# An automated approach to organizing BibTeX files

Rohde Fischer - 20052356

Master's Thesis, Computer Science

June 12, 2016

Advisor: Olivier Danvy

# Abstract

We successively describe the bibliographic software tool BibTEX (both how to use it in principle and how it is used in practice), list a range of practical issues BibTEX users encounter, propose an approach to handling these issues, and review how they are tackled in related work. We then present an analysis of BibTEX files that detects these issues and we describe how to solve them by organizing BibTEX files using the results of the analysis. We implemented a proof of concept, Orangutan, that analyzes existing BibTEX files and emits diagnostics and suggestions.

# Resumé

Vi beskriver successivt det bibliografiske software-værktøj BibTEX (både, hvordan det bruges i princippet og hvordan det bruges i praksis), lister en række praktiske problemer BibTEX-brugere møder, foreslår en fremgangsmåde til håndtering af problemerne og en gennemgang af, hvordan de takles i relateret arbejde. Vi præsenterer en analyse af BibTEX-filer, der påviser problemerne og vi beskriver, hvordan disse kan løses ved at organisere BibTEX-filerne ved hjælp af resultaterne fra analysen. Vi har implementeret et proof of concept, Orangutan, der analyserer eksisterende BibTEX-filer og udskriver diagnoser og forslag.

# Acknowledgments

Olivier Danvy advised the work that led up to this dissertation. His constant support and help have been of immense value.

René Rydhof Hansen served as an external evaluator and asked insightful questions at the defense. I am grateful for his time and attention.

Anders Lindkvist, Troels Fleischer Skov Jensen, and Martin Eik Korsgaard Rasmussen were fantastic office mates. They offered critical feedback on the work, comments on the dissertation, and moral support throughout. Thank you guys.

Ditte Maria Mikkelsen, Livia-Marina Gherghe, and Vera Ohlsen generously provided suggestions and comments on the dissertation. I am grateful for their feedback.

Throughout the last 5 months, the JabRef community has, one more time, proved its responsiveness pretty much 24/7.

The staff at Aarhus University has been providing a fruitful study environment. Through their diligent work, they have ensured the relevant infrastructure for my studies, from maintaining the computer systems to assisting with practical issues. They will never been thanked enough as generation after generation of students graduate from the university.

Book Studenterkørsel has provided a student job with an understanding and acceptance of the requirements for my studies. Both my boss and colleagues have been working hard on providing a pleasant balance between studies and work. A huge thanks is due to them.

My siblings, Karl and Anne, and my sister in law, Julija, have been tirelessly providing moral support, help, and feedback. I am grateful to them and to the rest of my family.

My good friends, Peter Laursen, Iskra Dinkova, Livia-Marina Gherghe, and Nina Sprogeø have graciously been providing constant moral support. You guys rock.

Kim Andersen, Rasmus Langager Berg, and Michael Lycke have been helping me with practicalities throughout this thesis work, and I am grateful to them for their kind effectiveness.

Finally, I'd like to add my name to the countless list of users who are grateful to Don Knuth, Leslie Lamport, and Oren Patashnik for LaTeX and BibTeX.

# Contents

# Chapter 1

# Introduction

*A beginning is the time for taking the most delicate care*
*that the balances are correct.*
– Frank Herbert

BibTEX is a software tool for handling bibliographic references in LATEX documents. For example, this dissertation is formatted in LATEX and its bibliographic references are handled with BibTEX. In this dissertation, we document an automated method for detecting lexical issues and inconsistencies in BibTEX files.

To this end, we successively describe BibTEX – both how to use it in principle and how it is used in practice (Chapter 2), we list a range of practical issues BibTEX users encounter (Chapter 3), we propose an approach to handling these issues (Chapter 4), and we review how they are tackled in related work (Chapter 5). We then present an analysis of BibTEX files that detects these issues (Chapter 6) and we describe how to solve them by organizing BibTEX files using the results of this analysis (Chapter 7).

# Chapter 2

# BibTEX

## 2.1 Introduction

The goal of this chapter is to introduce BibTEX: the course of events leading to it (Section 2.2), what it is (Section 2.3), how it is used in principle (Section 2.4), and how it is used in practice (Section 2.5).

## 2.2 Why BibTEX came to be

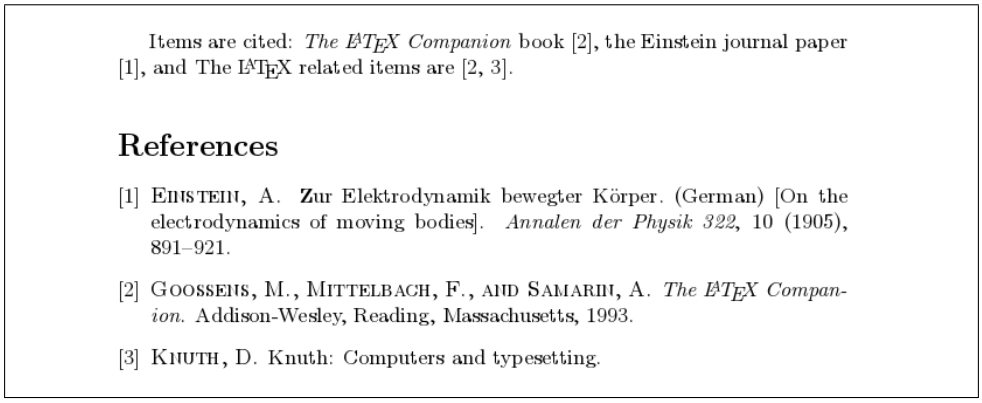
The advent of TEX has been a game changer for scientific writing, witness the number of articles written in TEX (or derivatives such as LATEX). For instance, on arXiv.org, nearly all articles are formatted with LATEX. Apart from being grateful to Donald Knuth (the creator of TEX) for providing such a robust system for scientific articles, writers can also be thankful that TEX has given the basis for Oren Patashnik and Leslie Lamport to create BibTEX for managing scientific bibliographies.

Before the wake of BibTEX, bibliographic references were managed entirely by hand and required a lot of labor. For instance, nearly half of Mary-Claire van Leunen's book, *A Handbook for Scholars* [23], is dedicated to citing, managing, and writing references, i.e., 150 out of 335 pages (excluding the index). Likewise, a significant fraction of Umberto Eco's book, *How to Write a Thesis* [13], is also dedicated to managing and citing references and to ensuring a proper bibliography. All this manual labor has almost been made obsolete by BibTEX.

Furthermore, BibTEX entries are now readily available online, and so the practical problem of bibliographic references is now solved, in principle.

## 2.3 What is BibTEX

In the same spirit as LATEX, BibTEX is a simple software tool for managing bibliographic references in scientific writing, using an ASCII file to specify



Items are cited: *The LaTeX Companion* book [2], the Einstein journal paper [1], and The LaTeX related items are [2, 3].

## References


[1] Einstein, A. Zur Elektrodynamik bewegter Körper. (German) [On the electrodynamics of moving bodies]. *Annalen der Physik 322*, 10 (1905), 891–921.

[2] Goosseins, M., Mittelbach, F., and Samarin, A. *The LaTeX Companion*. Addison-Wesley, Reading, Massachusetts, 1993.

[3] Kiuth, D. Knuth: Computers and typesetting.


**Figure 2.1:** BibTeX output using ACM citing style which uses numbers to index entries. Source: ShareLaTeX [36]

these references. Inside this ASCII file, the components of each reference are specified, such as the author, title, year and what kind of medium (e.g., a book or article) was used. This file will be referred to as 'the BibTeX file' or '.bib file'

Depending on the forum, there are differences in: how references should be cited, how they should be written, and the order in which the references should be listed. Each publisher has a mandatory setup. The specific set of rules a publisher has is referred to as a 'bibliography style' or 'citing style'.

When processing a document, BibTeX cites according to the selected bibliography style, ensures the formatting and the order in the reference list, and ensures that only relevant entries are included. The references are labeled consistently in the document according to the citing style, the labels are used when a reference is cited and these labels allow the reader to quickly find the reference in the bibliography, as illustrated in Figure 2.1.

Today, BibTeX has also enabled huge online resources with references, automated extraction tools, sharing of references and easy version control. The BibTeX format is originally designed for use with LaTeX, but it has plugins for other formats as well [15].

## 2.4 How BibTeX is used in principle

### 2.4.1 Macro-use

A BibTeX file consists of entries each corresponding to a bibliographic reference, such as an article or a book. Each entry in turn contains meta information about the reference, through tags that specify the kind of meta information such as the author or title. Also, quite commonly, a file will contain shortenings for text fragments that are reused.

To select the desired style the command *\bibliographystyle* is used, for

example *bibliographystyle*{*alpha*}. The style in turn controls the formatting and how the references are labeled, as can be seen in Figure 2.1 and Figure 2.3. On these Figures the labels are different, along with the abbreviation of author names and some of the visual formatting.

To build a LaTeX-document with BibTeX references in it, one first runs the *latex* command (or one of the derivatives) to produce (among other things) an .aux file. The .aux file contains auxiliary information from the LaTeX-compiler. Then the *bibtex* command is run, which uses the .aux file to find the entries in use and to give them labels according to the reference style. The output from BibTeX is a .bbl file with the formatted references, which is then used in subsequent runs of LaTeX, so the document will have labels at the appropriate places and a bibliography in accordance with these labels.

### 2.4.2 Micro-use

Inside a BibTeX file, the format itself is fairly simple. At the main level, we have @*STRING*, @*PREAMBLE*, @*COMMENT* and *entries*. Shortenings for later use in the BibTeX file can be made using @*STRING*, @*PREAMBLE* is for defining how to format the text and @*COMMENT* is for comments and *entries*. The *entries* correspond to the different medium types, such as @*ARTICLE*, @*BOOK* and @*PROCEEDINGS*, which in turn contain the relevant tags for the given entry. Each entry consists of an identifying key and a set of tags. The identifying key will be called 'entry key' to avoid ambiguity with the tag *key*.

For each entry type, there is a specification of tags relevant to the given medium with some of these being mandatory. For instance for an @*ARTICLE* the tags *author*, *title*, *year* and *journal* are mandatory, and supplementary information such as the *pages* for the article and *volume* of the journal can be added. For ease of use, BibTeX provides predefined strings for the months: *jan*, *feb*, *mar* and so on. The rules for the tags are quite simple, a tag can either be 'required' or 'optional'. Furthermore, there are cases where two tags are required, but with an "or" between them, e.g., in the specification of required fields for @*BOOK* it says "author or editor" [34], meaning either author or editor, but not both. These "or" rules will be called 'exclusive'. The last kind of rule is "and/or", which specifies that at least one of the tags has to be present and having both is allowed, this rule will be called 'inclusive'.

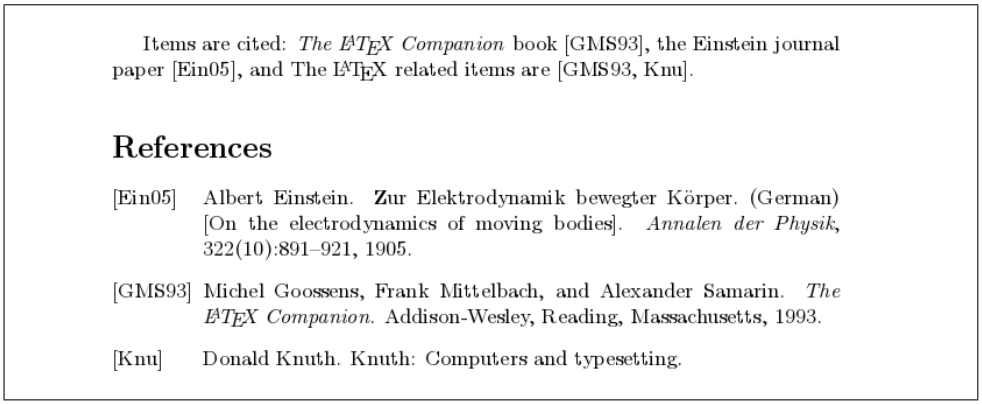
Tags and entries are case insensitive. The literal content (*text*) needs to be enclosed in either { and } or quotes and numbers can be written without. @*STRING* shortenings has to be without quotes and curly brackets. Concatenation of @*STRING* and/or text is done using # [15]. BibTeX is designed to ignore unknown entries and tags, so it allows additional information. An example BibTeX file can be seen in Figure 2.2

```
@String{JFP = "Journal of Functional Programming"}
@String{OUP = "Oxford University Press"}

@Article{abadi1991_substitutions
  author =      "Mart\'{\i}n Abadi and Luca Cardelli
                 and Pierre-Louis Curien
                 and Jean-Jacques L\'evy",
  title =       "Explicit substitutions",
  journal =     JFP,
  year =        1991,
  volume =      1,
  number =      4,
  pages = "375--416",
  note = "A preliminary version was presented at the Seventeenth
                 Annual {ACM} Symposium on Principles
                 of Programming Languages
                 (POPL 1990)"
}

@InBook{leunen1992_handbook,
  author =       "Mary-{C}laire van Leunen",
  title =        "A Handbook for Scholars",
  publisher =    OUP,
  year =         1992,
  pages =        "9--45,154--268"
}
```

**Figure 2.2:** BibTEX example



Items are cited: *The LaTeX Companion* book [GMS93], the Einstein journal paper [Ein05], and The LaTeX related items are [GMS93, Knu].

**References**


[Ein05] Albert Einstein. **Zur** Elektrodynamik bewegter Körper. (German) [On the electrodynamics of moving bodies]. *Annalen der Physik*, 322(10):891–921, 1905.

