5-5-2015

RLT-S: A Web System for Record Linkage

Robert H. Aseltine
University of Connecticut School of Medicine and Dentistry

Sanguthevar Rajasekaran
University of Connecticut - Storrs

Abdullah-Al Mamun
University of Connecticut, Storrs

RESEARCH ARTICLE

RLT-S: A Web System for Record Linkage

Abdullah-Al Mamun[1], **Robert Aseltine**[2], **Sanguthevar Rajasekaran**[1]*

1 Department of Computer Science and Engineering, University of Connecticut, Storrs, Connecticut, United States of America, **2** Institute for Public Health Research, University of Connecticut, East Hartford, Connecticut, United States of America

* rajasek@engr.uconn.edu

Abstract

Background

Record linkage integrates records across multiple related data sources identifying duplicates and accounting for possible errors. Real life applications require efficient algorithms to merge these voluminous data sources to find out all records belonging to same individuals. Our recently devised highly efficient record linkage algorithms provide best-known solutions to this challenging problem.

Method

We have developed RLT-S, a freely available web tool, which implements our single linkage clustering algorithm for record linkage. This tool requires input data sets and a small set of configuration settings about these files to work efficiently. RLT-S employs exact match clustering, blocking on a specified attribute and single linkage based hierarchical clustering among these blocks.

Results

RLT-S is an implementation package of our sequential record linkage algorithm. It outperforms previous best-known implementations by a large margin. The tool is at least two times faster for any dataset than the previous best-known tools.

Conclusions

RLT-S tool implements our record linkage algorithm that outperforms previous best-known algorithms in this area. This website also contains necessary information such as instructions, submission history, feedback, publications and some other sections to facilitate the usage of the tool.

Availability

RLT-S is integrated into http://www.rlatools.com, which is currently serving this tool only. The tool is freely available and can be used without login. All data files used in this paper have been stored in https://github.com/abdullah009/DataRLATools. For copies of the relevant programs please see https://github.com/abdullah009/RLATools.

OPEN ACCESS

Citation: Mamun A-A, Aseltine R, Rajasekaran S (2015) RLT-S: A Web System for Record Linkage. PLoS ONE 10(5): e0124449. doi:10.1371/journal.pone.0124449

Academic Editor: Hans A Kestler, Leibniz Institute for Age Research, GERMANY

Received: September 22, 2014

Accepted: March 13, 2015

Published: May 5, 2015

Data Availability Statement: All data are available through Figshare.com. Dataset doi: http://dx.doi.org/10.6084/m9.figshare.1340114. Source code doi: http://dx.doi.org/10.6084/m9.figshare.1340113.

Funding: This work has been supported in part by the following grants: NIH-R01-LM010101 and NSF-1447711. No other funding has been used for this research.

Competing Interests: The authors have declared that no competing interests exist.

Introduction

Record linkage has evolved as a crucial problem in many areas of science and engineering. A large number of health agencies store medical records of patients [1, 2]. Finding data of an individual across these sources requires efficient algorithms. Record linkage has also applications in disease evolution [3, 4], master data management, copy detection in digital documents [5, 6], historical data management, and so on.

Record linkage collects records of same individuals from multiple data sources possibly having some corrupted records due to typo, phonetic similarity, etc. Now-a-days data of an individual reside across multiple databases and at the same time data agencies keep records of millions of people. Accuracy as well as time efficiency in finding all the records of an individual make the problem challenging. A naive algorithm compares each pair of records and measures similarities. This method is very time consuming. Many algorithms have been devised to improve this naive algorithm [7–11]. We have already proposed efficient and effective sequential and parallel record linkage algorithms [12], which outperform previous best-known record linkage algorithms [13]. Our methods use single linkage hierarchical clustering which generates a dendrogram. By applying a threshold value on this dendrogram we get our expected clusters for individuals.

A large number of record linkage tools are widely available. Java-based fine-grained probabilistic record integration and linkage tool (FRIL) is an open source tool, which has support for parameters configuration and can handle millions of records [14, 15]. Another widely used record linkage tool is FEBRL (Freely Extensible Biomedical Record Linkage) which performs data standardization as well as probabilistic record linkage of one or more files [16].

In this paper we present details on our record linkage tool, RLT-S, which implements the record linkage algorithm based on single linkage clustering of [12]. This tool is freely available in www.rlatools.com website. The website also provides proper instructions, submission history and some other necessary features to ease the usage of the tool. The tool generates a well-formatted output to facilitate user perception. In this paper we describe the functionalities of the tool as well as necessary parameters for input handling, linkage processing, and generation of output.

