CEUR-WS.org/Vol-3019/LinkedArchives_2021_paper_8.pdf

A Ready-to-Use Solution to Explore Linked Archives with MetaindeX and Gephi*

Laurent Millet-Lacombe[1][0000−0002−2623−9648]

MetaindeX laurentmlcontact-metaindex@yahoo.fr
http://www.metaindex.fr

Abstract. MetaindeX [1] is an open-source [8], online application which offers an opportunity to explore and study linked archives. In this article we will focus on user interface features for loading data, displaying and studying those links within a set of documents from French Archives Nationales.

Keywords: links · graphs · user interface · NoSQL database · metaindex · gephi · archives

1 Context and Data Preparation

1.1 Corpus Overview

We will work with a corpus of more than 2800 references from French archives. This corpus is issued from a PhD work, whose author kindly accepted to lend its data for the demonstration.

Contents are mainly extracted from notarial archives (estate lawyers) from 16th-17th century in Paris. Delivered corpus is made of a set of Excel files for a total of about 20000 lines. Each line represents a specific person (full name, gender, professional situation, textual description of personal and professional connections), seen within a specific archive (date, type, institution, document id).

1.2 Documents and Links Modelisation

NoSQL Database MetaindeX tool relies on ElasticSearch NoSQL database [5], whose model can be summarized as a set of catalogs (called 'indices'), each catalog containing documents, each document being made of a unique identifier and some fields (string or number mainly). Though this database does not define constraints between fields, a minimal schema definition is still required, as a list of name and type of available fields. This schema is defined at catalog level, and then a document from this catalog can use any of those fields.

* Copyright 2021 for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

Links modelisation Links are represented as a 'string' field in database, containing identifiers of target documents to point at. As an example, if a catalog contains 3 documents with ID "doc001","doc002","doc003" and we want document "doc002" to reference documents "doc001" and "doc003", then "doc002" will contain a field (which we will call 'doclink' for this example) with text "doc001,doc003" as a value. Interpretation of this string is then done by application when needed.

An additional 'weight' information can be associated to each link by adding a suffix ':<weight>' to the referenced id. Following our previous example, value "doc001:4,doc003" for our 'doclink' field would then mean a link to "doc001" with a weight of 4 and a link to "doc003" with implicit weight of 1. This convention can be directly used in CSV files, where a column called 'doclink' for our example would contain, for each line, IDs and weight of documents to point at.

Actual meaning of links weight is left to user interpretation. As an example, if the link is between two "cities", it could represent some amount of commercial exchange, while if it is between two artists, it could represent the number of peaces of art they own from each other.

2 Preparing, Uploading and Exploring Data and Links

From original Excel files have been extracted a list of unique persons on the one hand and a list of unique archives on the other hand, as two separate CSV files (about 4400 individuals and 2800 archives), with the help of OpenRefine tool [7] for data cleaning, and a set of dedicated python scripts for data extraction and formatting. Also links have been reconciled for both personal and professional relationships between persons.

Uploading contents into the server is done by dragging CSV files over the catalog contents in the user interface. MetaindeX CSV-import module allows then user to map CSV columns to new or existing fields in the catalog.

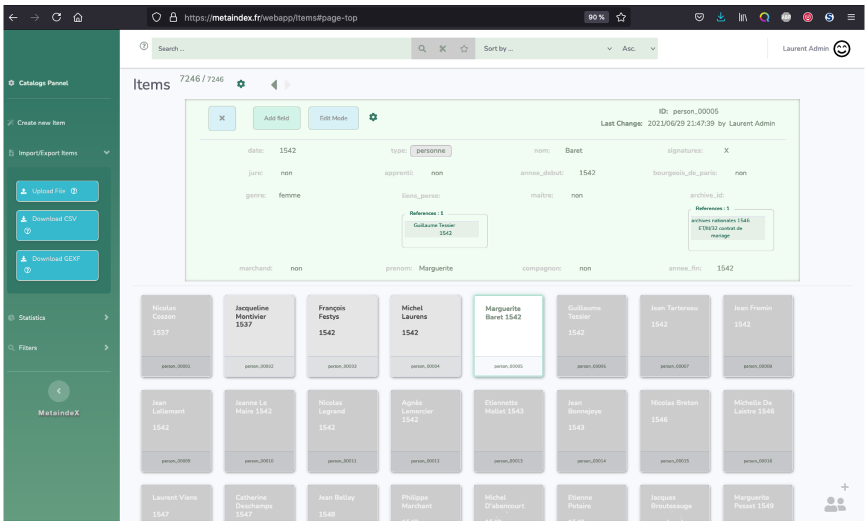
Once loaded, documents are represented as cards, each card being a single document, i.e. a single line from our input CSV files (in our case either an archive or a person). Each card can be expanded to see or edit document's fields as illustrated on Figure 1.