[GMS93] Michel Goossens, Frank Mittelbach, and Alexander Samarin. *The LaTeX Companion*. Addison-Wesley, Reading, Massachusetts, 1993.

[Knu] Donald Knuth. Knuth: Computers and typesetting.


**Figure 2.3:** BibTeX output using alpha citation style which uses author names and year to index entries. Source: ShareLaTeX [36]

When citing inside a LaTeX file, the desired entry key from the BibTeX is used inside the $\backslash cite$, for example if one has a reference named *some_article* then the reference is used by writing: $\backslash cite\{some\_article\}$. To link the document and the bibliography together, the command $\backslash bibliography$ is used together with a parameter: the name of the BibTeX file, for example: $\backslash bibliography\{mybib\}$.

## 2.5 How BibTeX is used in practice

To this day, BibTeX is routinely used by researchers. Observe that almost all online resources for finding references provide BibTeX entries. Many online databases are widely used to lookup entries, which has also given rise to a variety of ways to identify articles, such as: arXiv numbers, DOI and ISSN.

Since BibTeX is capable of printing only the references used in a LaTeX-document, most people start using BibTeX as a database having a complete file with the references they have used throughout time. People also find it practical to use their BibTeX file as a way to keep track of what they read.

The fact that BibTeX ignores unknown tags is used for de-facto standards, to add additional information and to comment out tags by prefixing them with $OPT$. De-facto standards are so widely in use that they are commonly used in the results from search engines for bibliographic references and there are citing styles that make use of the de-facto standards.

## 2.6 Summary and conclusions

BibTeX is a product of the advent of TeX and the need for managing bibliographic references. The inside of a BibTeX file is simple and intuitive, dividing the entries into types corresponding to the medium it represents and

having tags relevant to that medium. BibTEX has become widely used and given rise to useful de-facto standards and tools to assist with bibliographic references.

Like TEX, BibTEX has been a game changer, and is something for every scientific author to be happy about. So has BibTEX solved the problem of bibliographic references? It would seem so, had it not been for Murphy's Law, as detailed in the next chapter.

# Chapter 3

# The challenges in using BibTEX

*Anything that can go wrong, will go wrong.*
– Murphy's Law

## 3.1 Introduction

The goal of this chapter is to introduce the challenges of BibTEX through concrete examples. The terms of discourse, structural and conjunctural, are first defined (Section 3.2). The rest of the chapter is dedicated to problems with: duplicate entries (Section 3.3), spelling errors (Section 3.4), covering the issues of initials (Section 3.6), online lookups (Section 3.7), name changes of forums (Section 3.8), de-facto standards and conformity to the BibTEX specification (Section 3.9), journal abbreviations (Section 3.10), BibTEX strings that end up in the text (Section 3.11), inconsistent tags (Section 3.12), and inconsistent entry keys (Section 3.13) .

## 3.2 Structural vs. conjunctural issues

Unfortunately, even though BibTEX has made life a lot simpler for scientific authors, it is far from perfect. Inspired by economics, the challenges in BibTEX can be divided into *structural* and *conjunctural* issues.

- The *structural* issues are the ones intrinsic to BibTEX: they are caused by its design, its standard tools, and missing information in BibTEX files.
- The *conjunctural* issues are the combination of circumstances, for instance if the source used does not contain complete information (e.g., extracting a reference from an article where the authors only have

initials even though their full name is known) or the users having other practical priorities than assuring sound bibliographic references in their scientific writings.

Whether an issue is seen as conjunctural or structural is in part a matter of opinion, as the definitions can be stretched in either direction – an issue also faced in economics. Most of the issues are arguably a combination of the two: as they could be fixed by careful labor or by having the right tools available. For example, most bibliography managers have tools to switch between abbreviated and full journal names.

Addressing the human factor, i.e., a conjunctural solution, is one theoretically possible way for solving these issues. One option would be to "simply" motivate people to do things right. Alas, people are not machines and thus this approach will be impossible in practice: for most people, the interest is not the tools they use, but what they use them for. The interest in BibTEX is to ensure that their documents contain appropriately many relevant references.

Since a conjunctural solution is not realistic, the goal is to provide a structural solution to the issues. In the perfect world, the structural solution will be so complete that any issues left will be entirely conjunctural, because one is either not using or misusing the structural solution.

## 3.3 Duplicate content

When a group of authors is writing on the same document, work is often divided and each author has their own .bib file that they use for their references. All might be fine until the group combines their document, and it turns out that multiple authors has used the same reference, but with different keys, ending up having duplicate entries in the bibliography.

Arguably it can be seen as a structural or a conjunctural problem. The similarity of such entries will make them easier to spot with the naked eye, arguably making a conjunctural solution relevant. However due to the similarity (at least if abbreviated the same way), a structural solution is arguably also easy to make. Making the problem purely conjunctural would require a way to detect and merge these entries. Furthermore, if the entries use different key names, the corresponding document should also be corrected. A challenge in this case is when similar entries, that are still different entries, occur.

A similar issue happened in one of the drafts for this document: when an author who wants to create a new entry copy and pastes an existing entry, then giving the new entry an entry key, with intent to adjust the contents. This issue will also result in a duplicate entry, but unlike the situation above, the duplicate should not be merged but contain different content.

Yet another, but similar issue, comes from duplicated values. For instance, when one cites multiple references from inside a conference proceedings, multiple @*INPROCEEDINGS* entries can contain the same value for *booktitle*, and probably other meta-information can occur as well. Furthermore, if one cites multiple publications from the same forum, then the name of the forum, such as the journal name for articles, can be repeated. In general, any textual content can potentially be repeated.

Having a tool that just merges duplicate entries may provide a structural solution to the first case but will create a structural issue in the second as it will automatically introduce new issues. Thus the solution should not merge automatically.

## 3.4 Spelling errors in general

In almost, if not all, cases where people type text, spelling errors will arise, sooner or later, BibTEX files are no exception. Extracting meta-information automatically from documents may also run into a bad extraction leading to spelling errors. Furthermore, a spelling error might not even be in the BibTEX document, as the misspelling can be from the source, for instance if a title of a document is misspelled, then the correct BibTEX will have to contain this error, as it is the title of the source.

A user typing the entries manually is arguably the cause of spelling errors. Thus it can be argued that the issue is conjunctural, since one could do a spell check. However, one might also argue that BibTEX should do a spell check. In the case where automatic extraction is the source of the spelling error the tool will give a structural component to the issue. Having a way of ensuring a spell check and if such a way could ensure that the spelling error corresponds to the source would make this issue conjunctural.

## 3.5 Spelling errors in names

Using a spellchecker works for most entry tags, except names. One citing a name might get the name wrong and write “Rene Rødhof Hansen” instead of “René Rydhof Hansen”. In this misspelling the spellchecker will be of no help. The spellchecker used in this document suggests changing “René” to “Rene”, which would introduce a mistake. Using a spellchecker on names is likely to be the cause of more misspellings than corrections, and would thus cause a structural issue. Providing a structural solution to misspelled names would require either a spellchecker that works for names or some database containing valid names.

```
@article{alurresults,
  title={Results and Analysis of SyGuS-Comp'15},
  author={Alur, Rajeev and Fisman, Dana and Singh, Rishabh
          and Solar-Lezama, Armando}
}
```

**Figure 3.1:** Bad result from Google Scholar

## 3.6 Initials

Another issue in Figure 3.4 is the list of author names that are so heavily abbreviated to initials that one cannot realistically distinguish who the authors are. This might originate from the used citation style or from a resource where they are already abbreviated.

If the initials come from the citation style, the issue is of a structural nature. However, if it is copied from another source, or just written like that, it is conjunctural, as the user did not ensure full names. A structural variant of the copying issue is if the reference is copied by a tool, then it can be argued that such a tool should try to detect initials. Having a way to detect the initials and the full names will provide a structural solution. However, to make an entirely structural solution a reliable way to know when the initials are deliberate or not is needed.

## 3.7 Online lookups

Many writers use online lookups for their bibliographic references. In the Utopic case, all entries can be found online at all times. Even though the databases out there are really good, erroneous results can be found. A lookup on Google Scholar in the beginning of February 2016 for: "Results and Analysis of SyGuS-Comp'15" can be seen in Figure 3.1, which contains an erroneous output.

Having found the article originally on arXiv.org the source of the article is known to be EPTCS - Electronic Proceedings in Theoretical Computer Science. So not only does the Google Scholar result actually not conform to the requirements of an article, the resource is in fact not an article at all, but in the proceedings to a conference. Finding the correct entry details at the EPTCS page reveals the entry in Figure 3.2. Relying blindly on these being correct causes a structural issue since the tools could automatically introduce new errors.

The entries in those search engines cannot account for unpublished work either, and expecting all published work to be represented in the databases would be naive. Having a reliable way to ensure that all entries are present, however, is not a likely scenario.

```
@Inproceedings{EPTCS202.3,
  author    = "Alur, Rajeev and Fisman, Dana  and Singh, Rishabh
               and Solar-Lezama, Armando",
  year      = "2016",
  title     = "Results and Analysis of SyGuS-Comp'15",
  editor    = "\v{C}ern\'y, Pavol and Kuncak, Viktor
               and Parthasarathy, Madhusudan"b,
  booktitle = "{\rm Proceedings Fourth Workshop on}
               Synthesis,
               {\rm San Francisco, CA, USA, 18th July 2015}",
  series    = "Electronic Proceedings in
               Theoretical Computer Science",
  volume    = "202",
  publisher = "Open Publishing Association",
  pages     = "3-26",
  doi       = "10.4204/EPTCS.202.3",
}
```

**Figure 3.2:** Correct lookup on EPTCS, after failed lookup on Google Scholar

Apart from the desire to detect erroneous entries from bad lookups, using lookups for suggestions to most of the other issues can be a useful part of a structural solution. An optimal structural solution to the bad lookups would be if it was possible to detect the users' intended result. As there is no way to know for certain what the user intends, this is an Utopian idea. A realistic idea would be to provide a solution that limits the risk of bad lookups, which is only a partial structural solution since the user would still have to verify the result.

## 3.8 Name changes of forums

In Figure 3.10, spotting the consistency issues is relatively simple. When looking at Figure 3.3 it can be seen that the conference name is slightly different in one of the entries, but so close that they are probably the same conference.

A visit to the homepage of the conference reveals that "National Information Systems Security Conference" used to be named "National Computer Security Conference", which is probably the reason for the `{NIST}-{NCSC}` part of the first entry [38]. In the same source it turns out that there are also references to the old conference name, as seen in Figure 3.4, so to correctly identify potential inconsistencies, it should also recognize name changes and variations.

In BibTEX, there is no way of specifying that the same conference has different names. Therefore, there is a big structural part as there is no

```
\bibitem{stanifordchen96grids}
S.~S.-C. \emph{et al}.
\newblock {GrIDS} -- {A} graph-based intrusion detection system
          for large networks.
\newblock In {\em Proceedings of the 19th
          National Information Systems Security Conference},
          1996.

[...]

\bibitem{porras97emerald}
P.~A. Porras and P.~G. Neumann.
\newblock {EMERALD}: Event monitoring enabling responses
          to anomalous live disturbances.
\newblock In {\em Proc. 20th {NIST}-{NCSC}
          National Information Systems Security Conference},
          pages 353--365, 1997.
```

**Figure 3.3:** Inconsistent reference to the conference and heavily abbreviated author names

```
\bibitem{snapp91dids}
S.~R.~S. \emph{et al}.
\newblock {DIDS} (distributed intrusion detection system) -
          motivation, architecture, and an early prototype.
\newblock In {\em Proceedings of the 14th
          National Computer Security Conference},
          pages 167--176, Washington, DC, 1991.
```

**Figure 3.4:** Name change of a conference

```
@article{Acatrinei2003,
author = {Acatrinei, Alice I and Browne, D and Losovyj, Y B
          and Young, D P and Moldovan, M and Chan, Julia Y
          and Sprunger, P T and Kurtz, Richard L},
doi = {10.1088/0953-8984/15/33/101},
file = {:C$\backslash$:/Users/[...]pdf},
issn = {0953-8984},
journal = {Journal of Physics: Condensed Matter},
month = {aug},
number = {33},
pages = {L511--L517},
title = {{Angle-resolved photoemission study
          and first-principles calculation
          of the electronic structure of LaSb 2}},
url = {http://iopscience.iop.org/[...]},
volume = {15},
year = {2003}
}
```

**Figure 3.5:** Output from Mendeley containing additional information

support for identifying this issue. Furthermore, the owner of the BibTEX file might not even be aware of the issue, which arguably could be conjunctural, since the user could do his research, or structural, because the user should have a tool that can assist him. If a tool could reliably detect inconsistencies from the same forum with different names, the name changes would become conjunctural.

## 3.9 De-facto standards and specification conformity

An interesting point is that not all the structural issues are bad. There are practical ways to use the relaxed properties of BibTEX. For instance BibTEX ignores unknown tags by design which is useful in de-facto standards such as commenting entries out by prefixing with *OPT* or adding information that is not a part of the BibTEX specification, such as ISSN and DOI. The `crossref` tag is technically not specified in BibTEX, but is still part of the tool. In Figure 3.5, an example is provided by a PhD student from the Chemistry Department at Aarhus University. This example is created from Mendeley (see Section 5.4.1) and shows a lot of additional information about the article.

This design choice is an issue, if strict conformity to the specification is desired, since de-facto standards are practical and widely used, strict validation would be counterproductive. Some formatting styles make use of unspecified tags, as can be seen in Figure 3.7. It is interesting to find tags

that are not desired, i.e., tags that do not conform to the specification and the de-facto standards.

Another point is that some values can be clearly correct or wrong. For instance, if one were to write "spaghetti" for the value of the year. Having numbers in a name, or letters in an ISBN is other examples that are likely to be wrong.

