

Design Automation Renegades

GLOBETROTTING DIVISION

BIB_TE_X Analytics: For Automating Reference Management and Recognizing Emerging Trends

Zhiyang Ong¹

A DOCUMENT ON *Python*-BASED BIB_TE_X ANALYTICS
For Reference Management ...
and Emerging Trend Recognition

May 23, 2018

¹Email 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

Revision History	i
1 Organization of the BIB _T _E X Analytics Repository	1
2 Software Architecture of the BIB _T _E X Analytics Project	4
3 Future Work	5
Bibliography	6

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) *Python* script to automate regression testing.
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) 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) A *Python* script to display a set of editions from all the BIBTEX entries in a BIBTEX file/-database.
6. `incremental_test.py`:
 - (a) 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) A *Python* script to display a set of institutions from B_IB_TE_X entries in a B_IB_TE_X file/database.
9. `journal_titles.py`:
 - (a) A *Python* script to display a set of journal titles from B_IB_TE_X entries in a B_IB_TE_X database.
10. `keywords_display.py`:
 - (a) A *Python* script to display a set of keywords from B_IB_TE_X entries in a B_IB_TE_X 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 B_IB_TE_X *Analytics* repository, the software architecture of the B_IB_TE_X *Analytics* software, and future work of the B_IB_TE_X *Analytics* project.
13. `organizations.py`:
 - (a) A *Python* script to display a set of organizations from B_IB_TE_X entries in a B_IB_TE_X database.
14. `publishers.py`:
 - (a) A *Python* script to display a set of publishers from B_IB_TE_X entries in a B_IB_TE_X database.
15. `readme.md`:
 - (a) A *Markdown*-based `readme` document briefly describing this project.
16. `rm_bibtex_metadata.py`:
 - (a) A *Python* script to delint/remove B_IB_TE_X metadata from a B_IB_TE_X 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) A *Python* script to display series from B_IB_TE_X entries in a B_IB_TE_X database.
20. `utilities`:
 - (a) `file_io.py` is a *Python* script to perform input/output (I/O) operations on files, such as B_IB_TE_X databases/files and L^AT_EX documents.
 - (b) `queue_ip_arguments.py` is a *Python* script to process input arguments for a script to clean B_IB_TE_X databases/files.
21. `validate_url.py`:

- (a) A *Python* script to check each BibTeX entry in a BibTeX database if it has the non-standard BIB_TE_X 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) A *Python* script to display booktitles from all BIB_TE_X entries in a BIB_TE_X database.

Chapter 2

Software Architecture of the BIB_TE_X Analytics Project

Software Architecture of the BIB_TE_X *Analytics* Project

Chapter 3

Future Work

Future work of the $\text{BIB}\text{T}_{\text{E}}\text{X}$ *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*])
 - 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 [1].

Bibliography

- [1] Prateek Tandon, Stanley Lam, Ben Shih, Tanay Mehta, Alex Mitev, and Zhiyang Ong. Quantum Robotics: A Primer on Current Science and Future Perspectives, volume 10 of Synthesis Lectures on Quantum Computing. Morgan & Claypool Publishers, San Rafael, CA, January 2017.