Information Technology and Jewish Genealogy

Stephen P. Morse

INTRODUCTION

Although this talk has "Jewish Genealogy" in the title, there is nothing fundamental about Information Technology that restricts it to being Jewish. So the topics in this talk relate to genealogy in general and cover the issues of fetching genealogical data and processing genealogical data. The focus will be on using the World Wide Web since that is where much of the data is to be found in the current technological age.

STATE OF THE ART

The key issues in doing genealogical research are the digitizing of data, fetching of data, and the processing of data. Each of those will now be discussed.

Digitizing Data. Data can be digitized manually (transcribed) or automatically (OCR). Manual transcriptions undoubtedly are time-consuming and error prone. OCRing is much less time-consuming. Italso is error prone but in a more predicable fashion. By analyzing the types of errors encountered, methods can be developed for doing automatic error correction. Logan Kleinwaks has done some investigations into such error corrections.

Fetching Data. Data can be fetched manually using a client-based program (e.g., browser) or can be fetched automatically using either a server-based program or a client-based program.

Manual Fetching of Data (Browser based).
Typically the website hosting the data provides a search form that can be used for fetching the data, but in some cases it is more advantageous to use a third-party search form that accesses the same data.

Some advantages of using a third party search form are:

- The number of steps needed to fetch the data can be reduced
- Less navigation through the site is needed before the data can be fetched
- Additional search parameters can be used

Some obvious disadvantages of manual fetching are:

- More work for the user
- More error prone
- Cannot trigger automated processing

Automatic Fetching of Data (Server based).
Automated data fetching may be done in various ways. Two ways will *be* described here, Web services and screen *scraping.*

Web services are services set up by certain websites according to a contract that the website makes with the user. That contracts specifies the URL of the service, the format of the parameters sent to the service, and the format of the results delivered back by the service. Unfortunately, very few sites provide Web services.

Screen scraping is a means of extracting information from a page of results delivered by a website. The page has been formatted for a clean display on a computer screen and not for easy extraction of the underlying data. So a program is needed that wades through the display commands in order to find the relevant data. This method is vulnerable to future formatting changes made on the results page.

The way to implement automated fetching is to run a “man in the middle” server. That is, have a server that receives the original search request and it in turn forwards the request to the real server. The real server then returns the results to the middle man, and it can do whatever processing is necessary before returning the results to the original requestor.

Automated fetching presents the man-in-the-middle’s IP address to the host website. If too many requests come in from the same IP address, the host website might block that address, rendering that server incapable of making any future data accesses.

Automatic fetching of Data (Client based). Existing browser programs do not allow for reading of data unless the reader program comes from the same website domain as that from which the data was fetched. This is done for security reasons, protecting the user from rogue websites. But it also makes is impossible for us to do any processing of data fetched from external sites.

There are two ways to get around this. One is to write our own browser program and allow it to do cross-domain processing. But this means we would have to distribute that browser to all potential users, and that could be a prohibitive task.

The other alternative is to override the browser’s built-in security feature. Usually this can be done by using signed code, and the browser user is asked for his consent before such code is allowed to execute. There are several pitfalls with this approach, one of which is that it might not execute on all browsers.

The one advantage of automated data fetching using the client is that the traffic is charged to the user’s IP address and not to our common server.

Key Search-Engine Features.
Along with fetching of data, we need to consider the features that are provided by the search engine being used. It’s an investigation of such features (and their lacking) that have led me to develop alternate search forms for many existing database websites.

The following are the drawbacks that I have encountered most often:
- Search forms do not permit searches on all the fields transcribed in the database
- Search forms require typing in values that could be selected from a list
- Soundex not supported at all, or supported by making user type in soundex code

Processing Data

Two types of processing that we might like to do with the fetched data are filtering and/or modifying. Both of these are accomplished with the man-in-the-middle as described above.

An example of data filtering is to examine the records being returned and delete those that do not meet additional search constraints. For example, suppose that the underlying website does not allow for searches on age but does display the age of the people found. In that case the man-in-the-middle can filter out those people who are not in a specified age range before returning the results to the requestor.

An example of data modifying would be to change the language of the returned results. Suppose that the website returned the names in Hebrew and we want to have it displayed to the original requestor in English. In this case the man-in-the-middle could do an on-the-fly transliteration before returning the results to the requestor.

AREAS OF FUTURE RESEARCH

Below is a list of possible areas for future research. This list is not arranged in any prioritized order.

1. New techniques for OCR error correction
    a. Integrated with soundex
2. Define standards for genealogical web services
3. Adopt uniform methods of data presentation to simplify screen scraping
4. Expand on Soundex Development

a. Standardize DM soundex and create validation suites
b. Support for other Latin-based languages
c. Support for other alphabets

5. Develop algorithms for processing fetched data in order to

a. Determine equivalency of individuals
b. Construct family trees
c. Merge family trees
d. Determine patterns and create history.