## 3.10 Journal abbreviations

Most, if not all, journals require that journal names should be abbreviated when publishing, especially those from competing publishers. However, internally in the BibTEX file the owner's personal priorities are: consistent and correct naming. As BibTEX can be seen as a database of references, it makes sense to consider full names as correct and the abbreviations to be a matter of formatting. Unfortunately, BibTEX does not handle abbreviations at all, which for instance is apparent in articles from arXiv.org, as can be seen in the bbl output in Figure 3.6.

From the point of view that the style of BibTEX should format abbreviations properly, the issue is structural. In cases where the abbreviation is wrong (e.g., due to a typo), the issue moves towards being conjunctural, unless some kind of abbreviation specific spellchecker is being used. Using full names and then formatting them accordingly is the most sensible idea, since the style of abbreviation could be interchanged, should the need arise: it is more readable and it would create better conditions for output tools to provide consistent formatting.

Currently there are multiple strategies for ensuring consistency in abbreviations: some do a search and replace on the .bib file. A bit more structured one can use the strings in BibTEX to ensure a consistent naming of a journal which can further be combined with the usage of crossref. Another approach is the use of BibLATEX and biber Section 5.3.3, which provide the solution in the formatting options [1], provided that the abbreviation handling of the style is correct. This solution causes the formatting issue to become conjunctural.

Bibliography managers (see Section 5.4) tend to go with the strategy of storing the references using full names. When one using a bibliography manager export to a BibTEX file (or another format), the desired abbreviation style is applied to the export. This strategy moves the issue towards being conjunctural, for the same reasons as the BibLATEX and biber solution.

As the purpose is to work on the BibTEX files, the formatting in the end is technically not the primary concern. Optimally the concern is to ensure a consistent document. Since BibTEX styles do not take care of abbreviations, there is a need for considerations on how to deal with consistency and to ensure the desired style of abbreviations. A partially structural solution

```
\bibitem[\protect\citename{Baroni \bgroup et al.\egroup }2014b]
          {baroni2014don}
          Marco Baroni, Georgiana Dinu, and Germ{\'a}n Kruszewski.
\newblock 2014b.
\newblock Don't count, predict!
          a systematic comparison of context-counting vs.
          context-predicting semantic vectors.
\newblock In {\em Proceedings of the 52nd Annual Meeting of
          the Association for Computational Linguistics},
          volume~1, pages 238--247.

\bibitem[\protect\citename{Bruni \bgroup et al.\egroup}2014]
          {bruni2014multimodal}
          Elia Bruni, Nam-Khanh Tran, and Marco Baroni.
\newblock 2014.
\newblock Multimodal distributional semantics.
\newblock {\em J. Artif. Intell. Res. (JAIR)}, 49:1--47.

[...]

\bibitem[\protect\citename{Collobert \bgroup et al.\egroup}2011]
          {collobert2011natural}
          Ronan Collobert, Jason Weston, L{\'e}on Bottou,
          Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa.
\newblock 2011.
\newblock Natural language processing (almost) from scratch.
\newblock {\em The Journal of Machine Learning Research},
          12:2493--2537.

[...]

\bibitem[\protect\citename{Kalchbrenner \bgroup et al.\egroup}2014]
          {kalchbrenner2014convolutional}
          Nal Kalchbrenner, Edward Grefenstette, and Phil Blunsom.
\newblock 2014.
\newblock A convolutional neural network for modelling sentences.
\newblock In {\em Proceedings of EMNLP}.
```

**Figure 3.6:** Inconsistent naming of journal and conference names

ensures a consistent structure, which can be modified according to the desired style of abbreviations. In a fully structural solution, the styles handle abbreviations (such as the BibLaTeX and biber do).

## 3.11 BibTeX strings ending up as text

When working with bibliographic references, BibTeX strings (@*STRING*) can accidentally end up being inside textual content. For instance if one, as the PhD student earlier, exports from a program that does not make use of the strings. In the output from the aforementioned student seen in Figure 3.5 the month is actually a text and not a string as one would expect. The use of text over strings prevents re-use of fragments and localization. Another similar issue is, when the text, or part of the text, should have been moved to a string, because an appropriate string has been introduced.

When the usage as text over strings comes from a tool, such as in the example above, the issue is of a structural nature. When one enters a string as text by accident, it is again arguable whether the issue is considered structural or conjunctural. Providing a structural solution to this would be by being able to detect this and correct it.

## 3.12 Inconsistent tags

Take the inconsistency in Figure 3.7, found in an article on arXiv.org: two references from the same conference, but with different years. The inconsistency is easy to identify due to the consistent content. Correct and consistent content will help tools in detecting inconsistencies. This exposes a structural part of the issue, as no such tools exist (to the authors' knowledge).

The ISSN might not exist for the `MuKoJansson2009AoPA`-entry. In this case a structural detection system might be the cause of new structural issue, either the removal of relevant data or forcing entries when the data does not exist. In this specific case, the search result in Figure 3.8 reveals that the missing ISSN does exist and thus a tool pointing out the inconsistency would in this case make the issue conjunctural.

Provided a reliable way to lookup correct entries, a tool could move the issues towards being conjunctural. Take the inconsistency in Figure 3.9 where two entries from the same conference have different information. One has an additional "ICFP '10" and "ACM" in there, the other one does not.

An online search for BibTeX information gives the entries in Figure 3.10 for the two articles, which provides one possible option for a set of consistent entries. As can be seen, *ACM* is the name of the organization and is probably missing in the original BibTeX that produced the .bbl file inspected above. The Danielsson has additional information with the content "ICFP '10", which is not apparent in the search result.

```
\bibitem[Bernardy and Claessen(2015)]{bernardy_efficient_2015}
J.-P. Bernardy and K.~Claessen.
\newblock Efficient parallel and incremental parsing
          of practical context-free languages.
\newblock \emph{J. of Funct. Prog.}, 25, 2015.
\newblock ISSN 1469-7653.
\newblock \doi{10.1017/S0956796815000131}.

[...]

\bibitem[Mu et~al.(2009)Mu, Ko, and Jansson]{MuKoJansson2009AoPA}
S.-C. Mu, H.-S. Ko, and P.~Jansson.
\newblock Algebra of programming in {Agda}:
          dependent types for relational program derivation.
\newblock \emph{J. Funct. Program.}, 19:\penalty0 545--579, 2009.
\newblock \doi{10.1017/S0956796809007345}.
```

**Figure 3.7:** Additional tag ISSN is provided in one of the entries

```
@article{Mu:2009:APA:1630623.1630627,
 author = {Mu, Shin-cheng and Ko, Hsiang-shang
           and Jansson, Patrik},
 title = {Algebra of Programming in Agda:
          Dependent Types for Relational Program Derivation},
 journal = {J. Funct. Program.},
 issue_date = {September 2009},
 volume = {19},
 number = {5},
 month = sep,
 year = {2009},
 issn = {0956-7968},
 pages = {545--579},
 numpages = {35},
 url = {http://dx.doi.org/10.1017/S0956796809007345},
 doi = {10.1017/S0956796809007345},
 acmid = {1630627},
 publisher = {Cambridge University Press},
 address = {New York, NY, USA},
 }
```

**Figure 3.8:** Search revealing the ISSN

```
\bibitem[Bernardy and Claessen(2013)]{bernardy_efficient_2013}
J.-P. Bernardy and K.~Claessen.
\newblock Efficient divide-and-conquer parsing
          of practical context-free languages.
\newblock In \emph{Proc. of ICFP 2013}, pages 111--122, 2013.

[...]

\bibitem[Danielsson(2010)]{danielsson_total_2010}
N.~A. Danielsson.
\newblock Total parser combinators.
\newblock In \emph{Proc. of ICFP 2010}, ICFP '10,
          pages 285--296. ACM, 2010.
```

**Figure 3.9:** Capt

```
@inproceedings{bernardy2013efficient,
  title={Efficient divide-and-conquer parsing
         of practical context-free languages},
  author={Bernardy, Jean-Philippe and Claessen, Koen},
  booktitle={ACM SIGPLAN Notices},
  volume={48},
  number={9},
  pages={111--122},
  year={2013},
  organization={ACM}
}

@inproceedings{danielsson2010total,
  title={Total parser combinators},
  author={Danielsson, Nils Anders},
  booktitle={ACM Sigplan Notices},
  volume={45},
  number={9},
  pages={285--296},
  year={2010},
  organization={ACM}
}
```

**Figure 3.10:** Scholar lookup

## 3.13 Inconsistent entry keys

The naming scheme for entry keys may vary throughout a BibTEX file. For example, one of the users collaborating earlier might use entries from various online databases getting keys corresponding to Figure 3.10, Figure 3.2 and other structures in one big mess. One of the users collaborating might also be new to using BibTEX (could be a student learning) and needs to find a nice and consistent way of writing the keys.

A challenge could be to avoid duplicate key names, which with a consistent structure is more likely. The duplicate entry keys can have the advantage that it can be an indicator of a duplicate entry Section 3.7.

Inconsistencies in keys might range from not a problem at all to fully conjectural or structural, highly depending on one's point of view. A user may simply not care and apart from the potential to detect duplicate entries, may not feel a reason to. In the collaboration scenario, the additional way of detecting duplicates may be valuable. In this case, it arguably becomes conjunctural since the authors should have agreed on a style - which might also have helped a newcomer to acquire a good practice. It can also be argued that it is structural since BibTEX does not provide naming guidelines nor tools for ensuring a guideline.

To make this issue structural, a naming convention would have to be specified and some way of detecting deviations provided. A blind application of this method could result in two entries colliding with the same key, requiring the attention of the user.

## 3.14 Summary and conclusions

BibTEX has a lot of structural issues:

- Duplicate entries are not desired. A structural solution will be to detect and merge duplicates, because a duplicate originating from missing or wrong data automatically merging may cause issues.
- Spelling errors are frowned upon. A structural solution will be able to find and correct the spelling errors, preferably by having the same spelling as published. Alternatively, running a spell checker and having a way to account for deliberate misspellings, domain specific words and choice of language.
- Spelling errors in proper names will challenge normal spellcheckers. A structural solution is a spellchecker that works for names or a database with names to verify names.
- Use of initials in proper names can make it hard to identify an author. A structural solution is to detect initials and the full names, accounting

for when the initials are deliberate.

- Online lookups can contain wrong data. Optimally we can detect these wrong data reliably: the most realistic idea is to provide a way that limits the likelihood of erroneous results.

- Name change of a forum can affect detection of inconsistent tag use. A structural solution will have some way of detecting when sources are from the same forum.

- Conformity to de-facto standards and the BibTEX specification is desired, but should not prevent new de-facto standards. A structural solution requires something that checks the conformity to a combination of specifications and de-facto standards, that can account for changes in de-facto standards.

- Journal abbreviations can be an issue for analyzing tools and having full names in BibTEX files. A structural solution will be able to ensure that all entries are consistently abbreviated or de-abbreviated, preferably accounting for the fact that people may want to switch between for formats.

- BibTEX strings that end up as part of the text, can result in wrong data. Also text that is contained in a string and should have been referenced to the string, rather than written as raw text, can cause errors if the content is updated. A structural solution will be able to detect and make text into strings when a string is desired. Furthermore, a structural solution will be able to detect when some of the textual content is stored in a string.

- Inconsistent tag usage in similar entries causes messy bibliographies. A structural solution will detect these inconsistencies and will be able to suggest a course of action. This solution may be challenged by deviations in information and forums that change name.

- Inconsistent entry keys can be an issue in collaboration and may make it harder to detect duplicates. A structural solution could be to apply a naming scheme for the entry keys, accounting for similar entries that would result in identical entry keys.

With all these problems at hand, BibTEX may not seem like the optimal tool after all. What can be done with such a range of issues? Approaches to the issues are covered in the next chapter.

# Chapter 4

# Our approach to BibTEX

*We are all slaves to our histories.*
*If there is to be a... bright future,*
*we must learn to break those chains.*
– Delenn, Babylon 5

## 4.1 Introduction

The goal of this chapter is to organize the issues people have with BibTEX: identifying the issues in BibTEX are (Section 4.2), and how to approach them (Section 4.3).

## 4.2 What are the issues in BibTEX

BibTEX has changed the landscape for scientific writing. However, it is not without any issues. As seen in Chapter 3, the challenges range widely and trying to group similar looking issues we have:

The misspellings in general, misspellings in names, initials in author names, erroneous online lookups, use of abbreviations for journal names, and BibTEX strings that end up being text will be considered as lexical concerns. Conforming to the specification and de-facto standards is considered as a correctness concern. All of these issues will be considered combined as 'correctness and lexical concerns'.

Duplicated values, forum names that change, inconsistent use of tags and inconsistent entry keys will be considered as 'consistency concerns'.

The Utopian goal is to provide a structural solution for all these issues so that if any further issues exist, they would be purely conjunctural.

## 4.3 What can be done about the BibTEX issues

As previously stated (Section 3.2), we wish for a structural approach to the BibTEX issues reviewed in Section 4.2. This choice means to ensure that there are tools that are able to handle the issues - preferably to the level where all issues are solved. As touched upon shortly when inspecting the problems, it is not likely that all issues can be solved perfectly. For a structural solution, there are two approaches: Updating or replacing BibTEX (Section 4.3.1) and Augmenting BibTEX (Section 4.3.2)

### 4.3.1 Updating or replacing BibTEX

One way of handling the issues structurally would be to change or replace BibTEX, so it handles all lexical and consistency concerns. This way would include changing the BibTEX specification to account for relevant de facto standards, enforcing conformity, handling abbreviations and controlling all data. The updated version of BibTEX could then either correct the issues when running into them or fail building the .bbl file with appropriate error messages for issues that the user needs to take care of.

