user as to how to make a more restrictive definition.

By looking at what attributes would match, a histogram can be created that would say which attributes have a higher success ratio and which ones do not.

4 Analysis of the Ranking Function

The highest percentage of accuracy achieved was 96.28%. The standard deviation was used to determine the effect that different values of rank had on an attribute. Attributes that had standard deviations greater than one were considered highly influential in the level of accuracy. These attributes fell into two categories: Those that affected the rank and those that did not. According to Table 2, half of the attributes had a preferred ranking value of zero. Any value higher than this adversely affected the level of accuracy for the ranking function. Attributes that had a standard deviation equal to zero were determined to have no impact on the level of accuracy. The effect of these attributes was felt more in the ranking function, where the more attributes matched the higher the probability of a successful match. It is interesting to note that the highest ranking exists when

Table 3. Attributes with the highest preferred value

Preferred Value	***Attribute***
99	*Father not exist*
0 - 99	*Mother zero rank*
98	*Mother not exist*
81 – 94	*Spouse not exist*
82	*Mother First Name*
0 – 99	*Individual both first and last name match*
97	*Individual birth date all matches*
94	*Spouse both first and last name match*
0 – 99	*All information regarding Baptism and Death for all individuals*

both the mother and father do not exist and the spouse does. This is one of the tree structures that might prove to have a higher possibility of a match.

4.1 Further Analysis

The final accuracy of the tree-ranking algorithm with fuzzy dates was 96.56%. The final ranking for the non-tree algorithm was 94.12%. The main improvement in the final ranking was due to adding in newly found matches by following the graph-matching algorithm. Table 4 shows that keeping all the attributes at the same ranking value, the resulting accuracy is quite diminished compared to the results obtained from before this project began. The ranking function analysis, however, was able to converge upon an ideal ranking function more easily with the new algorithm. This would seem to imply that there are more optimal ranking functions available by using the new algorithm.

Some attributes should never be used in ranking the data. Father's last name, which should always match the primary individual, has a very adverse effect on the level of accuracy when used. Some attributes do not add value to the matching process. While the hill-climbing algorithm says that the information

Table 4. Attributes with the highest matches

Initial Accuracy	***Iter.s before Convergence***	***Matches Found***	***Algorithm***	***Fuzzy Dates***
9.55	*417*	*303*	*Tree Algorithm focus on tree average*	*Yes*
9.57	*421*	*300*	*Tree Algorithm focus on tree average*	*No*
2.77	*916*	*303*	*Tree Algorithm focus on tree size*	*Yes*
91.55	*955*	*258*	*Non-tree algorithm*	*Yes*
91.65	*1017*	*254*	*Non-tree algorithm*	*No*

about the baptism and death events can be ignored by the ranking function, they are still important to help determine which groups are more likely to be a real match.

Subjectively, we would think that the sex of an individual should have no bearing on the outcome. For out primary individual the sex attribute had a standard deviation of 0.26 and a preferred value of three. Odd results like this only mean that the data entered always has a chance of being incorrect. The source of the inaccuracy could be the result of the person recording the information in the registry, the person transcribing the registry or even the person reading the transcription and entering the data into the computer. It is because of these possible sources for errors that any algorithm for genealogical record linkage should be flexible and unrestricted.

In our quest for producing results that are useful for the user, inaccuracies can produce unwanted results. These unpredictable results need to be taken into consideration when creating new algorithms.

The fuzzy dates did not appear to enhance the accuracy of the query process. It might work better with data that focuses on a specific time and/or location that a change in calendar occurred.

This proved to be quite effective when the ranking function was not biased on the size of the tree. At one point, the tree's rank was the sum of the entities in the tree. This produced less then desirable results. Once the ranking focused on the average of the tree's entities, the hill-climbing algorithm was able to converge on an optimal solution more quickly. This would imply that there were more optimal solutions for this new approach to the tree's ranking. On small datasets, the hill-climbing algorithm reached an optimal ranking with only one pass through the data. As the datasets became larger, the number of passes through the data also became larger.

The focus has been on how a single query could be used to produce the best possible results. However, the data being considered might be too much for the computer/user to absorb. Being able to reduce the amount of data returned and only look at higher quality data can be advantageous. The best way to do this would be adding an additional attribute to the query. Having a list of attributes and how they could affect the outcome is very handy.

Table 5. The larger the number of groups(# of grps), the more iterations required before convergence(Iter.s Before Conv.)

