eXtended Sample BibTeX Styles

Nelson H. F. Beebe
University of Utah
Department of Mathematics, 110 LCB
155 S 1400 E RM 233
Salt Lake City, UT 84112-0090
USA
Email: beebe@math.utah.edu, beebe@acm.org,
beebe@computer.org
WWW home page: http://www.math.utah.edu/~beebe
Telephone: +1 801 581 5254
FAX: +1 801 581 4148

28 May 2018

Abstract

This report describes the motivation for extending the original four sample BibTeX bibliographic styles, and how the extensions are implemented to give users finer control over which extended field values are actually typeset, and how they are formatted.

The user output controls are simple Boolean options of the form `\showXXXfalse` and `\showXXXtrue` that can be changed at any time, *without* any need to edit either the BibTeX database files, or the `.bbl` files that contain the formatted reference lists automatically produced by BibTeX. All field values are wrapped in distinct user-redefinable macros.

The supported extended fields include **`articleno`**, **`bookpages`**, **`CODEN`**, **`day`**, **`DOI`**, **`ISBN-13`**, **`ISBN`**, **`ISSN`**, **`ISSN-L`**, **`LCCN`**, **`pagecount`**, **`price`**, and **`URL`**. The new styles also handle a new document style, `@Periodical{...}`.

As with many other extended bibliography styles, macros needed for typesetting in those styles must be supplied. For the new `x-*.bst` styles described here, they are available with either a LaTeX `\usepackage{x-bst}` preamble command, or with a plain TeX `\input x-bst.tex` command.

Because BibTeX was carefully designed so that style files ignore all field names that they have not been explicitly programmed to handle, the new field names are compatible with all existing BibTeX styles. The new field names are already in wide use in two public-domain BibTeX bibliography archives with over one million entries.

The new style files make it possible for users to create substantially enriched bibliographic databases, and thus, to produce formatted references that are of greater use both to readers, and to publishers.

Contents

List of Tables

List of Figures

0 Quick start

If you are interested primarily in using the new bibliography styles described in this document, rather than reading more about their design and implementation, you can do so with just *two* lines in a LaTeX or plain TeX file:

LaTeX:
```
\usepackage {x-bst}
\bibliographystyle {x-plain}
```

TeX:
```
\input x-bst
\bibliographystyle {x-plain}
```

Put the first line in your document preamble, before any typesetting is done. Put the second where you want your formatted reference list to be typeset.

You can change the style word `plain` to `abbrv`, `alpha`, or `unsrt`.

Fuller document outlines are presented later on **page 19** and **page 20**.

1 Historical BibTeX styles

The original release of LaTeX in 1986 included four sample bibliographic styles for BibTeX:

`abbrv.bst` `alpha.bst` `plain.bst` `unsrt.bst`

They are all derived from a common base file, `btxbst.doc`, with the four variants extracted by the widely available C-language preprocessor, **`cpp`**. At the time, the preprocessor in most C compilers was relatively simple, and tolerant of input that did not look a bit like C code. However, compilers have evolved, and some preprocessors are now more critical of their input streams.

The Solaris preprocessor behaves properly with `btxbst.doc`, and is the one that the author used for development, but the GNU **`gcc`** preprocessor issues warnings, and squeezes horizontal space, destroying code readability, while producing output that is usable by BibTeX. The **`clang`** preprocessor in recent BSD and Mac OS X (renamed macOS in 2016) releases rejects the `btxbst.doc` file entirely.

TO DO: Adapt my old **`lpp.awk`** as cpp replacement!

The four sample styles represent the most common practices of numbered and alphanumerically tagged references. Sorting is normally by `key` value or the family name of the first author, but the `unsrt.bst` style sorts entries according to their order of citation. That deplorable practice is used by some journals, and ensures that a later search for anything in the reference list is unnecessarily, and unreasonably, hard for the reader.

