Dealing with Ancient Works in Bibliographies

Jean-Michel Hufflen

The race of men that the immortals who dwell on Olympus made first of all was of gold. They were in the time of Kronos, when he was king in heaven; and they lived like gods, with carefree heart, remote from toil and misery. Wretched old age did not affect them either, but with hands and feet ever unchanged they enjoyed themselves in feasting, beyond all ills, and they died as if overcome by sleep. All good things were theirs, and the grain-giving soil bore its fruits of its own accord in unstinted plenty, while they at their leisure harvested their fields in contentment amid abundance. [...] Works and Days (8th century BC), Hesiod [1988] Lines 109–121.

Abstract

Bibliography processors used in conjunction with word processors have been designed in order to deal with recent works. That is, authors and publishing dates are known. Often, these informations are used to sort bibliographies, so they are supposed to be precise. This is not the case if ancient works are cited. We show how an extension of our MiBibTeX bibliography processor can allow us to deal with such cases by means of patterns.

Keywords Citing ancient works, inexact year specification, unknown authors, patterns, translated works, cross-referencing, Bibtex, extension of MiBibTeX, mlbiblatex program.

0 Introduction

Modern typesetting systems have been designed in order for new written documents to be developed, processed, and updated easily. Even if an ancient document’s body reached a final stage and does not have to be developed or updated, using such a modern processor to typeset ancient documents may be interesting, too, in order to digitalise works of the past or build new critical editions of such works. Good examples related to this approach are given by [Delprat and Orevkov 2012] and also by [Pignalberi et al. 2011, 2012]. Now let us consider bibliography processors, extracting references from bibliography database files. They have also been designed in order to build ‘References’ sections for new written documents. But often bibliographies are sorted, that is, bibliographical references are sorted with respect to authors’ names, and different works written by the same authors are sorted chronologically within the final list. Such a modus operandi is suitable if authors and publication dates are known precisely. That is obviously true for recent works, not for ancient ones:

῾Ησίοδος. ῎Εργα καὶ ἡμέραι. 8th century BC.

It can be noticed that such a reference has only historical interest, it does not point to a resource—printed or on-line—that readers can get. Most often, modern translations are used and cited, and such a translation has a publisher and a precise publication date:

Hesiod. Theogony — Works and Days [Θεογονία — ῎Εργα καὶ ἡμέραι], translated from ancient Greek by Martin Lichtfield West (Oxford University Press, 1988).

However, several translations of the same ancient work may be cited throughout the same article, if we aim to avoid the repetition of the original title and metadata associated with all these translations, a possible solution may look like:


1. There exist unsorted bibliographies, in which case the order of the references is the order of first citations of these items throughout the text. Unsorted bibliographies are traditional in some specific areas such as Medicine or History, but it seems to us that most bibliographies are sorted.
that is, by using a cross-referencing system for translations, as sketched in Hufflen [2008b]. In order for such a system to be put into action, the third item has to be specified as a separate entry, being type \texttt{MISC} or \texttt{UNPUBLISHED} if the \LaTeX{} bibliography processor \texttt{Patashnik}\cite{1988a} is used.

In this article, we do not go thoroughly into this notion of cross-references, but focus instead on the features provided by a new experimental extension of \LaTeX{}\texttt{MIBibTEX}\cite{Hufflen,2003}, so-called \texttt{inexact}, for \texttt{imprecise} metadata, such as \texttt{inexact} dates of publication or surmised author’s identity. As introduced before, we consider that an \texttt{inexact date} is a date whose year is not precisely known, e.g., \texttt{Oedipus the King}, a tragedy by Sophocles (495 BC--406 BC), was first performed around 429 BC. Another example is given by Hesiod’s \texttt{Works and Days}, as shown before. Concerning doubtful identity of authors, the following example is related to Italian music:

\begin{quote}
\end{quote}

Another suite arranged by Ottorino Respighi gives an example of an unknown author:

\begin{quote}
\end{quote}

In Section\texttt{1} we explain why such metadata are difficult to handle if ‘old’ \LaTeX{} is used. Section\texttt{2} briefly recalls the perspectives open by \LaTeX{}’s successors. Section\texttt{3} introduces and discusses \LaTeX{}\texttt{MIBibTEX}’s new syntactic features for ancient works’ metadata. Throughout the article, we mention that \LaTeX{}\texttt{MIBibTEX} provides advanced features for sorting bibliographical items, described in Hufflen [2012b]. In an annex that may be viewed as a complement to this document, we show how they have been updated. Reading this article only requires basic knowledge about \LaTeX{}\texttt{BibTEX} and \LaTeX{}\texttt{MIBibTEX}.