This approach would probably be perceived as invasive since it would cause existing BibTEX files not to work, and it would impose requirements one may not desire. The perception would of course depend on perspective because the user who wants structure and control might find it good that it is enforced.[1] Updating or replacing BibTEX to handle the issues does not ensure separation of concerns.

As described in Chapter 5 there are a few attempts at both changing and replacing BibTEX.

### 4.3.2 Augmenting BibTEX

Instead of changing or replacing BibTEX, an augmenting tool is another option. Such a tool, together with BibTEX, would provide or suggest improvements, instead of changing specifications. An augmenting tool will be a supplement to current use of BibTEX and be optional, rather than imposed on the users.

An augmenting tool also has the advantage of separating concerns. The concern of BibTEX is to cite references and print a bibliography according to a bibliography style. Where as the concern of an augmenting tool is to ensure that ones has a nice .bib file.

Ironically, getting people to use such a solution causes a new conjunctural issue, to which there is no structural solution.

[1] A similar debate exists for statically vs. dynamically typed programming languages.

## 4.4 How do we approach the BibTEX issues

### 4.4.1 Introduction

The goal of this section is to introduce our choice of solutions for the issues reviewed in Section 4.2.

### 4.4.2 Lexical and correctness concerns vs. consistency concerns

The relation between the lexical and correctness concerns and the consistency concerns reveals a dependency in the analysis. Going through the consistency concerns observing their relation gives:

- For inconsistent use of entry tags, it is needed to have a way to determine if entries are from the same forum. Such a way depends on consistent naming of the forum and a way to detect name changes.
- For duplicate entries having unique identifiers such as arXiv numbers, ISSN or DOI will make the detection trivial. Otherwise, the detection has to be based on the similarity of the information: at best the information is identical, otherwise it has to be as similar as possible to improve the detection. For repeated content, detecting when there are multiple instances of $@INBOOK$, $INCOLLECTION$ and $INPROCEEDINGS$ that reference the same source is needed. For textual content reused content, such as a journal name, should be detected. For textual values it should compare the values that are likely to be repetitive. Thus solving the lexical concerns will be of use.
- Inconsistent naming of entry keys can be handled by a naming scheme. Such a naming scheme is usually based on the information in the entries. So having the relevant tags and correct content in them will provide a way to ensure consistent entry keys.
- For name changes of forums, we need to be able to recognize the names, which is easier with correct and consistent names.

A common property about the consistency concerns is that they are easier to handle, once the lexical and correctness concerns have been handled. This property indicates that a twophase solution may be desired: first handling the lexical and correctness concerns, then handling the consistency concerns.

### 4.4.3 Duplicate content

Duplicate entries are fairly straightforward if the tags and the contents are identical. If the content and tags deviate, a way to detect “similarity” will be

needed. The easiest definition of similarity is if the title and author is identical. However, this definition might need to take things like revisions and year into account, as an author may decide to write a new version later or if one for some reason desires to refer to different revisions. Further challenges may arise if there are lexical and correctness issues. As per Section 4.4.2, one should fix these issues first.

For duplicate values, two ways of detecting are needed. Provided the correctness of the values, detecting duplicated meta-data is done by corresponding the *author/editor* and *title* for $@INCOLLECTION$, $@INBOOK$ and $@BOOK$, for conference proceedings comparing the *title* and *booktitle* of $@PROCEEDINGS$ and $@INPROCEEDINGS$ respectively will determine if the conference is the same,

### 4.4.4 Spelling errors in general

To detect misspellings, a spellchecker can be used. Alternatively, checking the resources in online databases is an option. If a spellchecker is used, one should be aware of false positives. Domain-specific terms might not be present in the dictionary and if the original source is misspelled, it would be a mistake to correct it (once published, the name published is the correct name of the reference!). An issue with the published version is that the title can be updated. An example of an updated name is from the International Conference on Functional Programming, where an error in the proceedings caused an article to be printed with the title: "Types, potency, and impotency: why[...]", where the intended content naturally was "Types, potency, and idempotency: why[...]". The misprinted version of "idempotency" has been corrected in online resources, such as DBLP [10]. Both versions of the title are arguably correct.

Provided a solution for the issues in online lookups, the correct spelling will be a matter of looking up, but references may not be in the databases. As stated in Section 4.4.7, there is no good way to ensure correct lookups. Therefore a spellchecker seems like a good way to get an indication of possible misspellings, but one would still have to verify them. Since entries may be in different languages, a way of specifying the language should be considered.

### 4.4.5 Spelling errors in names

Spell checking names with a normal spellchecker is likely to cause false positives. A possible way to approach these false positives would be to widen the online search to databases with scientific authors, such as DBLP and Google Scholar. This widening, however, will not help for authors who are not in these other databases. Extending this solution to contain more databases such as book authors will improve the solution, but will still be limited to

known author names. Another issue is that some authors use different versions of their name depending on the context. For example, Iain Menzies Banks wrote both mainstream fiction and science fiction, writing under the name Iain Banks for mainstream fiction and using Iain M. Banks for science fiction [9].

### 4.4.6 Initials

Finding the initials is a matter of being able to detect single letters with or without a period after it. However, if one for some reason groups initials together, e.g., making George R. R. Martin into George RR Martin, or J. K. Rowling into JK Rowling, or B. B. King into BB King, then further detection will be needed. Replacing the initials with full names is appropriate whenever possible, but since the full names may not be known, some way of specifying that initials are the only thing available is needed. The best approach will probably be the one described for spell checking names in Section 4.4.5.

### 4.4.7 Online lookups

Online database lookups can be a very useful tool for handling the lexical and correctness concerns, but getting incorrect data can cause problems. The best approach would be to ensure correct lookups, by using services that are known to be correct.

A situation where relatively reliable lookups is possible, as in the "EPTCS" lookup seen in Figure 3.2, can be used to improve the reliability of the results. However, there is still no certain way to know if the database of the service is correct, so it is still not certain. Most likely the ID systems, such as arXiv numbers, DOI and ISSN, will also provide a relatively reliable lookup mechanism, but that is still not guaranteed.

Another way to approach the bad lookups could be by doing the same lookup in multiple databases and then have some kind of voting system that decides on which entry to trust. This would however require knowledge of which information sources each online service uses, because their source of information may be the same and then the same error could get multiple votes. The approach can be refined by having increased trust in databases that are likely to be correct.

The most appropriate strategy is probably selecting the database most likely to be correct and then have the user select if one agrees with the result. Doing the vote system would be overkill in most situations, and the user would still have to validate the result, since the voting system will not provide a certain correct result. Having the user validate the result will make issues partly conjunctural, if one just accepts any result from the lookup.

### 4.4.8 Name changes of forums

Handling name changes of forums is supportive to ensuring consistency. Since name changes cannot be derived automatically, one approach would be a database of known name changes, which has the disadvantage that it needs to be maintained. Adding a configuration to specify name changes may also be appropriate.

### 4.4.9 De-facto standards and specification conformity

As stated in Section 3.9 it is desired to be able to validate if the file conforms to the specification and the desired de-facto standards. Validating conformity to the specification is a simple task, as the specification is just a set of rules (Section 2.4.2). A set of de-facto standards, likewise, is also a simple set of rules.

De-facto standards however, provide challenges, as they are the standards currently in use. This means that they both depend on who the user is and the standards are subject to change.

A tool handling conformity to the specification and de-facto standards should thus be configurable to account for changes in de-facto standards. For practicality, the de-facto standards that are not likely to change (such as ISSN and DOI) could be accounted for with a default setting.

Validating the values will improve this solution further. Having rules for valid values whenever possible. Some rules will be very easy to set up, e.g., ISBN consists only of numbers and dashes. Other rules such as not allowing numbers inside a name might be able to become an issue, e.g., if the queen of Denmark were to write something that is later cited, a correct name for the author would be “Dronning Magrethe 2.”. A structural solution will validate as much as possible, but too rigid validation has the potential of causing new structural issues.

### 4.4.10 Journal abbreviations

Ensuring a consistent use of either abbreviations or full names is desired. From the point of having the information in a complete version converting full names, ‘de-abbreviating’ is desired. Using a database of standard abbreviations for forums will be useful to de-abbreviate. Taking care of a consistent way to switch between full names and abbreviations is also desired. Making use of strings to handle the switching between full names and abbreviations is probably the best approach since this will keep it clear which forum is which. This also allows the user to use string names that are: full names, official names or their own style of abbreviations, to their choice.

### 4.4.11 BibTEX strings ending up as text

A BibTEX string can end up being part of a text by mistake: in the example used in Section 3.11 where the month ended up as a text rather than a string a simple check is possible, because for a month we know what to expect. Whenever something in the middle of the text should have been a string, the text would have to be checked for potential strings. Automatically correcting it would introduce a potential source of errors, because a text being identical to a string name could just be a coincidence, so it has to be the user's choice.

### 4.4.12 Inconsistent tags

Detecting inconsistent use of tags requires a way of detecting when entries are from the same forum. When such a way is provided, it is possible to check if the set of tags are the same. Having some kind of statistics on the usage may further improve the feedback, since it will be possible to suggest the shortest path to consistency, whether by adding or by removing tags.

Since a lot of the forums are continuous, such as a conference being held each year, the detail level of the information may change over time. Also, in some cases, a single item can have additional information that is not general to the forum, or for some reason not have information according to the general standard. Optimally, there should be a way to account for these cases, either by the user enforcing conformity, or having options for when deviations are desired.

### 4.4.13 Inconsistent entry keys

Provided that the lexical and correctness concerns have been solved, handling inconsistent entry keys require very little effort. Having a rule for how the key names should be formatted is all that is needed. Like in Section 4.4.3 there is the issue of similar, but different entries. Similar entries could result in the same entry key, so there is the need for a way to disambiguate the key names. Since a lot of users already have databases in use, support for one specifying a naming scheme would be appropriate.

## 4.5 Summary and conclusions

The issues in BibTEX files have been grouped in correctness and lexical concerns and in consistency concerns. Updating or replacing BibTEX was compared to augmenting BibTEX. It was observed that the consistency concerns in general depend on the solution of the correctness and lexical concerns.

- Duplicate entries can be found if there is a unique identifier or identical entries. To detect deviating duplicates, a definition of "similarity" will

be needed.

- Spelling errors in general can be solved by a spellchecker and the usage of online lookups. For a spellchecker, one must be aware of false positives and language.
- Spelling errors in names will challenge normal spellcheckers. Using online databases of authors will enable some checking, but the solution will be limited to known author names.
- Initials hiding people's names can be handled by online resources, just as misspellings in names.
- To get online lookups that contain the correct data is impossible, however, the results can be improved by selecting the most appropriate database for the lookup and by introducing detection of erroneous lookups.
- Name change of a forum can be handled by a database and by a way to specify name changes.
- Conformity to de-facto standards and the BibTEX specification should be handled by checking the rules for the standards and being able to specify the desired de-facto standards.
- Journal abbreviations should be moved into strings and be consistently de-abbreviated or abbreviated.
- BibTEX strings that end up as part of the text should be detected by matching string names that appear in text.
- Inconsistent tags should be detected based on when entries are from the same forum. One should be able to specify deviations from the general set of information for the forum.
- Inconsistent entry keys should be handled by a rule for the desired format for entry key names.

Handling the BibTEX issues will be done by augmenting BibTEX with a tool used in multiple phases. The tool will address the correctness and lexical concerns first, then the consistency concerns.

Approaches for handling the issues have been reviewed. However, how are they handled by others? This topic is reviewed in the next chapter.

# Chapter 5

# Related work

## 5.1 Introduction

The goal of this chapter is to review the existing tools for managing bibliographies and how these tools address the issues covered in Chapter 3.

The chapter is organized as follows: an overview of the pure tools for BibTEX (Section 5.2), some of the attempts to replace BibTEX (Section 5.3) and bibliography managers that strive to redesign how to manage bibliographies (Section 5.4).

## 5.2 BibTEX tools

There are a few pure BibTEX tools for handling reference databases. Some of these tools target some of the issues described. The various tools have either been tested or their functionality verified by code inspection.

### 5.2.1 Bibcleaner

Bibcleaner is a tool developed by Joos Buijs for his own use in his PhD studies and made publicly available in case someone can get a use for it. The purpose of the tool is to clean BibTEX files by using online lookups in the DBLP database [5, 6].

Bibcleaner is quite simple: it runs through each entry comparing it with the DBLP database and suggests improvements if it can find the current key in the database. The functionality of the tool is currently limited to computer science, as it only uses DBLP for lookups [5].

### 5.2.2 BibTooL

BibTooL is a BibTEX that was developed by Charalampos Nikolaou from University of Athens. The last version of the tool is from July 2012 and there is no indication of further development being done. BibTooL is a set

of shell scripts to assist in cleaning, organizing and ensuring consistency in BibTEX files [26].

The only method for handling of abbreviations in BibTooL is by DBLP-lookup, like Bibcleaner. However, unlike Bibcleaner, it also provides some tools for correcting inconsistencies. Mostly it is inconsistencies in the formatting, such as trailing spaces and capitalization of the entry types (such as @article) together with a tool for detecting duplicate entries.

### 5.2.3 JabRef

JabRef is an open source project, currently developed and maintained by Jörg Lenhard, Matthias Geiger, Oliver Kopp, Simon Harrer and Stefan Kolb [35]. It is using the pure BibTEX format and has features for automatic key generation, searching online databases and a plugin system [20]. There is a feature for handling Journal Abbreviation using predefined lists of abbreviations [19]. Currently, JabRef does not have any spell checking features, but it is planned for implementation in the next full release [2].