We can also notice on Figure 1 that links are resolved to get a more user-friendly summary of corresponding document, rather than simply IDs list. Also jumping from one document to another, following the links, is possible by clicking on them, allowing user to navigate through those connections.

Lucene query syntax [2] is available as a search engine, and allows advanced search, such as for example "find all persons whose first name approximately equals to 'Antoine' and was born before year 1700", which would be translated with following query:

```
type:person AND firstname:Antoine~ AND datestart:<1700
```

At last, though its usage and possibilities are out of scope of this demonstration, we can precise that a Kibana [6] module is integrated to MetaindeX, allowing user to create advanced statistic charts on its corpus.



ID: person_00005
Last Change: 2021/06/29 21:47:39 by Laurent Admin

date: 1542
type: personne
nom: Baret
signatures: X
jure: non
apprenti: non
annee_debut: 1542
bourgeois_de_paris: non
genre: femme
liens_perso: References : 1 Guillaume Tessier 1542
maitre: non
archive_id: References : 1 archives nationales 1546 ET/III/32 contrat de mariage
marchand: non
prenom: Marguerite
compagnon: non
annee_fin: 1542

Nicolas Cosson 1537 person_00001
Jacqueline Montivier 1537 person_00002
François Festys 1542 person_00003
Michel Laurens 1542 person_00004
Marguerite Baret 1542 person_00005
Guillaume Tessier 1542 person_00006
Jean Tartereau 1542 person_00007
Jean Fremin 1542 person_00008
Jean Lallemant 1542 person_00009
Jeanne Le Maire 1542 person_00010
Nicolas Legrand 1542 person_00011
Agnès Lemercier 1542 person_00012
Etiennette Mallet 1543 person_00013
Jean Bonnejoye 1543 person_00014
Nicolas Breton 1546 person_00015
Michelle De Laistre 1546 person_00016
Laurent Viens 1547
Catherine Deschamps 1547
Jean Bellay 1548
Philippe Marchant
Michel D'abencourt
Etienne Potaire
Jacques Broutesauge
Marguerite Pesset 1549

Fig. 1. Screenshot from MetaindeX when consulting contents of a document (here a character)

3 Generating Graphs

3.1 Basic Graph Generation

MetaindeX is able to generate a graph description file (GEXF format [3]) compatible with main graph applications such as Gephi [4]. In such a generated graph, each document would be a node and each link would be an edge. Generation module allows to select which fields to be used as nodes' metadata, allowing fine graph rendering customization based on our contents, for example by assigning a color to nodes depending on value of a given field.

Detailed usage of Gephi is out of the scope of this article, but once GEXF file generated from our data and loaded in Gephi, with only few settings we can already identify some clusters within professional and personal relationships among persons, as shown on Figure 2 (each grey dot represents a person, names have been hidden for better readability of the networks).

We can see there some clusters already well identified, where several links seem to converge around same groups of individuals. Those links were created each time two persons were found to have a professional relationship (for example master and student), seen in green, or personal relationship in pink (mariage witness, siblings, etc.). That information could help the researcher to get a better vision of social and professional relationships over his corpus, and maybe interpret with better accuracy historical facts he could find on archives contents.

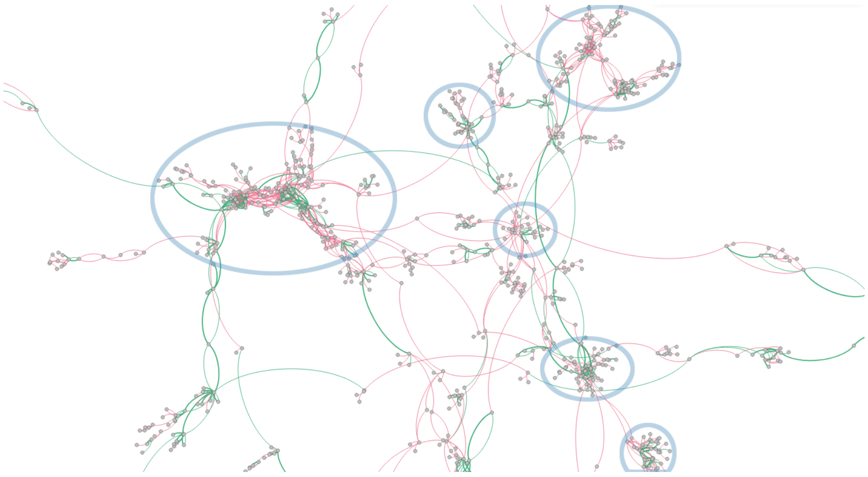


Fig. 2. Graph (detail) of professional (green) and personal (pink) relationships among individuals (grey dots).

