Design Automation Renegades

Globetrotting Division

BibTeX Analytics: For Automating Reference Management and Recognizing Emerging Trends

Zhiyang Ong [1]

A document on *Python*-based BibTeX analytics
For Reference Management ...
and Emerging Trend Recognition

May 23, 2018

[1]Email correspondence to: ongz@acm.org

Abstract

This documents how the repository of the BibTeX *Analytics* project is organized, and its software architecture. It also describes the future goals of the project for using a data analytics approach to recognize emerging trends in research, especially emerging research trends in electrical and computer engineering, computer science, and other fields, such as medicine, agriculture, and environmental science.

Insert abstract here.

More stuff to be included.

Revision History

Revision History:

1. Version 0.1, May 21, 2018. Initial copy of the report.
2.

Contents

Chapter 1

Organization of the BibTeX Analytics Repository

The main deliverables of the BibTeX *Analytics* project are a *Python*-based software to perform reference management, and data analytics on BibTeX entries to recognize emerging research trends.

The organization of the BibTeX *Analytics* repository is described as follows:

1. `automated_regression_testing.py`:
   (a) Run as: **./automated_regression_testing.py**
   (b) No input nor output required.
   (c) *Python* script to automate regression testing.
   (d) **Sort of deprecated.** Still works though.
2. big_input:
   (a) Data set for stress testing the software deliverable of the BibTeX *Analytics* project.
3. database
   (a) *bibtex_database* (`bibtex_database.py`) class represents (each instance of) a BibTeX database of BibTeX entries.
   (b) `bibtex_database_test.py` is a *Python* script to test the functionality of the *bibtex_database* class.
   (c) *entry* (`entry.py`) class represents each (instance of a) BibTeX entry.
   (d) `entry_test.py` is a *Python* script to test the functionality of the *entry* class.
   (e) `key_check.py` is a *Python* script to check if each BibTeX entry is valid.
   (f) `key_check_test.py` is a *Python* script to test the functionality of the *Python* script `key_check.py`.
   (g) `key_frequency_pairs.py` is a *Python* script to demonstrate how to sort a set of 2-tuples based on its first/former field and its second/last/latter field.
4. `duplicate_BibTeX_entries.py`:
   (a) Run as: **./duplicate_BibTeX_entries.py [-h] [BibTeX file]**
   (b) A *Python* script to determine if duplicate BibTeX entries exist in a BibTeX file/database. If such entries exist, warn the user that duplicate BibTeX entries exist.
5. `editions.py`:
   (a) Run as: **./editions.py [input BibTeX file] [-h]**
   (b) A *Python* script to display a set of editions from all the BibTeX entries in a BibTeX file/-database.
6. `incremental_test.py`:

   (a) Run as: **./incremental_test.py [input BibTeX file]**
   (b) A *Python* script to incrementally test features for performing reference management and data analytics operations with BIBTEX files/databases.

7. input:
   (a) A set of BIBTEX files to test my *Python*-based BIBTEX *Analytics* software.

8. `institutions.py`:
   (a) Run as: **./institutions.py [input BibTeX file] [-h]**
   (b) A *Python* script to display a set of institutions from BIBTEX entries in a BIBTEX file/database.

9. `journal_titles.py`:
   (a) Run as: **./journal_titles.py [input BibTeX file] [-h]**
   (b) A *Python* script to display a set of journal titles from BIBTEX entries in a BIBTEX database.

10. `keywords_display.py`:
   (a) Run as: **./keywords_display.py [input BibTeX file] [-h]**
   (b) A *Python* script to display a set of keywords/keyphrases from BIBTEX entries in a BIBTEX database.

11. makefile:
   (a) For build automation of *Python* scripts, not placed in subdirectories, in the repository.