The JabRef team is currently working on a tool named CloudRef which has a focus on a web based interface for collaboration. Currently, the project is in the planning phase and suggested features hint that it might end up correcting some of the issues, as there is some focus on correctly filled BibTEX references and corrections for those.

Looking through the lists of JabRef plugins does not reveal any that provide solutions to the issues of this dissertation. There are plugins for lookups in online databases, which can offer a partial solution (provided the online information is correct and that it is there for the reference) for new entries [21]. Since JabRef uses BibTEX for storage, tools for BibTEX can be used too.

When using JabRef, it is very easy to switch between journal abbreviations and full names, provided that the abbreviation is already known, with the ability to add custom abbreviations [19]. The supposed spell check was not to be found in the beta release of JabRef. Doing lookups online for details can in theory be done during PDF-import using Mr. DLib, which supposedly should search for entries based on similarity to .PDF files. The project seems to be abandoned as the site has not been updated since 2012 [11, 12]. It is worth nothing that JabRef has a system for finding duplicate entries, and does some validation of BibTEX files when opened to ensure valid data.

### 5.2.4 Bibcut

Bibcut is a tool to arrange and modify BibTEX files in ways that will assist scientists. It is developed by Dr. Clemens Barth with the latest release from 2014. Bibcut can be used to clean author strings to ensure consistent

formatting, compare BibTEX files, for instance to find duplicates and ways of handling journal names and change them from abbreviations to full names or vice versa [4].

### 5.2.5 BibTEX Check

BibTEX Check was developed by Fabian Beck to validate if required fields are present, to do consistency checks on conference proceedings and to provide an easy overview in html with links to relevant sources such as Google Scholar and DBLP.

The way BibTEX Check works is by running some very simple checks to find flaws in bibliographies. The checks are things like: check if the value for *author* contains a period(.), to find initials, check that the mandatory fields are present, and check if an entry contains fields from another entry type. For instance, if a $@PROCEEDINGS$ contains the *pages* tag, to suggest that the user might have meant $@INPROCEEDINS$ instead.

## 5.3 BibTEX alternatives

There are both variants of and alternatives to BibTEX. A quick overview of a few of the options is provided. Mostly they do not provide many new options.

### 5.3.1 BibTEXml

BibTEXML is a version of BibTEX where the entries are formatted with XML. It is by design made with strict schemas and conversion tools to convert from BibTEX. Having a system using strict schemas enables enforcing correct entries that conform to the specification. The strict approach will prevent usage of non-conforming tags. On the down side, this approach also makes it hard to introduce new de facto standards when they are needed [16].

### 5.3.2 MLBIBTEX

MLBIBTEX is an variant of BibTEX from Jean-Michael Hufflen to create support for multiple languages. The focus of MLBIBTEX is to add support for entries in different languages, accounting for appropriate naming of language specific content (such as months). This variant has been made to account for authors writing in different languages, such as a lecturer writing course material for students in their native language. It does not seem that MLBIBTEX takes care of the issues that arise from the lack of encoding support in BibTEX [18].

### 5.3.3 BibLaTeX and biber

BibLaTeX is a LaTeX package intended to replace BibTeX, but with backward compatibility with BibTeX. It is designed to be able to run with the BibTeX backend, but defaults to use biber instead. The idea behind BibLaTeX and biber is to overcome some of the limitations of BibTeX, such as the limited character support, limited options for sorting and the need for new entry types (for instance, BibTeX does not have an entry for online resources). Most notably, the BibLaTeX and biber combination adds the support for mappings of the entry content, for instance, to ensure that journal names get abbreviated when printing the bibliography.

## 5.4 Bibliography managers

Bibliography managers, sometimes also called reference managers, are software tools for bibliography management that are not directly targeted at BibTeX. Some of these provide partial solutions for the lexical and consistency concerns. All of the tools have been tested for relevant features that are not described on their respective feature overviews.

### 5.4.1 Mendeley

Mendeley is a bibliography management tool developed by Mendeley Ltd. The focus of the tool is to make it easy to manage and share references. Mendeley provides features such as importing .PDF files and detecting relevant details such as the author, title and DOIs, and updating entries from online databases using, for instance, the DOI. The target group for Mendeley is students and researchers [24].

Like BibTeX, Mendeley does not facilitate any systems for detecting inconsistencies such as spelling errors, revision numbers or initials. Mendeley can download meta-information based on a DOI which in part remedies the lack of these tools. Mendeley does not have any plugin/add on system, so it's unlikely that there are any third party tools to further handle the BibTeX issues. A search through Mendeley's request database reveals the same as the testing and features page. On Mendeleys online request system, there are requests for features such as: spell checking, fixing capitalization inconsistencies and bulk DOI lookups [22, 25, 37, 39].

### 5.4.2 EndNote and RefMan

EndNote and RefMan are also tools for reference mangement, made by Thompson Reuters. RefMan is an older product and Thompson Reuters recommends using EndNote instead [28, 33]. The focus is on: the sharing

capabilities, tools for PDF import and management, online lookup of references and detection of journal abbreviations [29, 32]. Their target group is students and university related groups. EndNote also provides a spell check feature that is not mentioned in the feature sets [17].

The tool for handling abbreviations in EndNote is only used to format the names when exported or used as references. This is done using a mapping from a predefined set of known abbreviations, not fixing the naming issues internally in the files [31].

When trying to remedy the spelling and consistency errors by using DOI lookups, it seems that the only way of doing that will create duplicate entries rather than updating the current one. This is just making the issues even worse, as the references then end up with duplications too. The spellchecker works on a per entry basis and there is no apparent way to spell check an entire bibliography. EndNote provides of extra downloads to provide features such as BibTEX support. However no downloads were found to remedy neither the lexical nor the consistency concerns further [30].

### 5.4.3 Zotero

Zotero is a bibliography manager developed by Roy Rosenzweig Center for History and New Media. It is targeted at people doing research. The organization and collaboration aspects are in focus for Zotero [42].

There is a built in spellchecker for notes written in Zotero. However the spellchecker is not applied for things like titles, which prevents it from detecting those errors. Like JabRef, Zotero has a plugin system. Plugins to remedy the BibTEX issues is neither found using the official list of plugins [14] nor by a normal search engine. There is a limited mechanism for detecting abbreviations in Zotero, in its .jar file there is a list of known abbreviations which it can use to choose the correct abbreviation when exporting. Entering an abbreviation does not seem to correct it to the non-abbreviated version. This list can be manually edited, and overruled [40].

It is possible (but not tested) that Zotero can be used to import a BibTEX file, then export it using the lists of known abbreviations and correct the abbreviations already known. As other tools, such as JabRef and Mendeley, already allow this procedure, we did not test for this feature.

### 5.4.4 RefWorks

RefWorks is a web based bibliography manager developed by ProQuest, meant for students, researchers and librarians. The focus of the tool is to have the simplicity of a web based solution, which makes the resources easy to share and available everywhere. The features include online lookups, statistics systems and collaboration features [27].

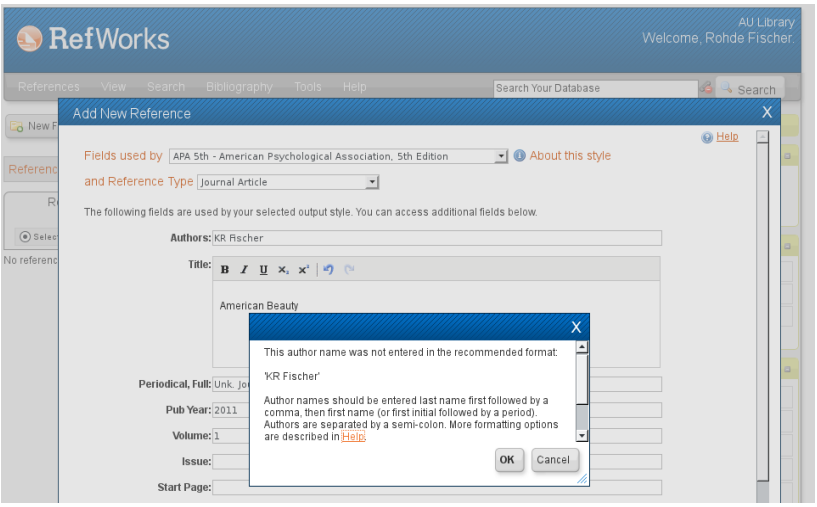



**Figure 5.1:** RefWorks detects some formatting on the name field

As with the other tools, testing shows that the error detection features are limited. In RefWorks, the work flow for adding references is to either enter them manually or to use a browser link called "RefGrab-It" to import from homepages. RefGrab-It can be used for DOI lookups, which enables the same advantages as Mendeey and EndNote. Contrary to the other tools, RefWorks tries to detect if the author name is malformed, although it only seems to detect if one types "John Doe" rather than "Doe, John" as it expects, as can be seen at Figure 5.1.

In RefWorks itself there seems to be no spellchecker, but the spelling errors are in part solved by most modern browsers, because they often include a spellchecker for input fields.

### 5.4.5 Papers

Papers is a bibliography manager developed by Mekentosj B.V., targeted for science and research. It provides a tool set for searching online libraries for references, collaboration and various ways of organizing [8].

It does not highlight any features of relevance [8] and as it is developed for Mac OS, it has not been possible to test if there is relevant features apart from what they mention on their page.

## 5.5 Summary and conclusions

There is a variety of ways to handle bibliographic references. Right from a multitude of tools that strive to assist with maintenance of one's BibTEX library, alternatives to BibTEX and bibliography managers that strive to replace the entire management process. Some of these help by doing online

lookups to double check entries, finding duplicates, assisting in transforming to and from abbreviated forms, enforcing strict rules and spell checking.

Some of these tools do provide partial solutions to the issues with BibTEX, but none of them provide complete solutions and none of them combines the solutions into a more general solution. The pure BibTEX tools can of course be combined. Since BibLATEX is designed to be close to BibTEX, there is a chance that some of the BibTEX tools work on BibLATEX too (which has not been tested). The alternative formats provide some ways to gain strict conformity and to manage abbreviations, but apart from that, not much consolidation on the issues. The reference managers in general focus on the collaboration between authors. Only a few of the tools provide assistance with the issues, such as spell checking and abbreviation handling.

Having seen approaches to the BibTEX issues and having reviewed how others approach them, one may wonder how one can analyze BibTEX files to detect these issues. This is the topic of the next chapter.

# Chapter 6

# Analyzing BibTEX files

## 6.1 Introduction

The goal of this chapter is to show how BibTEX files are analyzed: what should be done about BibTEX in principle and practice (Section 6.2) and an analyzing prototype named Orangutan (Section 6.3).

## 6.2 What should be done about BibTEX

### 6.2.1 In principle

Due to the practical issues in changing or replacing BibTEX and to ensure separation of concerns, an analyzing tool should be an augmenting tool.

To analyze BibTEX, one would need to parse a BibTEX file into a suitable representation. This representation would need to parse BibTEX entries and strings at a minimum. If one intents to 'pretty print', i.e., writing a BibTEX file formatted consistently in human readable way, the result after resolving all issues, the parsing also needs the preambles. Comments are technically optional, but should be kept too.

The easiest approach is a two step parse: first taking care of the lexical and correctness concerns, then the consistency concerns. By taking care of the lexical and correctness concerns first, the optimal conditions for taking care of the consistency concerns is made.

The term database is used for both local and online databases, since it does not matter if one uses a local copy, if it is kept consistent with the online database.

#### First step: Lexical and correctness

- Spelling errors in general can be detected either by online lookups or a spellchecker. A combination of the two might be an improvement, since it gives a potential indicator of false positives. Using a

spellchecker requires a way to configure the language for the individual entries.

- Spelling errors in names should use a database of names for detection of misspellings.
- Initials can be detected by finding cases with a single letter, perhaps followed by a punctuation. For suggestions, databases with names will be useful. Handling multiple letters combined can be done by looking for relatively few (probably up to 3) letters that are all capitalized. Another approach is to use a database to detect initials and their corresponding full names.
- Erroneous online lookups will be impossible to detect with certainty, because one's intended search result cannot be known by software. The detection of the other issues can give an indicator of a bad lookup. Since online lookups is a likely way to handle the databases needed for this tool, the quality of a lookup is a concern. The most trusted online database available should be used whenever possible. A system doing lookups in multiple databases will give further indications of potential issues. The results should always be confirmed by the user, to prevent erroneous data.
- Conformity to de-facto standards and the BibTEX specification requires a set of rules specifying: required tags, optional tags, exclusive tags, and inclusive tags. The values should be validated to the extent possible.
- Detecting invalid values should be done when clear rules for correct values can be specified: such as ISSN, year, and month. Furthermore, values that can be verified using a database should also be verified.
- Detection of journal names in abbreviated form, or in full form, can be done using a database of known journal names and their abbreviations. It can use a database to de-abbreviate or abbreviate. Furthermore, this can be refined by detecting unknown abbreviations.
- To handle BibTEX strings that end up as part of the text, the text should be compared with the strings in the .bib file.

**Second step: Consistency**

- To detect duplicate entries, the best way is to: first compare unique identifiers, if the identifiers are missing, then identical entries. Otherwise, potential duplicates are detected by a specification of similarity: having the title and authors as primary indicators. For duplicated values, it should use the strings if possible, to determine when the same

source is referenced. For the duplicated values, it should maintain a list of values that are likely to be repeated (such as journal names) and use that to detect duplicates in textual content.

- To detect name changes of forums, a database of known changes is used, in conjunction with a way of specifying the changes.

- Inconsistent tag usage should be handled by comparing entries from the same forums. The comparison should show missing and additional tags. Relating forum entries from different points in time might increase the usefulness, but adds the need to handle changes over time.