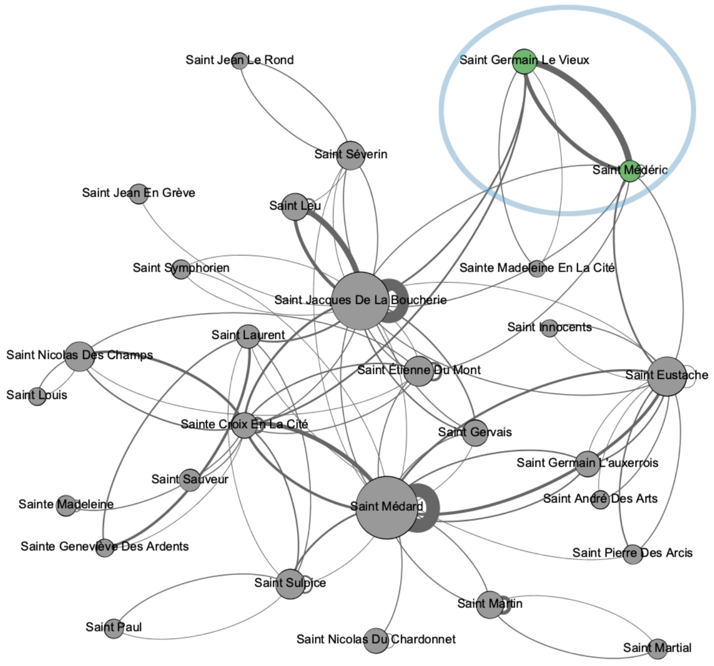


Fig. 3. Graph of professional relationships grouped by parish.

3.2 Aggregated Graph Generation

Since bigger graphs might be more difficult to read, MetaindeX offers also a specific algorithm to generate simplified aggregated graphs. This way, nodes and links are gathered up following values of a specific field. This "group-by" feature allows to get much more readable graphs directly focused on topic the researcher is interested in.

Following our example, we can group persons by parish they live in, which means that all persons having a "parish" field with a similar value will be grouped within a single node, and their respective links will also be aggregated to this node.

Figure 3 shows as a result all parishes found in the corpus, their size depending on amount of individuals registered as living there, while links thickness being based on amount of links that all individuals from given parish have with individuals from another parish. On that graph, we can see that parishes Saint-Germain-Le-Vieux and Saint-Médéric (top right) seems to have quite numerous professional relationships (thicker link) despite their smaller amount of persons recorded to live in (smaller nodes size) from our corpus. That could maybe lead the researcher to a new approach or hints to understand social relationships of this community.

4 Conclusion

Demonstration has been done that, starting from a consistent set of data, MetaindeX loads quickly and efficiently the corpus, let us explore contents with advanced queries and links navigation. GEXF export module, with both basic and group-by algorithms, let us easily load our data in Gephi for advanced graph exploration. If phase of data preparation remains a major step requiring sometimes some coding skills, the proposed environment made of MetaindeX and Gephi tools, offers a ready-to-use, coding-free, quick and efficient ecosystem to explore a set of linked archives for the researcher.

References

1. MetaindeX, http://www.metaindex.fr. Last accessed 3 Jun 2021
2. Lucene Query Language, https://lucene.apache.org/core/. Last accessed 3 Jun 2021
3. GEXF file format, https://gephi.org/gexf/format/. Last accessed 3 Jun 2021
4. Mathieu Bastian and Sebastien Heymann and Mathieu Jacomy. Gephi: An Open Source Software for Exploring and Manipulating Networks, http://www.aaai.org/ocs/index.php/ICWSM/09/paper/view/154.2009
5. ElasticSearch, https://www.elastic.co/guide/en/elasticsearch/reference/current/elasticsearch-intro.html. Last accessed 3 Jun 2021
6. Kibana, https://www.elastic.co/guide/en/kibana/current/introduction.html. Last accessed 3 Jun 2021
7. OpenRefine, https://openrefine.org/. Last accessed 3 Jun 2021
8. MetaindeX source code, https://github.com/laurentmldev/metaindex. Last update Aug 2021