12. notes:
   (a) `gpl-license.text`, `LICENSE`, and `mit-license.text` are text files of the GNU General Public License (GNU GPL) (`gpl-license.text`) and The MIT License (`LICENSE` and `mit-license.text`)
   (b) guidelines:
      i. A document containing a set of guidelines on how to collaborate with me.
   (c) report:
      i. This document that describes the organization of the BIBTEX *Analytics* repository, the software architecture of the BIBTEX *Analytics* software, and future work of the BIBTEX *Analytics* project.

13. `organizations.py`:
   (a) Run as: **./organizations.py [input BibTeX file] [-h]**
   (b) A *Python* script to display a set of organizations from BIBTEX entries in a BIBTEX database.

14. `publishers.py`:
   (a) Run as: **./publishers.py [input BibTeX file] [-h]**
   (b) A *Python* script to display a set of publishers from BIBTEX entries in a BIBTEX database.

15. `readme.md`:
   (a) A *Markdown*-based `readme` document briefly describing this project.

16. `rm_bibtex_metadata.py`:
   (a) Run as: **./rm_bibtex_metadata.py [input BibTeX file] [output BibTeX file] [-h]**
   (b) *[output BibTeX file]* is an optional parameter.
   (c) A *Python* script to delint/remove BIBTEX metadata from a BIBTEX database/file.

17. sandbox:
   (a) A set of *Python* scripts to test different concepts in *Python*.

18. statistics:

   (a) `test_statistics_tester.py` is a *Python* script to test the functionality of the *test_statistics* class.
   (b) *test_statistics* (`test_statistics.py`) class to perform statistical analysis on results of automated testing of a *Python* script.

19. `tutti_series.py`:
   (a) Run as: **./tutti_series.py [input BibTeX file] [-h]**
   (b) A *Python* script to display series from BIBTEX entries in a BIBTEX database.

20. utilities:
   (a) `file_io.py` is a *Python* script to perform input/output (I/O) operations on files, such as BIBTEX databases/files and LATEXdocuments.
   (b) `queue_ip_arguments.py` is a *Python* script to process input arguments for a script to clean BIBTEX databases/files.

21. `validate_url.py`:
   (a) Run as: **./validate_url.py [input BibTeX file] [output BibTeX file] [-h]**
   (b) *[output BibTeX file]* is an optional parameter.
   (c) A *Python* script to check each BibTeX entry in a BibTeX database if it has the non-standard BIBTEX field(s) "Bdsk-Url-1" (and "Bdsk-Url-2"), and if the "Url" (and "Doi") field(s) is(/are) missing; if these conditions are true, copy their values to the "Url" BibTeX field (and "Doi" field, if it is a DOI).

22. `z_booktitles.py`:
   (a) Run as: **./z_booktitles.py [input BibTeX file] [-h]**
   (b) A *Python* script to display booktitles from all BIBTEX entries in a BIBTEX database.

Chapter 2

Software Architecture of the BibTeX Analytics Project

Software Architecture of the BibTeX *Analytics* Project

Chapter 3

Future Work

Future work of the BibTeX *Analytics* project is described as follows:

1. Clustering of keywords/keyphrases:
   (a) **Problem statements**:
      i. For an author, find clusters of keyphrases, publishers, journal titles, conferences, ...
      ii. For each keyphrase, determine the cluster of publishers, years, journal titles, conferences, ...
   (b) Build dictionary of *(keyphrase, frequency)* two-tuples (or pairs).
   (c) Sort the dictionary based on the frequency term/element, *frequency*, in these two-tuples.
   (d) Alternate solution:
      i. Build a set of *(keyphrase, [list of years])* lists.
      ii. *[list of years]* is a list of years of publications; or it is a set of years for publications that include the keyphrase *keyphrase*.
      iii. Sort the set based on length of the list of years, *[list of years]*.
2.
3.
4.
5.
6.
7.
8. Find emerging research trends to consider pivoting towards, or to get involved in side projects
   (a) E.g., benchmark adiabatic quantum computers with topological computers and universal quantum computers [**?**].