- Inconsistent entry keys should be handled by having a naming scheme based on the data in the entries, with a way to disambiguate if two different entries would get the same name.

### Configuration

Both of the steps above require ways to configure the behavior, to prevent false positives. The use of configurations should be as close to BibTEX as possible.

The preference for using the BibTEX format allows people to use what they are familiar with. Using a format that is readily supported in programming frameworks, e.g., JSON or XML, might be easier by the implementer. However, people outside computer science, such as a physicist or the helpful chemist from earlier, will likely not be familiar with such formats, nor should they required to. A user of BibTEX should at best only be concerned with the BibTEX format, when working with BibTEX.

"To use something in anger", is an idiom for: when something has been tested in practice. The idea of coding in anger has been expressed by Philip Wadler for functional programming [41]. For the BibTEX user, coding in anger could be the situation where the deadline is getting closer and one just needs things to work, at best ten minutes ago. When faced with frustrations like that, one tends not to care about beautiful and elegant solutions, rather wanting something that works with a minimum of personal effort. To accommodate the user writing BibTEX in anger is another reason to keep the specification as close to BibTEX as possible, since it will minimize the effort.

Configuration should be done via de-facto standards inside the .bib file, whenever possible. For some configurations, de-facto standards inside the .bib file are unreasonable. These configurations are better put into separate files. However, the specification should still be designed to match BibTEX as closely as possible, i.e., still using entries, tags and values to configure.

### 6.2.2 In practice

#### Entry level configurations

For entry level configuration, it is appropriate to introduce two de-facto standards: `OLDforum` to mark a previous name of a forum, and `OPTanalyze` to configure the analysis.

The division into two de-facto standards is done for two reasons: for `OLDforum`, the additional standard will allow bibliography styles to make use of the additional information (one could easily imagine a bibliographic style write "`NISSC` *formerly known as* `NCSC`"), and for `OPTanalyze`, the content is considered unlikely to be relevant to print in a bibliography. Furthermore, the settings for `OPTanalyze` are kept in one tag to prevent a multitude of new tags.

For the `OLDforum` tag, the value should be the string containing the old name for the forum. In some cases, a forum can have multiple name changes. When multiple name changes occur, referring to the most recent name in the .bib file is desired. The tag is intended for disambiguation within a given file, not all files in general. If the tag is used for a bibliography style, having a multitude of names as the value will likely be confusing.

However, if a forum name has been omitted, because no entries with that name were in the .bib file, then a re-detection of name changes will be needed if this name is added later. Furthermore, using the old forum name in bibliography styles may cause issues if we only desire to show the previous name, when it is used in the references. Since the tag is presently not in use in any bibliography style, these issues are not considered further. Alternatively, the format could be a list of names (comma separated, since BibTEX uses a comma as a separator), this list would allow detailed backtracking.

Defining entry level deviations is done using `OPTanalyze`, using spaces to separate multiple settings. The configurations is written, so each option can be read stand alone. The values for desired deviations are as follows:

- `@DUPLICATEOK=tag` to specify that a potential duplicate is deliberate, replacing `tag` with the entry key of the potential duplicate.
- `@LANG=XX` to specify the desired spellcheck language, replacing `XX` with the language code desired, e.g.., `@LANG=FR`.
- `@SPELLINGOK` to mark the spelling as correct.
- `@AUTHORSPELLINGOK` to mark that the names are correct.
- `@AUTHORINITIALSOK` to mark that the initials are correct.
- `@NOLOOKUP` to mark that no lookup should be done for the content of this entry.

```
@BOOK{blendstrup1994Mistbaenk,
  author = "Jens Blendstrup",
  title = "Mennesker i En Mistb{\ae}nk",
  year = 1994,
  OPTanalyze = "@LANG=DA @AUTHORSPELLINGOK @CONFORMITYOK"
}
```

**Figure 6.1:** An example using the de-facto standards for configuration, setting the language for spell checking to Danish, accepting the name "Jens Blendstrup" and ignoring the missing publisher.

- `@CONFORMITYOK` to mark conformity to the specification and de-facto standards as correct.
- `@ABBREVIATIONOK` to mark an abbreviated form as correct.
- `@STRINGSOK` to mark that the text does not contain any strings.
- `@TAGSOK=forum` to mark that the usage of tags is correct and defines the standard for tag use for the entries from the *same occurrence* of the forum.
- `@TAGSOK=future` to mark that the usage of tags is correct and defines the standard for tag use for the entries from the *same and future occurrences* of the forum.
- `@TAGSOK=single` to mark that the usage of tags is correct for this single entry, not affecting other entries from the forum.
- `@ENTRYKEYOK` to mark the entry key as correct.
- `@LEXICALLYOK` to ignore all lexical checks for the entry. Should be used with care.
- `@CONSISTENCYOK` to ignore all consistency checks for the entry. Should be used with care.
- `@OK`, to mark an entry as fully correct, essentially the same as marking the entry with: "`@CONFORMITYOK @LEXICALLYOK @CONSISTENCYOK`". Should be used with care.

The settings: `@ABBREVIATIONOK`, `@LEXICALLYOK`, `@CONSISTENCYOK`, and `@OK` are mainly present for convenience, and to ensure that the configuration is consistent. An example of the configurations can be seen in Figure 6.1.

For the conformity and de-facto analysis, a consideration would be to have configurations, for explicitly specifying which deviations are accepted. For instance, specifying that a missing title is accepted in an entry. However,

```
@STRING{analyze_lang_danish = "@LANG=DA "}
@STRING{analyze_author_spelling_ok = "@AUTHORSPELLINGOK "}
@STRING{analyze_conformity_ok = "@CONFORMITYOK "}

@BOOK{blendstrup1994Mistbaenk,
  author = "Jens Blendstrup",
  title = "Mennesker i En Mistb{\ae}nk",
  year = 1994,
  OPTanalyze = analyze_lang_danish
               # analyze_author_spelling_ok
               # analyze_conformity_ok
}
```

**Figure 6.2:** Figure 6.2 rewritten to use strings for configuration.

explicitly allowing and denying tags, will likely be redundant, since once a deviation has been accepted, it will not be likely to change.

A similar set of tags could also be defined for the consistency check. For the consistency check, it can be argued that an entry that is not conforming to the standards of a forum, may be updated to do so. For instance, if the norm for a forum is to have an ISSN on all entries and some of the entries do not have an ISSN. Those entries might get an ISSN assigned later, which is now possible to add to the entry. After adding the missing information, the `@CONSISTENCYOK` configuration is redundant. When adding a tag, if is is the reason for the configuration `@CONSISTENCYOK`, then one can remove `@CONSISTENCYOK`. Another approach to maintaining the minimal set of configurations is to analyze whether each configurations is necessary. This approach is considered more appropriate, because of the simplicity for the user.

Another potential change is to have a configuration for trusted lookup services. For example, if one knows that an entry is correct in a certain database, then specifying that this database is trusted for that entry. This configuration might lead to a false sense of security, since there is no way to guarantee that the data will never be corrupted in that database. In some cases the database will be the definition for the correct value, e.g., for a DOI number the number from International DOI Foundation, is by definition the correct number.

To make the configurations even more intuitive for a BibTEX user, an option is to add BibTEX strings with the relevant options in the top of one's .bib file. That way, when configuring, one can just use the BibTEX strings and concatenate the relevant configurations. An example configuration using strings is displayed Figure 6.2. These strings would have to be added in the top of the .bib file, so BibTEX will not complain.

### Bibliography level configurations

Some configurations are not specific to an entry, but the entire bibliography. Two options for these configurations are: to put such options inside one's .bib file, or to put them in separate files. Adding them to one's .bib file will introduce an additional mess in the file, which is counterproductive, since the goal is to clean up the mess. Furthermore, the configurations may not correspond to proper BibTEX formatting. Having the configurations in separate files provides: separation of concerns, a clean .bib file and allows deviations from BibTEX, if needed. Furthermore, having configurations in separate files allows sharing of the files, for instance, if a publisher wants their authors to follow a certain setup.

Such configuration files should still, if possible, follow the BibTEX format. Thus, using entries with tags and values for configuration and for configurations where entries and tags do not make sense, defining BibTEX strings will be the favored choice.

Because the format described can be put into one file rather than many, there are a few issues with disambiguation. For instance, there can be multiple configurations for the title of a `@PROCEEDINGS`. To solve this, the first word in the entry keys should identify the type of configuration, followed by an underscore and a descriptive name of the user's choice. For example, the key `forum_ncsc`, seen in Figure 6.3, indicates that the configuration is regarding the forum (more specifically a name change). The key prefixes are: for forum configurations `forum`, for de-facto standards `standards` and for validation of values `validation`. Using this scheme allows a normal BibTEX parser to read the files.

For changes in names of forums, a database of such changes is needed. Currently, no such database exists (to the authors knowledge), so the user will need a way to specify his own. Even if such a database did exist, a way to configure name changes is still desired, since the database might not be complete. A name change can be specified by having two entries for the desired forum, adding the `OLDforum` tag, marking the name change. An example of a name change configuration can be seen in Figure 6.3.

However, this configuration ignores that the names in the actual .bib file may be in their abbreviated form. Some forums also have, as part of the name, text identifying which instance of the forum it is. For example, in Figure 6.3, an entry would be named something like `Proceedings of the 20th National Information Systems Security Conference` and not just `National Information Systems Security Conference`, as in the example. Instead of writing the name as a text, a better way might be to use strings. If the .bib file construct forum names by using strings, as in Figure 6.4. the configuration can reuse the string for identifying the forum (`nissc` in the example). This configuration will allow the analysis to detect that it is the same string, and thus enable it to detect that it is the same forum. The

```
@PROCEEDINGS{forum_ncsc,
  title = "National Computer Security Conference"
}

@PROCEEDINGS{forum_nissc,
  title = "National Information Systems Security Conference",
  OLDforum = "ncsc_forum"
}
```

**Figure 6.3:** Configuring a name change of a forum

```
% Re-usable strings
@STRING{PROCintro = "Proceedings of the"}
@STRING{nissc = "National Information Systems Security Conference"}

% Conferences
@STRING{nissc20 = PROCintro # "20th" # nissc}

% Proceedings
@INPROCEEDINGS{porras1997emerald,
  title = "EMERALD: Event monitoring enabling response " #
          "to anomalous live disturbances",
  author = "Porras, Phillip A and Neumann, Peter G",
  booktitle = nissc20
}
```

**Figure 6.4:** .bib file using strings for conference names

corresponding configuration will look like Figure 6.5.

The configuration of name changes should be using the most general entry type available, such as `@ARTICLE`, `@PROCEEDINGS` and `@BOOK`. The name change analysis recognizes and maps the general entry types to the specific types. For instance, recognizing that `booktitle` in `@INPROCEEDINGS` corresponds to the `title` in a `@PROCEEDINGS`.

Specifying the de-facto standards is done using BibTEX entries, and only deviations should be specified. The configurations correspond to the rules in Section 2.4.2, with the addition of the option to refuse a tag. The configurations are:

```
@PROCEEDINGS{forum_nissc,
  title = nissc,
  OLDforum = "forum_ncsc"
}
```

**Figure 6.5:** .bib file using strings for conference names

```
@ARTICLE{standards_article,
  address = "@DENY",
  DOI = "@REQUIRED @EXCLUDES=ISSN",
  ISSN = "@REQUIRED @EXCLUDES=DOI",
  url = "@OPTIONAL"
}

@BOOK{standards_book,
  ISBN10 = "@REQUIRED @INCLUSIVE=ISBN13",
  ISBN13 = "@REQUIRED @INCLUSIVE=ISBN10"
}
```

**Figure 6.6:** A snippet of the desired BibTEX based configuration for the correctness checker

- `@REQUIRED` for a tag that is required to be present in the entry type.
- `@OPTIONAL` for optional tags.
- `@DENY` for tags that are in the default configuration, that we want to reject.
- `@EXLUDES=tag` for a tag that excludes the use of another tag, replacing `tag` with the name of another tag. For example, if one allows both `ISSN` and `DOI` as tags, but wants to ensure that only one of the tags is present, one would have the following: `ISSN = "@REQUIRED @EXLUDES=DOI"` and `DOI = "@REQUIRED @EXCLUDES=ISSN"`.
- `@INCLUDES=tag` for tags where one of them is required and the other optional. Usage is similar to `@EXCLUDES=tag`.

An example of a configuration of standards can be seen in Figure 6.6. This example sets article entries to: reject `address` tags, that either `DOI` or `ISSN` is present (but not both) and adds `url` as an optional tag. For book entries, the example sets: that ISBN10 and/or ISBN13 must be present.

The configuration of standards also allows usage of a `*` as a 'wildcard'. The wildcard will then match anything, for example `@*PROCEEDINGS` will match `@PROCEEDINGS`, `@INPROCEEDINGS` and any entry type that one introduces that has a name ending in proceedings. These wildcards can be used for both tag names and entry types.

For validation of values, some kind of valid patterns are required. In general regular expressions can be used for validation. Using those, one could introduce a set of entries for validation rules, just as done for the de-facto standards, having the patterns as the values. However, using regular expressions contradicts the desire to keep the configuration close to the BibTEX specification.

```
@*{validation_all,
  author = LETTERS_ONLY,
  year = NUMBER
}

@ARTICLE{validation_article,
  DOI = DOI,
  ISSN = ISSN,
  url = URL
}
```

**Figure 6.7:** A snippet of the desired BibTEX based configuration for the correctness checker

To remedy this, using strings with common patterns can be used. For instance, having a string named: `LETTERS_ONLY`, `NUMBER`, `ISSN` and so on. This use of strings will keep it close to the BibTEX specification, allowing the flexibility of regular expressions. Using strings containing regular expressions is done to allow adding more complicated rules than just the predefined strings, and one can imagine sharing of useful collections of validation strings. An example of such a configuration can be seen in Figure 6.7.