1. Doing it with \LaTeX{}\texttt{BibTEX}

\LaTeX{}\texttt{BibTEX} was designed in the ’80s and, as mentioned above, is suitable for ‘modern’ documentation: bibliographical entries refer to works that came out for a short period of time. Years are supposed to be written using the same format, in practice, either two digits or four ones. As a counter-example, let us consider two bibliographical items specified this way:

\begin{quote}
\begin{verbatim}
\@...\{i_0, \ldots, \text{YEAR} = 800, \ldots\}
\@...\{i_1, \ldots, \text{YEAR} = 1492, \ldots\}
\end{verbatim}
\end{quote}

the \texttt{i_0} item will be ranked after the \texttt{i_1} item because ‘800’ and ‘1492’ are viewed as \texttt{strings} rather than numbers; a string beginning with ‘1’ takes precedence over a string beginning with ‘0’. In fact, the sort key used by \LaTeX{}\texttt{BibTEX}’s standard bibliography styles for an entry is a concatenation of authors’ names, year, and title. As a consequence, any string may be associated with an entry’s \texttt{YEAR} field. For example, we could write:

\begin{quote}
\texttt{YEAR = \{8th century BC\}}
\end{quote}

That would not cause any error, \LaTeX{}\texttt{BibTEX} would not crash, but if there are other bibliographical entries with the same author name—e.g., some modern translations—the sort’s result might be strange. For the same reasons, negative years are syntactically allowed within \texttt{YEAR} fields:

\begin{quote}
\texttt{YEAR = \{-753\}}
\end{quote}

but \LaTeX{}\texttt{BibTEX} does not handle them correctly.

The situation is better about uncertain identity for authors, because we can use a command for a dummy accent:

\begin{quote}
\texttt{AUTHOR = \{Mersenne Marin\unsure\}}
\end{quote}

provided that this \texttt{\textbackslash unsure} command has been defined within \LaTeX{} documents using this entry. If this command is surrounded by additional \texttt{\textbackslash}’s braces within values associated with \texttt{AUTHOR} fields, \LaTeX{}\texttt{BibTEX} views it as an accent command and ignores it during the sorting step. However, let us notice that the two strings ‘\texttt{Mersenne Marin?’ and ‘\texttt{Mersenne Marin\unsure}’ are viewed equivalent during the sort step. That is, the ‘actual’ works of this author are merged with the works that could be attributed improperly.

4. When \LaTeX{}\texttt{BibTEX} was launched, this choice was very common: that moment was a long time before Y2K (the Year 2000 problem or Millennium bug).

5. This operation is fully explained in Hufflen [2008a].

6. Let us recall that in \LaTeX{}\texttt{BibTEX}, delimiters—braces or double quotes—can be omitted around natural numbers. If a ‘+’ sign is used for a negative number, the complete value must be surrounded with delimiters. Concerning years, we also recall that the ‘+’ sign is never used for a year \texttt{AD} (\texttt{Anno Domini}), so only years \texttt{BC} (Before Christ) are written using a sign.

7. In fact, \LaTeX{}\texttt{BibTEX}’s sort procedure is not very discriminating [see Hufflen [2008a] for complete details]. As another
2 BibTeX’s successors

For a long time, BibTeX was unrivalled as the bibliography processor associated with the \LaTeX\ word processor [Mittelbach and Goossens, 2004]. However, this program is ageing and difficultly meets some new requirements about bibliographies, as we explained in Hufflen [2011]. So modern programs aim to replace it. In particular, they aim to get rid of BibTeX’s language for bibliography styles [Patashnik, 1988b], old-fashioned and based on handling a stack. This replacement process is long, but more and more \LaTeX\ users build their documents’ bibliographies with the \biblatex\ package [Lehman, 2014]. The \MLBibTeX\ program [Hufflen, 2003] is less widespread but some big-sized projects using it have succeeded.

2.1 The \biblatex\ package

Let us recall that when the \biblatex\ package [Lehman, 2014] is used, formatting ‘References’ sections is deferred to \LaTeX\. So the files generated by \BibTeX\ in this case contain texts marked up with \LaTeX\ commands customised when this package is loaded. A new bibliography processor, \biber\ [Kime and Charette, 2014], is able to build files suitable for this package, and aims to replace \BibTeX\ within this framework. When it works, \biber\ is able to deal with more fields. For example, a \SORTYEAR\ field can take precedence over the \YEAR\ field when entries are sorted, even if only \YEAR\ fields’ contents appear within references. For our purpose, these two fields \YEAR\ and \SORTYEAR\ might be used as follows:

\[
\text{YEAR} = \{154?\}, \text{SORTYEAR} = 1549
\]

that is, the \SORTYEAR\ field allows such an entry to be ranked at the end of the entries originating from the same decade (1540’s), whereas the \YEAR\ field expresses that the unit digit is not really known. However, \biber\—like \BibTeX\—uses only string sorts, that is, negative years are not handled correctly. Besides, no syntax for authors’ uncertain identity is provided.