Implementation

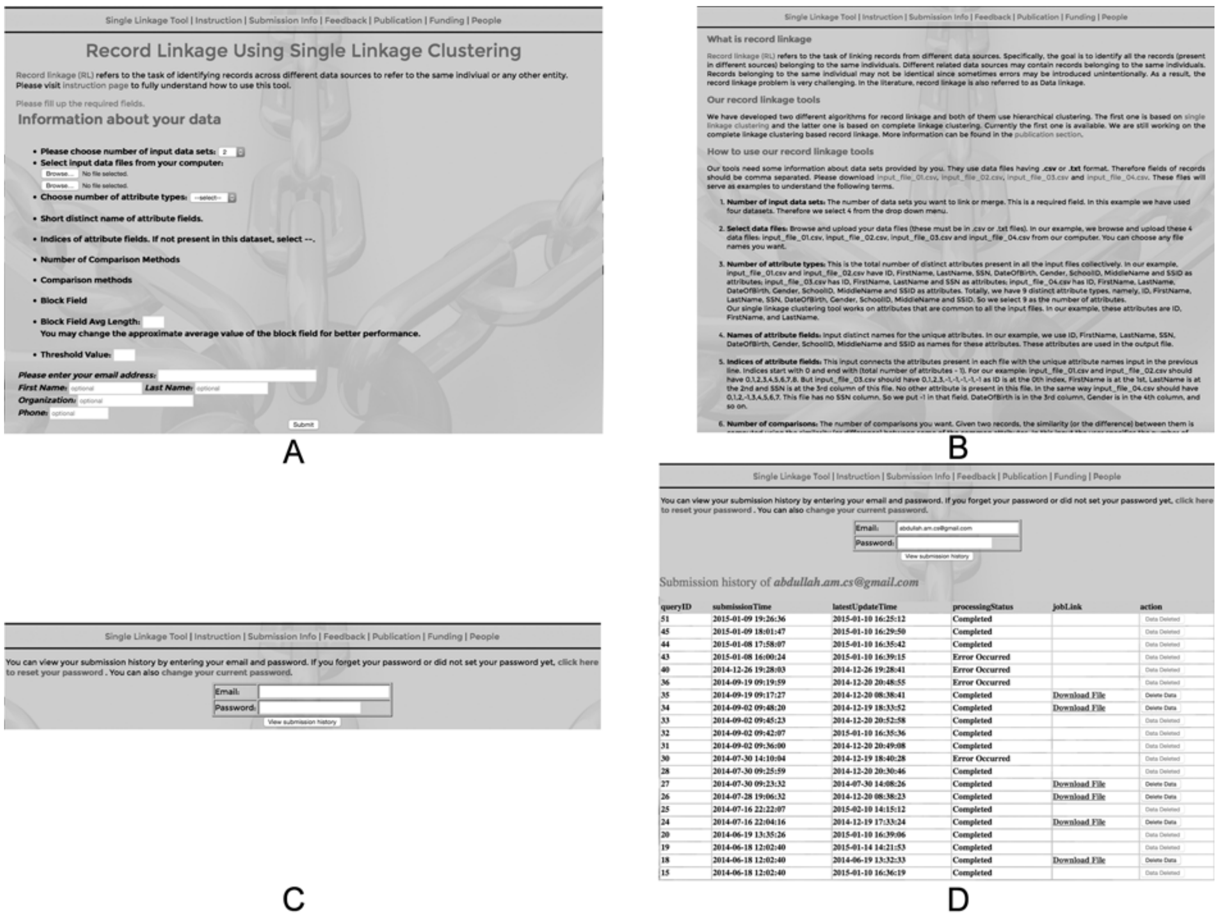
RLT-S is a Java implementation of sequential RLA (Record Linkage Algorithm) [12]. This algorithm clusters records of individuals using single linkage hierarchical clustering. It merges records from all the data sets as if they were from one data set. Therefore the performance of the algorithm is independent of the number of input data sets. It sorts records on common attribute fields using radix sort. Sorting helps us to separate duplicate records, which indicates exact matching. Real life applications do not contain much error. Therefore the sorting phase reduces the size of the unified data set into a smaller data set with no exact duplicates of records. Next phases work on only the representative records from all the exact match clusters. We call the first record of each cluster as the representative record of that cluster. Our RLA employs single linkage hierarchical clustering. Comparison between each pair of records consumes a lot of time. To facilitate finding groups of similar records it employs blocking on a specified attribute field. It finds linkages among those records in a block. We use l-mers (i.e., substrings of length l) of the attribute field for blocking. Any L-length record will be present in (L—l + 1) blocks. In this way different blocks are connected. We then employ hierarchical clustering with single linkage by measuring distances between pairs of records in a block using any combination of edit distance, reversal distance, and truncation distance methods. If we consider each record as a vertex in a graph and linkage as an edge between two vertices, then we get our desired graph. We remove multiple edges and self-loops from this graph. Each connected component of this graph is a cluster of records of an individual. The algorithm outputs these clusters and all their identical records generated by exact matching as final clusters.