Abbreviations of journal names can be configured by adding `@ARTICLE` entries, using the two tags: `abbreviated` and `fullname`, specifying the abbreviated journal name and full journal name respectively.

The specification of the format for entry keys is done by using a BibTEX string named `ENTRY_KEY`. Inside the string, some way of specifying the desired template for entry keys is needed. Using a template scheme, such as `{tag}` to match tags, is probably the best solution. This template system contradicts the desire to keep the format close to BibTEX, but no better way has been found. A template could look like: `{author}{year}{title}`.

However, this template is insufficient for two reasons: spaces are not allowed in the entry key, and when people name entries, they often use names such as: the last name of the first author in the list and one significant word from the title. Refining the templates to allow 'selectors' before the fields, such as: selecting the first part of an entry, last name, first name, and significant word, will improve the usability. For example, `[lastname]{author}` would select the last name of the authors. One can use these selectors in conjunction, to refine the result. For instance, to get the last name of the first author: `[lastname][first]{author}`. Again this moves the format away from how BibTEX is defined and no better solution has been found.

Fortunately, BibTEX strings comes to the rescue - at least partially. Having strings for the most common matches will ensure that most users will never need to see, nor even know about, the underlying pattern matching

```
@STRING{ENTRY_KEY = LASTNAME # OF # FIRST # AUTHOR # THEN # YEAR}
```

**Figure 6.8:** An example of a entry key pattern using the first name of the first year, followed by the year.

system. Using strings will allow the user to use concatenations to build the desired pattern. And introducing an empty string named `OF` and one named `THEN` to support a more natural language. An example can be seen in Figure 6.8.

The strings and selectors are:

- `FIRST` is the first part of a tag value, if the tag data is separated by `and`, like a list of authors, the first part of this list should be selected, otherwise select the first word. The corresponding selector `[first]`.
- `LAST` is the last part of a tag value, if the tag data is separated by `and`, like a list of authors, the last part of this list should be selected, otherwise select the last word. The corresponding selector `[last]`.
- `FIRSTPART` selects the first part of a tag value, if the tag data is separated by `and`, like a list of authors, the first word in each part of the list should be selected, otherwise just the first word. The corresponding selector `[firstpart]`.
- `LASTPART` selects the last part of a tag value, if the tag data is separated by `and`, like a list of authors, the last word in each part of the list should be selected, otherwise just the last word. The corresponding selector `[lastpart]`.
- `FIRSTNAME` same as `FIRSTPART`, included to allow a more natural language.
- `LASTNAME` same as `LASTPART`, included to allow a more natural language.
- `SIGNIFICANT` selects the significant words of a sentence, by removing anything that is not nouns. The corresponding selector `[significant]`.
- `OF` and `THEN` are empty placeholders, included to allow a more natural language.

The templates for tags are:

- `AUTHOR` is the content of the author or editor tag. The corresponding template `[author]`

- `FORUM` is the tag containing the relevant forum, such as: journal name, conference name and publisher. The corresponding template `[forum]`.

- `TITLE` is the content of the title tag. The corresponding template `[title]`

- `YEAR` is the content of the year tag. The corresponding template `[year]`

One can introduce new tags in this manner and define appropriate strings. A way of expanding the selectors is desired, but would complicate things. Rules for handling empty tags and alternative actions would be useful, however, this would complicate things even further.

## 6.3 Orangutan

### 6.3.1 Introduction

We have created a prototype for some of the analysis, under the name 'Orangutan'. The name Orangutan is inspired from Terry Pratchett's Discworld books, where the librarian at the Unseen University is an orangutan.

### 6.3.2 Why Orangutan came to be

There are a lot of tools that provide partial solutions, such as using online lookups to compare entries and to detect duplicate entries. None of these use the idea of providing a general framework for reuse, or de-facto configuration inside a BibTEX file. Orangutan is a proof of concept that analysis can be done using the de-facto configurations.

### 6.3.3 What is Orangutan

In the same spirit as BibTEX, Orangutan is designed to be a simple software tool to help one improve bibliographic references. The analysis is designed over the same principles as in Section 6.2.

### 6.3.4 How Orangutan is used in principle

When analyzing BibTEX files, Orangutan operates on the first step of the analysis: correctness and lexical concerns. The tool uses options, set by introducing a de-facto standard and JSON files. The options are for specifying the language for the spell check, entries that are considered to be correct, and the standards used.

```
{
  "book": {
    "author": {
      "required": true,
      "excludes": "editor"
    },
    "editor": {
      "required": true,
      "excludes": "author"
    },
    [...]
  }
}
```

**Figure 6.9:** A snippet of the JSON for configuring the correctness checker

### 6.3.5 How Orangutan is used in practice

In the current version, three analyzing modules are in use: a spellchecker, a correctness checker, and an abbreviation checker.

The configuration format for entry level configuration uses a small subset of the configuration specified. It adds the `OPTanalyze` tag and the configuration `@OK` and `@LANG`.

The spellchecker module runs 'aspell' in the background to do the spell check. Currently, the spellchecker module is limited to titles only. When spell checking, it uses the configuration to determine the language, e.g., `OPTanalyze = "@LANG=DA"`.

The correctness checker verifies the conformity with the BibTEX specification and a few known de-facto standards. Currently, the format for specifying entry rules is JSON, but the BibTEX based format described in Section 6.2.2 would have been better. The format in use is essentially the JSON equivalent of the one specified earlier. Figure 6.9 displays a snippet of the configuration.

In Orangutan, the de-facto standards added are `crossref`, `issn`, `doi`, `oldforum` and `opt*`, the last one using the wildcard to match all tags that have been commented out. All the de-facto standards are accepted on `*` entries, again using the wildcard to match all entry types.

The abbreviation checker runs through journal names using a known list of abbreviations. The detection can be improved by detecting abbreviations that are not on the list, using known standards for how to abbreviate and detecting the various ways people abbreviate. Furthermore, it can be improved by detecting if the journal name is written with text rather than using a BibTEX string.

## 6.4 Summary and conclusions

Analyzing BibTEX files can be done using the two phases suggested, starting with the lexical and correctness concerns, followed by the consistency concerns.

To find the lexical and correctness issues, the following can be done: running a spellcheck on entries, checking author names in databases, detecting initials using single letters optionally followed by a punctuation, prioritizing trusted sources for online lookups, using rules to check conformity with the standards, using patterns to verify values, using databases of known abbreviations to detect abbreviations, and detecting when names of strings appear inside text.

To find the consistency issues: entries should be compared to find duplicate content, to find changes in forum names, entries from the same forum should be compared to detect inconsistent tag usage, and a pattern should be used to detect inconsistent entry keys.

For the analysis, a configuration is provided to prevent false positives. A format for configuring has been introduced, keeping as close to the BibTEX format as possible. For entry level configuration by introducing two de-facto standards, and for general configuration by introducing an approximation to a BibTEX file, using additional wildcard notations, strings with special meanings and entry key naming to disambiguate the purpose of the configuration.

A prototype, Orangutan, has been introduced, showing a proof of concept that the analysis can be done and that de-facto standards can be used for configuration.

Having a tool that can analyze BibTEX files, and finding the issues in the .bib file is the first step in handling the issues. However, just like the lookup services can lure a user into a false sense of security, so can the result of an analysis. Dijkstra's famous quip "testing only shows the presence of bugs, not their absence" [7] also applies to this analysis tool: it can only reveal the issues that are tested for. If it does not find anything, it does not mean that there are no issues. A final question remains though: what should we do with the issues that the analysis tool detects? This is the topic of the next chapter.

# Chapter 7

# Organizing BibTeX files

*Now that we have found love*
*what are we gonna do*
*with it?*
– Stevie Wonder

## 7.1 Introduction

For organizing BibTeX files, the following is covered: how to organize them principle (Section 7.2.1), how to organize them practice (Section 7.2.2), what the choices for the prototype Orangutan are (Section 7.3.1), how Orangutan handles them in principle (Section 7.3.2), and how Orangutan handles them in practice (Section 7.3.3)

## 7.2 Organizing in general

### 7.2.1 In principle

Having found the issues that are possible, a way to react to them is needed. Correcting is either done by changing the entries to a state, where the analysis cannot find any more issues, or by using the configurations to tell the analysis that the issues are false positives.

There are two basic ways of reacting to the issues, one is to automatically correct them and the other is to present the issues and have the user decide what to do. As pointed out in the problem descriptions, it is hard, if not impossible, to automatically correct, since it can lead to incorrect changes and thus introduce a new structural issue. Thus, the organizing is done by presenting the issues, letting the user decide on the action to take.

When presenting the user with the issues, there are again two approaches: letting one do a range of choices to handle the issues, before outputting a corrected .bib file, or giving a list of issues and suggestions for corrections.

The last one requires the user to manually edit his .bib file, and if the issues are listed with line numbers, they should be listed backwards, so the lines will be correct throughout the correction.

After all the concerns have been addressed, a pretty printing utility will be a nice final touch, ensuring a consistent structure inside one's .bib file and ensuring consistent indentations and formatting.

### 7.2.2 In practice

When multiple suggestions are available, all the alternatives should be given to the user. All suggestions should include a way to find the violating entry, i.e., either the entry key or a location in the file. For all the detected issues, it should suggest using the relevant configurations from Section 6.2.1, with exception of the ones to disable entire checks, i.e., `@LEXICALLYOK`, `@CONSISTENCYOK` and `@OK`. The suggestions that the tool gives should be as follows:

- For duplicate entries, it should suggest: merging the entries, updating the content of one of the entries, and adding a de-facto configuration to mark the duplication as deliberate. In the case where a merge is not possible, or if the online lookup contains additional information, it should also suggest the result from an online lookup. For duplicated content, it should suggest using `crossref` when relevant and otherwise changing the content to make use of strings.

- For general spelling errors, it should inform about the misspelled word and show the suggestions from the spellchecker. Furthermore, if online lookups are used to prevent false positives, the result from those should also be presented.

- For spelling errors in names, it should only show the names that cannot be verified using databases. Then it should show the name or names that present challenges, and if possible, suggest other likely names from the databases.

- Initials should be presented to the user, and if possible, suggestions from the name databases.

- Whenever online lookups are used, if there are suggestions, they should be presented to the user, along with the option to disable the online lookups. This presentation should also be the case when the lookup is a part of another tool.

- For name changes, it should suggest usage of a de-facto standard to highlight a previous name of the forum, following the rule of suggesting only names actively used in the document, and suggesting the latest

name, prior to the current entry. If the entry is from the earliest instance of the forum in the file, no suggestions should be made.

- De-facto standards and specification conformance should result in information about the deviations from standards, i.e., if a field is missing, this should be marked with the suggestion of adding it, if a field is not specified or has been marked with deny, it should suggest removal. In the case of exclusive and inclusive fields where both are missing, it should suggest an alternative action. For the case of exclusive fields where both are present, it should suggest removing one of the violating fields. Furthermore, it should show if the validation of the values fail, and if possible, where in the check they fail. It should also show the suggestions from any online lookups in order to provide possible corrections.

- For abbreviations, it should suggest changing to usage of strings, if the values are written as text. It should suggest changing the content of the strings to either full names or abbreviated forms depending on the desired format for the user.

- When detecting possible strings that ended up in text it should suggest either changing the entire text to a string, if it is the entire text that matches the name of the string and otherwise splitting the text, concatenating the matched string with the rest of the text. If a text has a string representation, it should suggest removing the text and replacing it with the string, concatenating it with the rest of the text, if relevant.

- For inconsistent tags, it should suggest the shortest path to consistency, whether it is by removing tags or adding tags to the entries.

- When entry keys are inconsistent with the specified pattern, it should suggest replacing the entry key with the value that matches the pattern.

By following these instructions, it should at all times be possible to reach a point where the analysis does not show any more issues: a fixed point.

## 7.3 Orangutan

### 7.3.1 What

Orangutan is designed as a framework rather than as an end-user application. The output it gives is the suggestions from the analysis. For instance, a tool can list the suggestions from Orangutan in a backward order, for manual editing.

### 7.3.2 How in principle

The output is given in JSON, which most programming frameworks support. The output can be in a trimmed version showing only the detected issues, or in a full version containing all the entries. Thus, the output could be used for listing only the issues, or for printing an entire corrected BibTEX file after choosing the desired solutions.

### 7.3.3 How in practice

Orangutan gives back a JSON string containing at least the entries with detected issues. If configured to do so, it keeps the entries without issues. For the entries in the output that have issues, an object is attached with the name `orangutan`, detailing the specific issues and suggestions.

All detected issues will be put in an object named *orangutan* on the internal object for the entry. Having the corrections along the object will allow printing the corrected entry, once an action has been decided on. Effectively, the *orangutan* object functions as a map or dictionary, having an item for each entry tag with detected issues, and an object containing the details of the issues.

The spell checker will add an object named *spelling* to the object with the issues. It contains the details from the spellcheck: how many words it checked *wordCount*, how many misspellings it found *misspellingCount* and most importantly a list named *misspellings* for the details from the spell checker. Each item in *misspellings* is an object consisting of: the misspelled word in *word*, the position of the misspelled word in *position* and a list of suggestions inside *alternatives*. An example of a spelling error can be seen in Figure 7.1.