2.2 \MLBibTeX\n
Initially, \MLBibTeX\ [Hufflen, 2003] was designed in order to provide a ‘better’ \BibTeX, with particular examples, \BibTeX\’s standard bibliography styles do not use \MONTH\ fields during the sorting step. As a consequence, works of the same authors and same years of publication cannot be sorted according to this month information.

8. The \biblatex\ package has been introduced in Italian in Pantieri [2003].
9. A comparable approach exists within Con\TeX’t’s \bib\ module [CONTEXTGARDEN, 2012].
10. Let us mention that the \biblatex\ package allows a complete date—that is, year, month, and day—to be specified by means of a \DATE\ field, taking precedence over \BibTeX\’s standard fields \YEAR\ and \MONTH\. In practice, this \DATE\ field is more and more used, but does not really apply for inexact dates.

focus on multilingual features. Later, it integrated other services, as detailed in Hufflen [2012b]. When \MLBibTeX\’s users experiment \MLBibTeX\, they immediately notice that the latter is less permissive than the former. A particular case of this feature: only non-zero integers—including negative ones—are allowed as values associated with \YEAR\ fields. This kind of type-checking has some advantages: for example, whenever field names are unknown, warning messages are emitted. On the contrary, \BibTeX\ would have silently ignored such fields if they are optional. As another advantage, \MLBibTeX\ can perform numerical sorts on numerical data, such as years. At first glance, this feature runs counter to the introduction of more expressive power. But within \MLBibTeX\’s source files, associating a field name with a new definition of the corresponding parsing function is easy. In fact, we developed \MLBibTeX\’s parser in order for syntactical extensions to be added easily.

3 \MLBibTeX\’s inexact extension

The features we present hereafter belong to the \inexact\ extension. This extension can be used as an option of the commands \mlbibtex\ and \mlbibtex2xml\:

\mlbibtex\ [-inexact] job-name

where ‘[...]’ stands for an optional part and ‘job-name’ refers to an auxiliary (.aux) file, as in ‘old’ \BibTeX. The \inexact\ extension increases expressive power about inexact years and uncertain identity, while performing strict type-checking for some specialised fields such as \YEAR, \AUTHOR, or \EDITOR.

3.1 Inexact dates

Syntactically, a value associated with the \YEAR\ field must be either an exact year, that is, an integer beginning with a non-zero digit and possibly preceded by the ‘?’ sign, or an inexact one. If the prefix ‘ca’ (for ‘circa’) is used before an exact year, this prefix means ‘around this date’:

\[
\text{YEAR} = \{\text{ca1492}\} \quad \text{or} \quad \text{YEAR} = \{\text{ca-429}\}
\]

Other inexact years may be specified in means of \patterns, in which case ‘?’ is for an unknown digit. Examples:

‘154?’ means a year between 1540 and 1549
‘15??’ ………………… 1500 … 1599

An imprecise digit cannot be followed by a precise one. In other words, the question mark ‘?’ cannot

11. But not zero, because there was no Year zero, Year 1 immediately succeeded Year −1.
12. Let us recall that on the contrary, \BibTeX\ and \biber\ only run lexicographic sorts.
13. See Hufflen [2012b] about the commands provided by \BibTeX\.
be followed a digit from 0 to 9, so ‘1575’ is an incorrect value for a YEAR field. The zero digit is allowed at the leftmost position if it is immediately followed by a question mark, e.g., ‘0?’ is for a year between 1 AD and 9 AD, ‘-0?’ is for a year between 9 BC and 1 BC.