The website and the associated tool perform record linkage among one or more data files. We have simplified the usage of the tool by minimizing queries and text input. Whenever possible we have provided drop-down lists to select possible values. Generated output is also well formatted so that the users can easily identify records of an individual.

Results and Discussion

http://www.rlatools.com hosts RLT-S tool and provides all the other necessary features to ensure the best functioning of the tool. Anyone can use this freely available tool with or without login information. Registered users have the facility to view all of their submissions information and outputs. But without login also users can use the tool and find their outputs using links via email. Currently we keep all the output files in our server so that the users can view and download them at any time. Fig 1 shows a diagram of the website pipeline.

Our tool requires some specific information. Some of these are required, some are highly recommended and a few of them are optional. We have set default values for some attributes if the users do not want to input them. RLT-S works in three separate phases. The first and the


A
B
C
D
Single Linkage Tool | Instruction | Submission Info | Feedback | Publication | Funding | People
Browse...
No file selected.
Submit
View submission history

Record Linkage Using Single Linkage Clustering

Record linkage (RL) refers to the task of identifying records across different data sources to refer to the same indiviual or any other entity. Please visit instruction page to fully understand how to use this tool.

Please fill up the required fields.

Information about your data

- Please choose number of input data sets: 2
- Select input data files from your computer:
- Choose number of attribute types: --select--
- Short distinct name of attribute fields.
- Indices of attribute fields. If not present in this dataset, select --.
- Number of Comparison Methods
- Comparison methods
- Block Field
- Block Field Avg Length:
  You may change the approximate average value of the block field for better performance.
- Threshold Value:

Please enter your email address:
First Name: optional Last Name: optional
Organization: optional
Phone: optional

What is record linkage

Record linkage (RL) refers to the task of linking records from different data sources. Specifically, the goal is to identify all the records (present in different sources) belonging to the same individuals. Different related data sources may contain records belonging to the same individuals. Records belonging to the same individual may not be identical since sometimes errors may be introduced unintentionally. As a result, the record linkage problem is very challenging. In the literature, record linkage is also referred to as Data linkage.

Our record linkage tools

We have developed two different algorithms for record linkage and both of them use hierarchical clustering. The first one is based on single linkage clustering and the latter one is based on complete linkage clustering. Currently the first one is available. We are still working on the complete linkage clustering based record linkage. More information can be found in the publication section.

How to use our record linkage tools

Our tools need some information about data sets provided by you. They use data files having **.csv** or **.txt** format. Therefore fields of records should be comma separated. Please download input_file_01.csv, input_file_02.csv, input_file_03.csv and input_file_04.csv. These files will serve as examples to understand the following terms.

1. **Number of input data sets:** The number of data sets you want to link or merge. This is a required field. In this example we have used four datasets. Therefore we select 4 from the drop down menu.
2. **Select data files:** Browse and upload your data files (these must be in .csv or .txt files). In our example, we browse and upload these 4 data files: input_file_01.csv, input_file_02.csv, input_file_03.csv and input_file_04.csv from our computer. You can choose any file names you want.
3. **Number of attribute types:** This is the total number of distinct attributes present in all the input files collectively. In our example, input_file_01.csv and input_file_02.csv have ID, FirstName, LastName, SSN, DateOfBirth, Gender, SchoolID, MiddleName and SSID as attributes; input_file_03.csv has ID, FirstName, LastName and SSN as attributes; input_file_04.csv has ID, FirstName, LastName, DateOfBirth, Gender, SchoolID, MiddleName and SSID as attributes. Totally, we have 9 distinct attribute types, namely, ID, FirstName, LastName, SSN, DateOfBirth, Gender, SchoolID, MiddleName and SSID. So we select 9 as the number of attributes.
   Our single linkage clustering tool works on attributes that are common to all the input files. In our example, these attributes are ID, FirstName, and LastName.
4. **Names of attribute fields:** Input distinct names for the unique attributes. In our example, we use ID, FirstName, LastName, SSN, DateOfBirth, Gender, SchoolID, MiddleName and SSID as names for these attributes. These attributes are used in the output file.
5. **Indices of attribute fields:** This input connects the attributes present in each file with the unique attribute names input in the previous line. Indices start with 0 and end with (total number of attributes - 1). For our example: input_file_01.csv and input_file_02.csv should have 0,1,2,3,4,5,6,7,8. But input_file_03.csv should have 0,1,2,3,-1,-1,-1,-1,-1 as ID is at the 0th index, FirstName is at the 1st, LastName is at the 2nd and SSN is at the 3rd column of this file. No other attribute is present in this file. In the same way input_file_04.csv should have 0,1,2,-1,3,4,5,6,7. This file has no SSN column. So we put -1 in that field. DateOfBirth is in the 3rd column, Gender is in the 4th column, and so on.
6. **Number of comparisons:** The number of comparisons you want. Given two records, the similarity (or the difference) between them is