When checking for correctness, it performs a conformance check that marks entries according to the BibTEX specification and de-facto standards. This check is named conformance check. The output specifies when an entry type or a tag is in violation with the rules specified. The conformance checker adds an object for the tags with conformance errors to the *orangutan* object. The object for the tag will then have an object named $specificationConformance$ containing a description of the issues and a corresponding code. Figure 7.2 displays an example of a violation of an exclusive rule.

The current version just suggests the full name whenever an abbreviation is detected. It builds a structure with suggestions for full names when an abbreviation is detected. Figure 7.3 displays an Orangutan output for an abbreviation.

A trimmed version of the output for an entry is displayed in Figure 7.4, which contains enough information to re-print the entry. A tool using Orangutan can thus take this output, update the representation of the entry

```
{
  "title": {
    "spelling": {
      "wordCount": 3,
      "misspellingCount": 1,
      "misspellings": [
        [{
          "type": "misspelling",
          "word": "Algoritm",
          "position": 10,
          "alternatives": [
            "Algorithm",
            "Alacrity",
            "Ageratum",
            "Alacrity's" ]
        }]
      ]
    }
  }
}
```


**Figure 7.1:** An example of Orangutan output on a spelling error


```
{
  "author": {
    "specificationConformance": {
      "description": "[author] and [editor] " +
                     "cannot be in the same entry",
      "code": 3,
      "field": "editor"
    }
  },
  "editor": {
    "specificationConformance": {
      "description": "[editor] and [author] " +
                     "cannot be in the same entry",
      "code": 3,
      "field": "author"
    }
  }
}
```


**Figure 7.2:** An example of Orangutan output on a conformity error where the exclusive use has been violated

```
{
  "journal": {
    "abbreviations": {
      "abbreviation": [
        "am. j. potato res."
      ],
      "suggestions": {
        "am. j. potato res.":
            [ "American Journal of Potato Research" ]
      }
    }
  }
}
```

**Figure 7.3:** An example of Orangutan output on an abbreviation.

according to one's choices, and output a complete suggestion for a corrected entry. It should be noted that the tool does not suggest the configurations for analysis. The suggestion of configurations can arguably be the responsibility of a framework, such as Orangutan, and of the frontend.

## 7.4 Current status of the prototype

At the time of writing, the prototype can do a spell check, detect abbreviations and check the conformity to a given specification.

The language for the spell checker can be set using the de-facto `OPTanalyze`. Spell checking is only done for title tags. The spell checker used is *aspell*, which mush be in the path. There is no option to add new words to the spell checker via Orangutan itself. However, the spell checker is configured to use the folder *aspell/* for local word lists, which has to be relative to the directory, from which the implementing tool is run. The format for the word lists is specified in the Aspell manual under *Format of the Personal and Replacement Dictionaries* [3].

For checking the specification and conformity a simple set of rules are defined. The default rules are the BibTEX specification according to *BibTEX Entry and Field Types* [34], and de-facto standards for `crossref`, `OPT*`, `OLDforum`, `DOI`, and `ISSN`. Thus is allows tags commented out by prefixing the tag name with `OPT`. The specifications are currently set in the directory of Orangutan.

Abbreviations are detected using a list of know abbreviations, taken from JabRef. It can only detect abbreviated forms and suggest full form, not vice versa. The abbreviations are stored in a JSON file, mapping from the abbreviated form to the full form. The abbreviation configuration is

```
{
  "type": "other",
  "citationKeyUnmodified": "jelly_baby",
  "citationKey": "jelly_baby",
  "entryType": "article",
  "entryTags": {
    "author": [{
      "type": "text",
      "delimiter": "{",
      "part": "Rincewind the Wizzard"
    }],
    "title": [{
      "type": "text",
      "delimiter": "{",
      "part": "Interesting Times with Potatoes and Jelly Beans"
    }],
    "journal": [{
      "type": "text",
      "delimiter": "{",
      "part": "Am. J. Potato Res."
    }],
    [...]
    "year": [{
      "type": "text",
      "delimiter": "{",
      "part": "1994"
    }]
  },
  "orangutan": {
    "journal": {
      "abbreviations": {
        "abbreviation": ["am. j. potato res."],
        "suggestions": {
          "am. j. potato res.":
              ["American Journal of Potato Research"]
        }
      }
    }
  }
}
```


**Figure 7.4:** A trimmed example of an entry from Orangutan

currently located in the directory of Orangutan.

Inside the Orangutan source, the work for consistency checks has been started, but is far from a working state. There is no analysis for any of the other issues.

The Orangutan framework is located at `https://github.com/rohdef/orangutan`, and a simple command line implementation is located at `https://github.com/rohdef/orangutan-demo`. Note that nodejs and npm is required to run the tools, and that running `npm install` is a prerequisite to running the demo or using the framework.

## 7.5 Summary and conclusion

Using the results from the analysis of a .bib file, sensible suggestions for a course of action can be provided. For most results, a straightforward solution can be provided, such as the correct spelling of a word. The output from the prototype Orangutan shows a proof of concept for providing output that can be used to resolve the issues from the analysis.

# Chapter 8

# Conclusion and perspectives

*All that matters on the chessboard is good moves.*
– Bobby Fischer

Let us recapitulate: we have first described BibTEX – both how to use it in principle and how it is used in practice (Chapter 2); we have then listed a range of practical issues BibTEX users encounter (Chapter 3), we have proposed an approach to handling them (Chapter 4), and we have reviewed how they are tackled in related work (Chapter 5); we have then presented an analysis of BibTEX files that detects these issues (Chapter 6), and we have described how to solve them by organizing BibTEX files using the results from this analysis (Chapter 7). We have implemented a part of this analysis in a prototype, Orangutan.

BibTEX, despite its wide use, is far from a perfect tool, and BibTEX user's face challenges that range far and wide. Many tools surround BibTEX and so do a lot of alternatives, some of which do provide partial solutions to some of the issues one faces. However, as analyzed in Chapter 5, these tools are far from sufficient. For most of the challenges a BibTEX user faces, it is possible to provide analysis tools that detect potential issues.

Such analyses cannot be perfect though, since any set of rules yields false positives or relies on assumptions that are ill-founded, since BibTEX is not formally specified: while the soundness of the specification of BibTEX is not questioned, its completeness is unknown. Analyses of BibTEX files therefore need to have configurations, through de-facto standards, to detect and ignore false positives. In most cases, these analyses can also provide suggestions for improvements, such as the suggestions from a spell checker. These suggestions can then be used to organize one's .bib file, either by correcting the issues that have been detected or by adding de-facto tags to prevent false positives.

Just like chess players who strive to always improve the quality of their moves, one can wish to improve the quality of one's BibTEX files. We have designed Orangutan as a proof of concept for suggesting improvements. It

is our hope that this proof of concept can contribute to improving the general quality of BibTEX files and, consequently, can improve the precision of bibliographic references in documents as well as save time for their authors and their readers.

# Bibliography


[1] Andy and Clemens Koppensteiner. *How to abbreviate journal name in citation*. 2011. URL: http://tex.stackexchange.com/questions/33441/how-to-abbreviate-journal-name-in-citation/34764#34764 (visited on 02/21/2016).

[2] Anonymous. *spell chek for review*. 2015. URL: http://sourceforge.net/p/jabref/feature-requests/568/ (visited on 11/01/2015).

[3] Aspell. *Format of the Personal and Replacement Dictionaries*. 2011. URL: http://aspell.net/man-html/Format-of-the-Personal-and-Replacement-Dictionaries.html (visited on 03/29/2016).

[4] Clemens Barth. *Bibcut LaTeX BibTeX*. 2014. URL: http://www.development.root-1.de/Bibcut.php (visited on 11/10/2015).

[5] Joos Buijs. *Bibcleaner*. 2014. URL: https://github.com/joosbuijs/bibcleaner (visited on 11/06/2015).

[6] Joos Buijs. *Is there a tool/service that can enrich a BibTex database?* 2014. URL: http://tex.stackexchange.com/questions/174509/is-there-a-tool-service-that-can-enrich-a-bibtex-database (visited on 11/06/2015).

[7] John N Buxton and Brian Randell. *Software Engineering Techniques: Report on a Conference Sponsored by the NATO Science Committee*. NATO Science Committee; available from Scientific Affairs Division, NATO, 1970.

[8] Mekentosj B.V. *Papers - Your personal library of research*. 2015. URL: http://papersapp.com/ (visited on 11/28/2015).

[9] Wikipedia community. *Iain Banks*. 2016. URL: https://en.wikipedia.org/wiki/Iain_Banks (visited on 03/27/2016).

[10] dblp team. *BibTeX record conf/icfp/NeergaardM04*. 2006. URL: http://dblp.uni-trier.de/rec/bibtex/conf/icfp/NeergaardM04 (visited on 03/28/2016).

[11] Mr. dLib. *Metadata Extraction for JabRef*. 2011. URL: http://www.mr-dlib.org/docs/jabref_metadata_extraction_alpha.php (visited on 11/03/2015).

[12] Mr. dLib. *Mr. DLib*. 2012. URL: http://www.mr-dlib.org/ (visited on 11/03/2015).

[13] Umberto Eco. *How to Write a Thesis*. The MIT Press, 1977.

[14] fbennet. *Plugins For Zotero*. 2015. URL: https://www.zotero.org/support/plugins (visited on 10/08/2015).

[15] Alexander Feder. *Your BibTeX resource*. 2006. URL: http://www.bibtex.org/ (visited on 10/03/2015).

[16] Vidar Bronken Gundersen and Zeger W. Hendrikse. *BibTeX as XML markup*. 2007. (Visited on 02/26/2007).

[17] Hannah. *Spell checking references in EndNote*. 2007. URL: http://covendnote.blogspot.dk/2007/07/spell-checking-references-in-endnote.html (visited on 11/01/2015).

[18] Jean-Michel Hufflen. "MlBIBTEX: a new implementation of BIBTEX." In: (2001).

[19] JabRef. *Journal abbreviations*. 2014. URL: http://jabref.sourceforge.net/help/JournalAbbreviations.php (visited on 11/01/2015).

[20] JabRef. *Overview*. 2015. URL: http://jabref.sourceforge.net/ (visited on 11/01/2015).

[21] JabRef. *Resources*. 2015. URL: http://jabref.sourceforge.net/resources.php (visited on 11/01/2015).

[22] kylie.cantwell. *DOI/ArXiv/PMID lookup - bulk lookup*. 2015. URL: http://feedback.mendeley.com/forums/4941-general/suggestions/301296-doi-arxiv-pmid-lookup-bulk-lookup (visited on 10/18/2015).

[23] Mary-Claire van Leunen. "A Handbook for Scholars." In: Oxford University Press, 1992, pp. 9–45, 154–268.

[24] Mendeley Ltd. *Features Overview*. 2015. URL: https://www.mendeley.com/features/ (visited on 10/18/2015).

[25] Thomas McLean. *Fix capitalization of journal titles*. 2015. URL: http://feedback.mendeley.com/forums/4941-general/suggestions/444383-fix-capitalization-of-journal-titles (visited on 10/18/2015).

[26] Charalampos Nikolaou. *BibTooL*. 2012. URL: http://cgi.di.uoa.gr/~charnik/oss/bibtool/ (visited on 11/08/2015).

[27] ProQuest. *RefWorks Features*. 2015. URL: http://www.proquest.com/products-services/refworks.html (visited on 11/01/2015).

[28] Thompson Reuters. *Choosing the best solution for your research and reference management*. 2014. URL: http://refman.com/switch (visited on 10/18/2015).

[29] Thompson Reuters. *EndNote Basic Product Details*. 2015. URL: http://endnote.com/product-details/basic (visited on 10/18/2015).

[30] Thompson Reuters. *EndNote Downloads*. 2015. URL: http://endnote.com/downloads (visited on 10/23/2015).

[31] Thompson Reuters. *EndNote: Generate full or abbreviated journal names*. 2013. URL: http://endnote.com/kb/82228 (visited on 10/18/2015).

[32] Thompson Reuters. *EndNote X7 Product Details*. 2015. URL: http://endnote.com/product-details/x7 (visited on 10/18/2015).

[33] Thompson Reuters. *Reference Manager Roduct details*. 2014. URL: http://refman.com/product-details (visited on 10/18/2015).

[34] Andrew Roberts. *BibTEX Entry and Field Types*. 2011. URL: https://www.andy-roberts.net/res/writing/latex/bibentries.pdf.

[35] scripts/generate-developers.sh. *Developers*. 2015. URL: https://github.com/JabRef/jabref/blob/master/DEVELOPERS (visited on 11/01/2015).

[36] ShareLaTeX. *Bibtex bibliography styles*. 2016. URL: https://www.sharelatex.com/learn/Bibtex_bibliography_styles (visited on 03/21/2016).

[37] shawnwtan. *Spelling check*. 2015. URL: http://feedback.mendeley.com/forums/4941-general/suggestions/250221-spelling-check (visited on 10/18/2015).

[38] National Institute of Standards and Technology. *NISSC 1977-2000, National Information Systems Security Conference*. 2014. URL: http://csrc.nist.gov/nissc/ (visited on 02/25/2016).

[39] Catharina Steentoft. *Prevent that middle names with an "A" are wrtten in lower case "a"*. 2015. URL: http://feedback.mendeley.com/forums/4941-general/suggestions/5318473-prevent-that-middle-names-with-an-a-are-wrtten-i (visited on 10/18/2015).

[40] Various Users. *local list of journal abbreviations*. 2013. URL: https://forums.zotero.org/discussion/29501/local-list-of-journal-abbreviations/ (visited on 10/08/2015).

[41] Philip Wadler. "Functional programming: An angry half-dozen." In: *Database Programming Languages*. Springer. 1997, pp. 25–34.

[42] Zotero. *Zotero Home Page*. URL: https://www.zotero.org/ (visited on 10/18/2015).