# Data acquisition, extraction, and storage

Pierre Senellart

12 December 2023

This exam lasts three hours. The only documents allowed are ten A4 sheets (both sides), with the content of your choice. Communicating devices are strictly forbidden.

When writing code, imprecision in the syntax of the languages will be tolerated.

The exam is formed of a single problem, graded out of 20 points.

## A dataset of scientific research articles in computer science

In this exam, we are investigating how to produce a dataset of scientific research articles in computer science, that could be used, for instance, to train a large language model specialized in that area.

We want to construct a dataset with information about all kinds of scientific papers relevant to computer science, both published and preprints. For each paper, we would like to obtain:

- its title;
- its list of authors, with the affiliation of each author;
- the venue (conference, journal) it is published and the publisher, if any;
- the year it was published;
- its DOI (an identifier assigned to every published paper, which is also a URL linking to the Web page the paper can be accessed from, if it is in open access), if any;
- its abstract, if available;
- the full text of the paper, if available.

To do that, we consider using the following sources:

**Crossref** a comprehensive database of all scientific publications in all fields (with titles, authors, affiliations, venues, publishers, years, DOIs, and sometimes abstracts), but that does not contain any information about preprints. Information comes directly from publishers, and is about $\approx 150\,000\,000$ papers.

**DBLP** a curated database of publications and preprints in venues specific to computer science (with titles, authors, venues, publishers, years, DOIs). Information comes from the curators of the database, but it is limited in scope (some subfields of computer science are not represented, some minor venues and preprint repositories are not indexed), and is about $\approx 7\,000\,000$ papers.

**arXiv** a preprint repository with a specific category (`cs`) for computer science, with full text and basic information (titles, authors, years, abstracts) of preprints that are stored there. It contains around 2 million preprints, roughly one third of them in computer science.

**the Web** where other preprint repositories can be accessed and open-access publications can be downloaded.

1. (1 point) Estimate the total size of the resulting dataset, assuming that DBLP contains roughly one half of the amount of relevant papers, that the metadata of a paper weighs around 1 kB on average and that the full text of a paper (available in around 25% of all papers) weighs around 100 kB on average.

2. (2 points) Propose a relational schema suitable for storage of all relevant data. Make sure that your schema limits redundancy of the data as much as possible (a specific fact should only occur once). Specify which attributes may be null.

3. (3 points) Crossref proposes a JSON REST API to obtain meta-information about a paper by DOI. For intance, `http://api.crossref.org/works/10.1038/nature14539` returns a JSON document which looks like (simplified from the real example):

```
{
  "status": "ok",
  "message": {
```

```
    "publisher": "Springer Science and Business Media LLC",
    "DOI": "10.1038/nature14539",
    "title": [
      "Deep learning"
    ],
    "author": [
      { "given": "Yann", "family": "LeCun", "affiliation": [] },
      { "given": "Yoshua", "family": "Bengio", "affiliation": [] },
      { "given": "Geoffrey", "family": "Hinton", "affiliation": [] }
    ],
    "published-online": {
      "date-parts": [ [ 2015, 5, 27 ] ]
    },
    "container-title": [ "Nature" ],
    "issued": { "date-parts": [ [ 2015, 5, 27 ] ] }
  }
}
```

Either using Python and a JMESPath-like JSON querying language, or the command line and the jq processor, write code that issues a query to Crossref's JSON API to extracts all relevant metadata about a paper given its DOI, in a form suitable to import into your relational schema.

4. DBLP provides an XML export of its database, which looks like this (showing only a single article, and simplifying somewhat):

```
<dblp>
  <article key="journals/nature/LeCunBH15">
    <author>Yann LeCun</author>
    <author>Yoshua Bengio</author>
    <author>Geoffrey E. Hinton</author>
    <title>Deep learning.</title>
    <year>2015</year>
    <journal>Nat.</journal>
    <publisher>Springer</publisher>
    <ee>https://doi.org/10.1038/nature14539</ee>
    <ee>https://www.wikidata.org/entity/Q28018765</ee>
  </article>
</dblp>
```

   a) (1 point) Write an XPath expression that would return the title of all articles published by Yann LeCun over the DBLP database.
   b) (2 points) In pseudo-code, propose a SAX parser that would extract all relevant metadata in a form suitable to import into your relational schema.

5. (1 point) As shown above, some DBLP papers have links to Wikidata entities about these papers. Wikidata may have some more interesting metadata about this paper. Propose a SPARQL query that returns all facts known about a given Wikidata entity found in the DBLP dataset.

6. (2 points) Articles found on Crossref, DBLP, arXiv, Web can be identical; when the DOI is present, it can be used to identify uniquely a paper; but sometimes no DOI is available and one has to determine whether two papers are actually the same, based on their metadata. Propose a deduplication approach to do this.

7. (2 points) Adapt your deduplication approach so that it can be run as a MapReduce program. In pseudo-code, detail what will be done in the Map and Reduce phases. You can choose to simplify the approach to make it more amenable to distributed computation, but it should still be a reasonable way to deduplicate articles.

8. (4 points) Propose an overall approach that combines information from all four sources to collect data about all relevant papers. You should explain in detail how to use Crossref (you can make some assumptions about its available APIs), how to use DBLP (using the XML dump), arXiv (you can make some assumptions about is API), how to retrieve relevant data from the Web, and how to combine all of these in a single clean dataset of metadata and data of scientific articles in computer science.

9. (2 points) Estimate for each source how costly your proposed approach would be (in terms of number of API calls, number of HTTP requests, etc.).