Using recibi in EDG

BV

April 20, 2024

1 Context

Publications and documents such as reports for funding review require bibliographies that are narrowed to a selection of reference defined based on various criteria such as.

- By group member author.
- Published in a given time period.
- Produced by a specific collaboration.
- Is a of a general reference nature.
- Forming a short, select list for CVs.

The publications and other documents are often produced by many authors and the number of references produced by the group and otherwise referenced is large. Thus some automated way to manage this variety and volume of references is desired.

2 General approach

The approach to managing references illustrated here is broken down into these major areas:

1. A number of **reference identification sources** are provided, typically hand-curated.
2. A **parser** for each source extracts **reference identifiers**.
3. A **bibliography resolver** derives bibliography database (BibTeX) entries when queried with reference identifiers to produce bibliography database files (`.bib`).
    - InspireHEP web API
4. A **bibliography database tool** transforms `.bib` files.
5. A **workflow** automates parsing, deriving and transforming so a final result may be reproduced when a **source** or a **resolver** updates.
6. LaTeXprovides support of consuming such `.bib` files to produce a variety of bibliography lists.

2.1 Sources

The sources of reference identifiers include:

- EDG Publications by year list.
- EDG publications categories spreadsheet.
- List of EDG member authors.

The primary document identifier is of the type `arXiv:XXXX.YYYYY`. The primary author identifier is of the type `author:NNNNN`. Both are as recognized by the InspireHEP web API.

2.2 Parsers

Simple parsers for document identifiers consist of regex matching on the `arXiv:` prefix or by matching `XXXX.YYYYY` given a limited document context.

The `recibi parse` provides a simple regex based parsing mechanism to achieve this for sources provided as text files.

InspireHEP provides the tool refextract to extract references as JSON from PDF files. This JSON is similar to what is available from the InspireHEP web API. In practice, document identifiers should be scraped from the JSON and used to query the API to render BibTeX. (Note, `recibi` currently has no support for this).

2.3 Resolver

The primary resolver is the InspireHEP web API. It may be queried via `recibi inspire`.

2.4 Transforms

The `recibi` commands `tag`, `merge` and `filter` are provided.

Other tools such as `bibtool` exist.

2.5 LaTeX

See blow and the example.

3 Conventions

With the above in place, the following tagged sets of references can be formed:

`edg` references identified from the EDG Publications by year list.

`snowmass` references from the tab of this name in the .... spreadsheet.

`collaboration` a tag formed from the value of any `collaboration` field.

`author` a tag placed on all references from an `author:XXX` query.

These tagged sets may each be initially produced in one or more `.bib` files. These files can be combined with `recibi merge` into a single `.bib` file. This merge operation is “deep” in that the final `keywords` set of a given entry is the union of all that are found for that entry. A union is also formed over any disparate fields each entry of a given key. Though when all entries for a given bib key are derived through InspireHEP they will be identical.

Manually curated BibTeX database files (`.bib`) can be accommodated in two ways. First, they may simply be added along with generated files in the LaTeX. In this manner all entries across all files are required to have unique keys and each should refer to a unique publication. In particular, this method of aggregating bib files affords no way to override entries.

The second approach is use `recibi merge` to merge generated and curated bib files to a single file. This allows for the curated file to override or “patch” a generated file. This approach also allows for effort to not be lost as the human may edit the curated file and the merge can be repeated. Editing a generated file will typically ensure that human effort will be lost when the file is inevitably regenerated in order to incorporate updates to the upstream curated lists.

4 Example workflow

This section outlines one possible workflow. It is not meant to be definitive but to give a flavor of what is possible. It starts with visiting the EDG Publications by year and selecting “Download as Plain Text” to produce the `EDG-Publications.txt` file which represents one upstream source.

```
$ recibi parse -o edg.arxiv -m ’(ar[Xx]iv:(\d+)\.(\d+))’ EDG-Publications.txt
$ recibi inspire -o -S 1000 edg.arxiv | \
  recibi tag -o edg.bib -t edg -T collaboration
```

This second step applies the `edg` tag and “transfers” the value of the `collaboration` field to the `keywords` set.

To generate a bib file holding references to documents for which a single EDG member contributed one must know an InspireHEP author identifier.

```
$ recibi inspire -S 1000 author:B.Viren.1 | \
  recibi tag -t bv -T collaboration -o bv.bib -
```

Like the prior example, this applies a tag `bv` to all and transfers the value of any `collaboration` field to the `keywords` set.

To merge bib files.

```
$ recibi merge -o edg+bv.bib edg.bib bv.bib
```

This command can be extended to any number of input bib files including those that are curated by human editing.

5 LaTeXexample

See the file `example.tex` which uses two BibTeX files: `generated.bib` and `curated.bib`. These contrived and brief in order to show what can be done without getting lost in verbose listings. The result of building the example is available as a PDF file. See the text of the example for explanation details.