You can view your submission history by entering your email and password. If you forget your password or did not set your password yet, click here to reset your password . You can also change your current password.

Email:
Password:

You can view your submission history by entering your email and password. If you forget your password or did not set your password yet, click here to reset your password . You can also change your current password.

Email: abdullah.am.cs@gmail.com
Password:

Submission history of *abdullah.am.cs@gmail.com*

queryID	submissionTime	latestUpdateTime	processingStatus	jobLink	action
51	2015-01-09 19:26:36	2015-01-10 16:25:12	Completed		Data Deleted
45	2015-01-09 18:01:47	2015-01-10 16:29:50	Completed		Data Deleted
44	2015-01-08 17:58:07	2015-01-10 16:35:42	Completed		Data Deleted
43	2015-01-08 16:00:24	2015-01-10 16:39:15	Error Occurred		Data Deleted
40	2014-12-26 19:28:03	2014-12-26 19:28:41	Error Occurred		Data Deleted
36	2014-09-19 09:19:59	2014-12-20 20:48:55	Error Occurred		Data Deleted
35	2014-09-19 09:17:27	2014-12-20 08:38:41	Completed	Download File	Delete Data
34	2014-09-02 09:48:20	2014-12-19 18:33:52	Completed	Download File	Delete Data
33	2014-09-02 09:45:23	2014-12-20 20:52:58	Completed		Data Deleted
32	2014-09-02 09:42:07	2015-01-10 16:35:36	Completed		Data Deleted
31	2014-09-02 09:36:00	2014-12-20 20:49:08	Completed		Data Deleted
30	2014-07-30 14:10:04	2014-12-19 18:40:28	Error Occurred		Data Deleted
28	2014-07-30 09:25:59	2014-12-20 20:30:46	Completed		Data Deleted
27	2014-07-30 09:23:32	2014-07-30 14:08:26	Completed	Download File	Delete Data
26	2014-07-28 19:06:32	2014-12-20 08:38:23	Completed	Download File	Delete Data
25	2014-07-16 22:22:07	2015-02-10 14:15:12	Completed		Data Deleted
24	2014-07-16 22:04:16	2014-12-19 17:33:24	Completed	Download File	Delete Data
20	2014-06-19 13:35:26	2015-01-10 16:39:06	Completed		Data Deleted
19	2014-06-18 12:02:40	2015-01-14 14:21:53	Completed		Data Deleted
18	2014-06-18 12:02:40	2014-06-19 13:32:33	Completed	Download File	Delete Data
15	2014-06-18 12:02:40	2015-01-10 16:36:19	Completed		Data Deleted

Fig 1. Web-based user interface. (A) shows the first and main page of the website, where users select data files, choose configurations and submit them. (B) is the instruction page. Users can view their submission history through login (C). (D) shows a sample submission history page.

doi:10.1371/journal.pone.0124449.g001

Table 1. Records for 5 people having 9 attributes.

ID	FN	LN	SSN	DoB	G	SchID	MN	SSID
1	Risa	Pierce	133183594	09261990	M	1524	Vesta	0676221410
2	Maile	Kramer	135370878	07261991	F	1526	Lenna	0957261480
3	Kimberly	Battle	141274186	04071982	F	1527	Jacki	0144591609
4	Kamal	Mcclain	148965694	10091991	M	70000	Luisa	0278635088
5	Yvonne	Vaughan	153614228	02061992	F	70003	Basil	0368901550

Each row of the table represents each row of Input01.csv file.

doi:10.1371/journal.pone.0124449.t001

third phases work with the input and the output. The second phase tunes parameters and controls the working process of the tool.