The rules for MiBib\TeX’s increasing sort operation are organised as follows:

- an inexact year is ranked after an exact year being the same value; e.g.:
  
  \[
  1492 < \text{ca1492} < -429 < \text{ca-429}
  \]

- a pattern for an inexact year is ranked after the greatest year it can replace;

- inexact years are sorted according to imprecision levels, that is, the number of occurrences of the ‘?’ sign. For example:

\[
1599 < \text{ca1599} < 159? < 15??
\]

because the imprecision levels of the last three—inexact—years are respectively 0, 1, and 2. If years before Christ are considered, applying the same rules yields:

\[
-1500 < -\text{ca1500} < -150? < -15??
\]

Let us go back to syntactical rules about years and denote them by means of regular expressions, as allowed in programming languages like \text{Perl} \footnote{Practical Extraction and Report Language [see \cite{Wall2000} for an introductory book about this language]} or taxonomy description languages like XML Schema \cite{W3C2008}. Exact years can be specified by the following regular expression \footnote{Readers unfamiliar with regular expressions can consult \cite{Stubblebine2007} for a good introduction}:

\[
-?\{1-9\}\{0-9\}*\]

Both exact and inexact years are described by the deterministic regular expression:

\[
\text{ca-?\{1-9\}\{0-9\}* | -?\{1-9\}\{0-9\}*\{\?\}* | -?0\{0\}?*}
\]

3.2 Uncertain identity

An unknown author is specified by the ‘??’ sequence. Using this sequence before an author name means that this identity is uncertain.

\[
\text{AUTHOR} = \{?? \text{ and Mersenne Marin}\}
\]

is for two authors, the first is unknown, the second is Mersenne Marin. The notation:

\[
\text{AUTHOR} = \{?? \text{ Mersenne Marin}\}
\]

means that the author’s identity is surmised. You can also write:

\[
\text{AUTHOR} = \{?? \text{ Marin, Mersenne}\}
\]

or put this ‘??’ notation before MiBib\TeX’s keywords introducing a person name’s parts \footnote{MiBib\TeX’s initial library now includes Scheme functions able to deal with such data. For example, given an inexact year, such a function allows us to know its imprecision level. As a consequence, it is easy to write a function that displays the corresponding year or century with a question mark if the imprecision level is respectively 1 or 2. Let us recall some previous examples, applying this function when a ‘References’ section is built results in:}

\[
\text{AUTHOR} = \{?? \text{ first => Mersenne, last => Marin}\}
\]

It is well-known that an organisation name can be used an author’s name, provided that additional braces allow this organisation name to be viewed as the Last part of a name handled by \text{Bib\TeX} \cite{Patashnik1988d}. This notation:

\[
\text{AUTHOR} = \{?? \text{ org => Gruppo Utilizzatori Italiani di \TeX}\}
\]

or use MiBib\TeX’s \text{org} keyword:

\[
\text{AUTHOR} = \{?? \text{ org => Gruppo Utilizzatori Italiani di \TeX}\}
\]

The rules for increasing sort operation are:

- an unknown author is ranked before any known one; e.g.:

\[
?? \text{ < Claudio Beccari}
\]

- an author with uncertain identity is ranked after the same author when its identity is sure; e.g.:

\[
\text{Mersenne Marin} < ?? \text{ Mersenne Marin}
\]

4 References’ look

Let us recall that MiBib\TeX is written using the Scheme programming language \cite{Kelsey1998} and running our parser on bibliography database files results in a structure that may be viewed as an XML tree. Technical details related to our XML format fall out of the scope of this paper, but we mention that this format has been enlarged in order to include markers related to inexact years and uncertain identity. Likewise, MiBib\TeX’s initial library now includes Scheme functions able to deal with such data. For example, given an inexact year, such a function allows us to know its imprecision level. As a consequence, it is easy to write a function that displays the corresponding year or century with a question mark if the imprecision level is respectively 1 or 2. Let us recall some previous examples, applying this function when a ‘References’ section is built results in:

\[
\text{AUTHOR} = \{?? \text{ Marin, Mersenne}\}
\]
Likewise, the full power of a programming language easily allows us to display a negative year—
\texttt{YEAR} = \{-753\}—as such or as a year BC:
\begin{verbatim}
    -753   753 BC
\end{verbatim}

At the present time, \textsc{Mlbibtex}'s \textit{inexact} extension is experimental. Since \textsc{BibTeX}'s bibliography styles cannot deal with these new features, it is impossible to use the compatibility mode allowing such ‘old’ styles to be applied. So only ‘new’ styles using the \texttt{nbst}'\textsuperscript{17} language could be updated. Another solution consists of bibliography styles wholly written using \texttt{Scheme}, the only style available now being an extension of plain, that is, references are sorted—as we showed above—and labelled by number.\textsuperscript{18} The only way to customise this style is to redefine some \texttt{Scheme} functions.\textsuperscript{19} Of course, we plan to go on with developing new styles and interfaces if users are interested. Criticisms and suggestions to do that are welcome.