# of pos. grps found out of 362	*# of grps evaluated*	*# of attribs used out of 98*	*Iter.s Before Conv.*	*Query*
254	*267*	*49*	*417*	*modified_given, modified_surname, Date(not_null:birth:indiv), MaxGroups(100000)*
279	*3446*	*73*	*916*	*modified_given, modified_surname, Date(birth:indiv), MaxGroups(100000)*
354	*256943*	*95*	*13212*	*modified_given, modified_surname, MinRank(5000), MaxGroups(100000)*

Table 5 shows that the more initial data that the grouping query obtains, the longer it takes to obtain a satisfactory result for the ranking function analysis.

5 Summary and Conclusions

This paper discussed the importance of using graph matching in handling record linkage. As a result of this study, it was found that using a single query over a large number of data produces good results. However, we found out that we need to investigate the accuracy of combining the results of multiple queries running against smaller cross sections of the data. It would be interesting to find out if combining the results of one phone and one string comparison algorithm would produce a better result. Strings to be compared include: First name, last name and location. We also decided to investigate the effect of adding inferred dates when some, but not all the dates are missing within the tree of individual being ranked. Focusing on the tree structure can help arrange the unknown category of records by using probabilistic reasoning. Two individuals have a high match. The parents, siblings, spouses along the tree structure originating at these individuals also produce a high probability of a match. It would then make sense to keep these matches close together. Comparisons could be made based on spatial closeness.

References

1. Bengtsson, T., Lundh, C.: Name-standardization and automatic family reconstitution. Lund Papers in Economic History. Department of Economic History, Lund University 29, 1-24 (1993)
2. Bloothooft, G.: Multi-Source Family Reconstruction. History and Computing 7(2), 90–103 (1995)

3. Bouchard, G.: Current issues and new prospects for computerized record linkage in the province of Québec. Historical Methods 25, 67–73 (1992)
4. Dyck, J., William, H.: Reinlaender Gemeinde Buch (1994)
5. Ens, A., Jacob, E.P., Otto, H.: 1880 Village Census of the Mennonite West Reserve (1998)
6. Family History Department, CJCLS (2002) `http://www.familysearch.org/Eng/Home/FAQ/faq_gedcom.asp`
7. Fellegi, I.P., Sunter, A.B.: A Theory for Record Linkage. Journal of the American Statistical Association 64, 1183–1210 (1969)
8. GRANDMA, Genealogical Project Committee of the California Mennonite Historical Society, vol. 3 (2000)
9. Jaro, M.A.: Advances in Record-Linkage Methodology as Applied to Matching the 1985 Census of Tampa, Florida. Journal of the American Statistical Association 89, 414–420 (1989)
10. Katz, M., Tiller, J.: Record-linkage for everyman: A semi-automated process. Historical Methods Newsletter 5, 144–150 (1972)
11. NeSmith, N.P.: Record Linkage and Genealogical Files. Record Linkage Techniques, pp. 358–361. National Academy Press, Washington, DC (1997)
12. Nygaard, L.: Name standardization in record linking: An improved algorithmic strategy. History & Computing 4, 63–74 (1992)
13. Newcombe, H.B., Kennedy, J.M., Axford, S.J., James, A.P.: Automatic Linkage of Vital Records. Science 130, 954–959 (1959)
14. Quass, D., Paul, S.: Record Linkage for Genealogical Databases. In: ACM SIGKDD 2003 Workshop on Data Cleaning, Record Linkage, and Object Consolidation, August 2003 (2003)
15. Ruggles, S.: Linking Historical Censuses: A New Approach. In: IMAG Workshop (2003)
16. Schofield, R.: Automatic family reconstitution - the Cambridge experience. Historical Methods 25, 75–79 (1992)
17. Unger, H., Martha, M., Ens, A.: Sommerfelder Gemeinde Buch (2004)
18. Unpublished genealogical records Swift Current Gemeinde Buch. vol. 1, 2
19. Wilson, R.: Graph-Based Remerging of Genealogical Databases. In: Provo UT. Proceedings of the 2001 Family History Technology Workshop, pp. 4–6 (2001)
20. Winkler, W.E.: Advanced Methods of Record Linkage. American Statistical Association. In: Proceedings of the Section of Survey Research Methods, pp. 467–472 (1994)
21. Winchester, I.: The linkage of historical records by man and computer: Techniques and problems. Journal of Interdisciplinary History 1, 107–124 (1970)