Input data sets and configurations

The tool starts with reading and organizing the input data sets. The number of input data sets is required to browse those data files from the user's computers. It accepts. txt or. csv extension files where the attributes of each record are comma separated and each record is in a separate line. We illustrate the working of RLT-S with an example. This example pertains to Tables 1, 2, and 3.

Table 1 shows records of five people. Each record has ID, FN, LN, SSN, DoB, Gender, SchID, MN, and SSID. Table 2 has also 5 records having ID, First Initial, Last Name and Social Security Number as attributes. We have another 5 records from Table 3, each of which has ID, FirstName, LastName, DateOfBirth, Gender, SchID, MN, and SSID as attributes. These 3 tables have different numbers of attributes. We see that the attribute names are quite different from each other although some of them represent the same type. FN, First Initial and FirstName represent first name of a person. Similarly LN, Last name and LastName are similar types.

Consider the task of integrating these 3 tables using our tool. The first required field in RLT-S is the number of input data sets. In this case we select 3 from drop-down list. Then we have to select input files from our computer. As we see some attribute type names are different although they represent the same attribute, we remove first row from each input file. There are 3 browse fields. We browse our computer and select one file at a time. Input01.csv is added at the first browse field, Input02.csv and Input03.csv are added at the second and third fields, respectively. Next required field is the number of attribute types. In this example we see that 9 unique attribute types are present. So we select 9 from the drop-down list. We have seen that the same type has been represented differently in these files. So we choose 9 suitable names for these attribute types, for example, ID, FirstName, LastName, SSN, DateOfBirth, Gender,

Table 2. Records for 5 people having 4 attributes.

ID	First Initial	Last Name	Social Security Number
1	R	Pierce	133183594
2	M	Kramer	135370878
3	K	Battle	141274186
4	K	Mcclain	148965694
8	L	MUELLER	184498846

Each row of the table represents each row of Input02.csv file.

doi:10.1371/journal.pone.0124449.t002

Table 3. Records for 5 people having 8 attributes.

ID	FirstName	LastName	DateOfBirth	Gender	SchID	MN	SSID
1	RISA	PIERCE	09261990	M	001524	VESTA	0676221410
2	MAILE	KRAMER	07261991	F	001526	LENNA	0957261480
3	KIMBERLY	BATTLE	04071982	F	001527	JACKI	0144591609
5	YVONNE	VAUGHAN	02061992	F	070003	BASIL	0368901550
8	KELSIE	MUELLER	01131992	M	070020	JAKE	7243583370

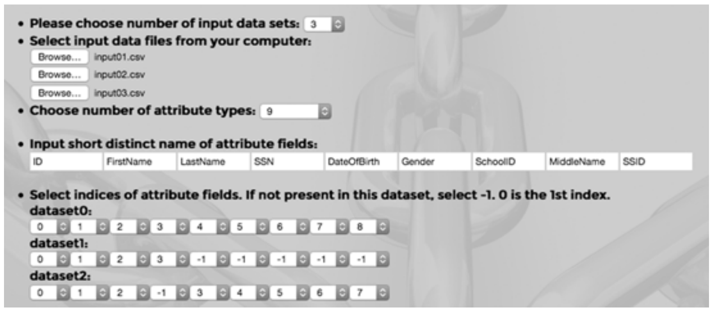
Each row of the table represents each row of Input03.csv file.

doi:10.1371/journal.pone.0124449.t003

SchoolID, MiddleName and SSID. Now we have to connect these names with attribute field names of each data set. We note that ID is the 0th index of 1st input file, FirstName is the 1st, LastName, SSN, DateOfBirth, Gender, SchoolID, MiddleName and SSID are subsequent indices. So we select 0, 1, 2, 3, 4, 5, 6, 7, 8 from the drop-down list for dataset0 or Input01.csv. For dataset1 or Input02.csv, ID is the 0th, FirstName is the 1st, LastName is the 2nd and SSN is the 3rd index. This file has no DateOfBirth, Gender, SchoolID, MiddleName and SSID attribute fields. So we select -1 for each of them. Dataset1 should have 0, 1, 2, 3, -1, -1, -1, -1, -1. Input03.csv has no SSN field. So we put -1 for this index. DateOfBirth is at index 3 of this file, Gender is at 4, and so on. Therefore we select 0, 1, 2, -1, 3, 4, 5, 6, 7 for this dataset. This is the last step of the input phase. Fig 2 shows the above selection of input files.