\section{Conclusion}

On the one hand, we think that the features introduced by the \textit{inexact} extension of \textsc{Mlbibtex} should remain optional. As mentioned above, most bibliographies—in particular, for recent works—are composed of items with precise metadata about dates and authors, and \textsc{Mlbibtex}'s default type-checking procedures allow us to be ensured about that. On the other hand, the results we got within our experiments of this extension are encouraging, but we only tried basic examples. We could certainly improve our ‘inexact’ features, but we need further experiment in collaboration with people involved in ancient studies in order to go on with this direction.

\section{A Sorting with \textsc{Mlbibtex}}

In this annex, we do not detail all the features of the sort procedures used within \textsc{Mlbibtex}, we just aim to show how the framework described in \cite{Hufflen2012b} has been extended.

Let us recall that when \textsc{Mlbibtex} processes an \texttt{.aux} file, it builds a list of citation keys, according to the order of first citations using these keys throughout the text. If the bibliography to be built is unsorted, we use this list as it is, as soon as corresponding entries are associated with each member. Otherwise, this list of citation keys is \textit{sorted}. All the sorts performed by \textsc{Mlbiblatex} are \textit{stable}\textsuperscript{20} all the lexicographic sorts are language-dependent, the default language being English. The generation of such sorts is explained in \cite{Hufflen2007} and additional details concerning person names are given in \cite{Hufflen2008a}.

As shown in \cite{Hufflen2012b}, all the order relations used by our sort procedures have a \texttt{thunk}\textsuperscript{21} as an additional argument representing what is to be run when the two data are equal. This feature allows us to chain the use of several sort keys easily: if two data are viewed as equal by a sort key, the comparison based on the next sort key is immediately called. Our numerical order relations are written within this framework, so do our new comparison procedures including inexact dates. Lexicographic order relations have additional optional arguments controlling the sort’s direction and the case comparisons \cite{Hufflen2012b}. The \texttt{<authors>}\texttt{<} function, allowing us to compare bibliographical items regarding authors or editors has been extended:

\begin{verbatim}
((<authors?= rel?) i_0 i_1 k_0 f_0 f_1)
\end{verbatim}

where \(i_0\) and \(i_1\) are the two items to be compared, \texttt{rel?} the language-dependent order relation used to compare strings, \(k_0\) the thunk that it called if \(i_0\) \(i_1\) are equivalent. The new optional arguments \(f_0\) and \(f_1\) are respectively called when comparisons involve unknown authors and uncertain identity. Default values for \(f_0\) and \(f_1\) behave as we shown in \S\ 5.2 but other functions can be used. In other words, processing such cases can be customised.

Often the order relation used by the sort procedure is closely related to the bibliography style used and does not appear explicitly when the bibliography processor is invoked. Practically, bibliographies are usually increasingly sorted with respect to authors’ or editors’ names, then dates, then titles. As shown in \cite{Hufflen2012b}, the \texttt{mlbiblatex} program, a variant of \textsc{Mlbibtex} that builds bibliographies for the \texttt{biblatex} package, uses another approach: the order relation used for sorting bibliographical items is given within the command line, by means of mnemonics. In a future version, we could do the same for the order relations used to sort inexact metadata.

\section*{Acknowledgements}

I thank Claudio Beccari for his patience when he was waiting for this article, and for his Italian translations of the abstract and keywords. Many

\begin{footnotesize}
17. New Bibliography STyles.
18. At the present time, if you use both the \textit{inexact} extension and a \texttt{\textsc{bibliographystyle}} command within your \LaTeX{} document, a warning message is emitted, and this command is useless.
19. That can be done by means of \texttt{Scheme} expressions stored into a \texttt{.snmblibx} file belonging to the current user’s home directory.
20. That is, the relative order of items with equal sorting keys is maintained.
21. A zero-argument function, with respect to \texttt{Scheme}'s terminology.
\end{footnotesize}
thanks also to Gianluca Pignalberi who proof-read a first version.

References


Lorenzo Panzieri. L’arte di gestire la bibliographia con \bibtex. ArsTEXnica, 8:48–60, October 2009.

Oren Patashnik. B uu\TeX Xing. Part of the B uu\TeX distribution, February 1988a.

Oren Patashnik. Designing B uu\TeX Styles. Part of the B uu\TeX distribution, February 1988b.


Jean-Michel Hufflen

FEMTO-ST (UMR CNRS 6174) & University of Franche-Comté, 16, route de Gray, 25030 Besançon CEDEX France
jmhuffle@femto-st.fr