Linkage parameters

This stage handles the working process of the tool. RLT-S works on common attributes across all the data sets. We need a measure of distance between two attributes (i.e., two strings of characters). RLT-S supports three distance measures. The tool has three types of fields for each comparison. The first field is to select the comparison method (i.e., the distance measure), second one is for selection of the index on which the selected comparison method should be applied and the third type is truncation count, which is the number of characters of the selected attribute that should be used for comparison. We integrate three different comparison methods. The first comparison method is the edit distance calculation. Edit distance or Levenshtein distance measures the minimum number of edit operations required to transform one string to another. Operations include insertion, deletion and substitution. For example, consider the strings A = "computer" and B = "conuterr". If we substitute 'n' to 'm', insert 'p' after this 'm' and delete the last 'r' from B, we get A. Edit distance method needs at least three operations to transform B to A. To use this method, the user has to select an index of the common attribute on



- Please choose number of input data sets: 3
- Select input data files from your computer:
  - input01.csv
  - input02.csv
  - input03.csv
- Choose number of attribute types: 9
- Input short distinct name of attribute fields:

ID	FirstName	LastName	SSN	DateOfBirth	Gender	SchoolID	MiddleName	SSID

- Select indices of attribute fields. If not present in this dataset, select -1. 0 is the 1st index.

dataset0:	0	1	2	3	4	5	6	7	8
dataset1:	0	1	2	3	-1	-1	-1	-1	-1
dataset2:	0	1	2	-1	3	4	5	6	7

Fig 2. Screenshot of input parameter selection for our 3 example files.

doi:10.1371/journal.pone.0124449.g002

which this method will apply. Another distance measure is the reversal distance. Consider a string of two attributes separated by comma, A = "James,Hudson" and another string B = "Hudson,Hames". This method first calculates the edit distance between A and B. We note that a large number of operations are required to transform one string to another. Then it alters the positions of the two strings of B generating B' = "Hames,Hudson" and then measures the distance between A and B'. It finds that only one operation is needed for the transformation. The reversal distance measure is defined as the minimum of the distance between A and B and the distance between A and B'. This method is very useful for the first name and the last name attributes or any other related attributes as users may occasionally input the first name in the last name field and vice versa. Reversal edit distance method needs two attributes to work with. So there are two index fields, each of which should be a unique common attribute index. Truncation distance method is the last method used in our tool. This method is the same as the edit distance method except that it only compares truncation count number of initial characters of both attributes. For example, if A = "James" and B = "J" and if the truncation count is 1, this method calculates the edit distance between A' = "J" and B' = "J". Truncation often occurs for first names as some sources keep only the first name initials. Our single linkage-clustering algorithm reduces a major portion of time for linkage calculation by using blocks on a specific attribute. Each block stores information of similar records. Therefore output accuracy also depends on the choice of the attribute field and its average length. In [12], we have used the last name as the blocking field because the last name was the most appropriate attribute in our collected records. Any other important and reliable common attribute may be the blocking field as well. In most of the cases there is no way to measure a perfect average length of the block field. But an approximate average length makes our job easy. If the user does not fill in this field, the tool uses 7 as the default value. In [12], we show how the value of k affects the blocking performance. In our example we have used 2 comparisons, edit distance calculation on the last name, and truncation distance method on the first name with a truncation count of 1. Our linkage criteria are shown in Fig 3.

Output

The third stage requires some information to generate and send outputs. RLT-S employs hierarchical clustering and generates a dendrogram. A dendrogram represents clusters produced by hierarchical clustering in a tree in a well-organized way. A threshold value is needed to output the desired clusters. This threshold value says how many errors RLT-S endures. To understand fully the threshold value, let us consider another example. One record has James, Rodriguez, and 01011990 for FirstName, LastName and DateOfBirth fields, respectively, and another record has Rodriuez, 123456789, and Hames as LastName, SocialSecurityNumber and FirstName attributes. We see that 'James' has been transformed into 'Hames' which indicates that one substitution is needed to correct it and one character has been deleted from 'Rodriguez'. The matching process finds 2 errors between these two records. If we input a threshold value of 1, then the tool produces 2 clusters, each of which contains only one record. But if the

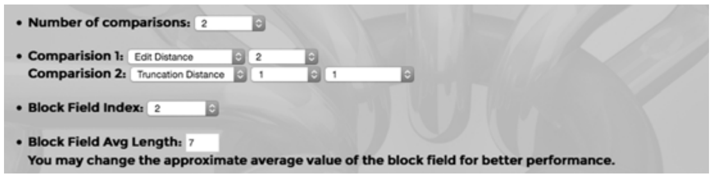

- Number of comparisons: 2
- Comparision 1: Edit Distance 2
  Comparision 2: Truncation Distance 1 1
- Block Field Index: 2
- Block Field Avg Length: 7
  You may change the approximate average value of the block field for better performance.

Fig 3. Screenshot of linkage criteria for our 3 example files.

doi:10.1371/journal.pone.0124449.g003

Table 4. Generated output for our example data sets.

Cluster ID	File Name	ID	First Name	Last Name	SSN	Date Of Birth	Gender	School ID	Middle Name	SSID
1	Input02.csv	1	r	pierce	133183594					
1	Input01.csv	1	risa	pierce	133183594	09261990	m	1524	vesta	0676221410
1	Input03.csv	1	risa	pierce		09261990	m	1524	vesta	0676221410
2	Input02.csv	2	m	kramer	135370878					
2	Input01.csv	2	maile	kramer	135370878	07261991	f	1526	lenna	0957261480
2	Input03.csv	2	maile	kramer		07261991	f	1526	lenna	0957261480
3	Input02.csv	3	k	battle	141274186					
3	Input01.csv	3	kimberly	battle	141274186	04071982	f	1527	jacki	0144591609
3	Input03.csv	3	kimberly	battle		04071982	f	1527	jacki	0144591609
4	Input02.csv	4	k	mcclain	148965694					
4	Input01.csv	4	kamal	mcclain	148965694	10091991	m	70000	luisa	0278635088
5	Input01.csv	5	yvonne	vaughan	153614228	02061992	f	70003	basil	0368901550
5	Input03.csv	5	yvonne	vaughan		02061992	f	70003	basil	0368901550
6	Input03.csv	8	kelsie	mueller		01131992	m	70020	jake	7243583370
6	Input02.csv	8	l	mueller	184498846					

doi:10.1371/journal.pone.0124449.t004

threshold value is at least 2, only one cluster having these two records will be generated. A threshold value of 0 generates exact clustering in which every record matches on common attributes. If the users do not input any value, the tool uses the most used threshold value of 1 as the default value. User's email address is required as the output link is sent to this address. For our example of 3 input files, if we choose 1 as the threshold value, then RLT-S will generate the output shown in Table 4.

Submissions history

Users may want to check their previous submissions and outputs. The website allows users to login to view their submissions history. They can check query id, submission time, processed time, download link, etc. of all the submitted jobs.

Any user can use the tool without login information. Valid email address is needed to get the download link of the output. First time users need to reset their password from the "submission info" section to be registered. They can also change their current password from this page.

Feedback

The website provides a feedback option for further improvements to RLT-S. Users can select a feedback type and post comments. We will study the feedbacks very carefully and modify the system accordingly.

RLT-S application has been implemented in Java. We have used Apache server with MySQL to host the website. A service works in the background to trigger the application when a new job is posted. This service also keeps track of the finished jobs, updates database and sends email to corresponding users. MySQL database stores users' information and their submission history. Our tool takes negligible amount of time for thousands of records. If needed, we will integrate our parallel implementation of single linkage clustering algorithm into this website in future.

Table 5. Time comparison of RLT-S with FEBRL, FRIL, and TPA (FCED).

Tool Name	(1000, 1000)	(2000, 2000)	(3000, 3000)	(4000, 4000)	(5000, 5000)
RLT-S	95	110	142	212	237
FEBRL	330	834	1630	2770	4150
FRIL	841	1992	3555	6043	8683
TPA (FCED)	172	223	274	360	433

Times shown are in milliseconds. Computation times are taken for (number of records in first file, number of records in second file).

doi:10.1371/journal.pone.0124449.t005

Comparisons

FEBRL and FRIL are well known and widely used freely available record linkage tools. These tools perform standardization or deduplication of a file or linkage between two files. On the other hand, our system RLT-S can handle any number of input datasets. Several experimental results reported in [12] show that our RLA algorithm outperforms previous best-known algorithms for error-induced datasets. Those experiments also describe the process of choosing suitable threshold values for different datasets. Our algorithm achieves around 98% accuracy on four real datasets having 1 million records in total. This algorithm was 70 times faster than the previous best-known algorithm, TPA (FCED) [13], for these datasets.

Many of the available record linkage tools achieve very good accuracy, but they suffer from higher time complexities to generate linkages among datasets. Efficient TPA (FCED) (Two Phase Algorithm with Faster Computation of the Edit Distance) has been compared with FEBRL for two datasets of different sizes [13]. In this paper we go through some experiments which use the same parameter configurations as [13]. All of these four tools, namely RLT-S, FEBRL, FRIL, and TPA (FCED), achieve 100% accuracy for these simulated datasets. Table 5 shows computation times of these four tools for 1000, 2000, 3000, 4000, and 5000 records. In this table the notation (X, Y) stands for the number (X) of records in the first file and the number (Y) of records in second file.

We see that FEBRL is performing better than FRIL for each of these data sets. TPA (FCED) is faster than FEBRL that is also shown in [13]. RLT-S outperforms all of these tools. Our tool performs its best for real datasets, where the possibility of error occurrences and the number of errors in the input datasets are low.

Conclusions

We have developed a record linkage tool called RLT-S. This tool is integrated into www.rlatools.com. This site contains instructions for usage as well as submissions history and some other useful features. RLT-S is the implementation of our efficient sequential record linkage algorithm, which has outperformed previous best-known algorithms in this area [12]. The tool requires very compact but necessary parameter selections for expected output in the shortest possible time. We also track user movements through this website. Tracking information and user feedback will help us to fine-tune the features and functionalities of RLT-S.

Author Contributions

Conceived and designed the experiments: AM SR. Performed the experiments: AM. Analyzed the data: SR AM RA. Contributed reagents/materials/analysis tools: RA. Wrote the paper: AM SR.

References

1. Clark DE (2004) Practical introduction to record linkage for injury research. Injury Prevention 10(3): 186–191. PMID: 15178677
2. Victor TW, Mera RM (2001) Record linkage of health care insurance claims. Journal of the American Medical Informatics Association 8(3): 281–288. PMID: 11320072
3. Fayyad U, Piatetsky-shapiro G, Smyth P (1996) From data mining to knowledge discovery in databases. AI Magazine 17(3): 37.
4. Maizlish NA, Herrera L (2005) A record linkage protocol for a diabetes registry at ethnically diverse community health centers. Journal of the American Medical Informatics Association 12(3): 331–337. PMID: 15684130
5. Brin S, Davis J, Garcia-Molina H (1995) Copy detection mechanisms for digital documents. ACM SIGMOD Record 24(2): 398–409.
6. Shivakumar N, Garcia-Molina H (1996) Building a scalable and accurate copy detection mechanism. Proceedings of the First ACM International Conference on Digital Libraries. ACM: 160–168.
7. Gomatam S, Carter R, Ariet M, Mitchell G (2002) An empirical comparison of record linkage procedures. Statistics in Medicine 21(10): 1485–1496. PMID: 12185898
8. Gu L, Baxter R, Vickers D, Rainsford C (2003) Record linkage: Current practice and future directions. CSIRO Mathematical and Information Sciences Technical Report 3: 83.
9. Zhao Y, Karypis G (2002) Evaluation of hierarchical clustering algorithms for document datasets. Proceedings of the Eleventh International Conference on Information and knowledge management. ACM: 515–524.
10. Rajasekaran S (2005) Efficient parallel hierarchical clustering algorithms. IEEE Trans. Parallel Distrib. Syst. 16(6): 497–502.
11. Mi T, Aseltine R, Rajasekaran S (2008) Data integration on multiple data sets. Proceedings of the 2008 IEEE International Conference on Bioinformatics and Biomedicine: 443–446.
12. Mamun A-A, Mi T, Aseltine R, Rajasekaran S (2014) Efficient sequential and parallel algorithms for record linkage. Journal of the American Medical Informatics Association: JAMIA 21(2): 252–262. doi: 10.1136/amiajnl-2013-002034 PMID: 24154837
13. Mi T, Rajasekaran S, Aseltine R (2012) Efficient algorithms for fast integration on large data sets from multiple sources. BMC Medical Informatics and Decision Making 12(1): 59.
14. Jurczyk P, Lu JJ, Xiong L, Cragan JD, Correa A (2008) Fine-grained record integration and linkage tool. Birth Defects Research Part A: Clinical and Molecular Teratology 82: 822–829. doi: 10.1002/bdra.20521 PMID: 18985680
15. Jurczyk P, Lu JJ, Xiong L, Cragan JD, Correa A (2008) FRIL: A tool for comparative record linkage. AMIA Annual Symposium Proceedings: 440–444. PMID: 18998844
16. Christen P (2008) Febrl: a freely available record linkage system with a graphical user interface. Proceedings of the Second Australasian Workshop on Health Data and Knowledge Management 80: 17–25.