2 Deficiencies of the basic BibTeX styles

The year of the release of the four sample styles predated the Internet and the World Wide Web [**?**], so they had no possibility of displaying such things as *Digital Object Identifiers* (DOIs) and *Uniform Resource Locators* (URLs). Nor did they support the already-available *Chemical Abstracts* serial numbers (CODENs), *International Standard Serial Numbers* (ISSNs), *International Standard Book Numbers* (ISBNs), or *Library of Congress Call Numbers* (LCCNs), all of which are important handles for finding publications in online databases, library catalogs, publisher Web sites, and reseller product lists.

ISO 3297:2007, the fourth major revision of the standard that defines the ISSN system, introduced *Linking ISSNs* (ISSN-Ls), to serve as a single common handle for a serial published in multiple forms, such as print, electronic, and Braille, each of which requires a separate ISSN. Most commonly, the

Table 1: DOI adoption as measured by members of the BibTeX `@Article{...}` document class from the 1.27-million entries in the combined BibNet Project and TeX User Group bibliography archives. They can be found at `http://www.math.utah.edu/pub/bibnet` and `http://www.math.utah.edu/pub/tex/bib`, respectively.

Year	Articles	DOIs	DOI use	Year	Articles	DOIs	DOI use
2000	23856	9191	38.5%	2009	25358	16496	65.0%
2001	31474	9336	29.6%	2010	25984	17031	65.5%
2002	31233	9961	31.8%	2011	27155	19791	72.8%
2003	21380	9721	45.4%	2012	39054	33092	84.7%
2004	21591	11040	51.1%	2013	28183	19658	69.7%
2005	22476	13550	60.2%	2014	28647	18098	63.1%
2006	22945	14699	64.0%	2015	28237	17801	63.0%
2007	23940	16582	69.2%	2016	27273	17225	63.1%
2008	25187	17059	67.7%	2017	7804	4129	52.9%

ISSN-L value is identical to the ISSN of the print form, but there are occasional exceptions among the more than 1.6 million already-assigned ISSN values.

In late 1999, this author released a set of four extended styles, named as above, but with a prefix `is-`, for the International Standard Book and Serial Numbers. At about the same time, he also released similar extensions in the `xchicago.bst` style file, extending the standard `chicago.bst` file, an author–date package with quite different output formatting and citation practices. They allow formatting of, and output control for, all of the above identifiers, except for DOIs, which were first introduced in 2000. In 2016, another extension, `xxchicago.bst`, added DOI output in support of a complex book with a large bibliography [**?**]

At the time of writing this in 2017, DOIs are routinely assigned to journal articles and books by many publishers, including the largest commercial scientific publishers (Elsevier, Springer, and Wiley), and most professional scientific societies (ACM, ACS, AIP, AMS, APS, IEEE, SIAM, ...).

Table 1 gives a flavor of how widely used DOIs now are. Although we start the table entries for the year 2000, many articles before that year have been retroactively assigned DOI values; the oldest recorded so far [**?**] is from 1726! Almost half of the almost 680 000 recorded article entries published before 2000 have DOI values, and if it were feasible to look the remainder up in the DOI agency database, that fraction could be increased substantially. Also, remember that DOIs can be assigned to any document, not just journal articles: about 6% of the almost 49 000 recorded entries for books have DOI values.

3 Excursion: Decoding an ISBN

All characters in an *International Standard Book Number*, except possibly the last, are decimal digits, and there are ten characters, divided into four fields by single hyphens (or single spaces, but that practice is strongly deprecated):

- The first field is a digit string that identifies the language or country group: 0 and 1 are English, 2

is French, 3 is German, 4 is Japanese, 5 is Russian, 6 is unused, 7 is Chinese, . . . , and 99938 is the Republic of Srpska.

- The second field identifies the publisher. Big publishers get small numbers, and little ones get big numbers. For example, in the English-language groups, 03 is Holt Rinehart & Winston, 07 is McGraw-Hill, 13 is Prentice-Hall, 87942, 7695, 7803, and 8186 are IEEE, and 58113, 59593, and 89871 are ACM. The last examples show that when a smaller publisher exhausts its assigned ISBN space, it receives a new publisher number. You can even gauge the growth of some publishers by their publisher numbers: O'Reilly & Associates went from 56592 to 596, allowing 100 times as many books under the new number.
- The third field is a digit string identifying the book number within the publisher. Each format of a book (paperback, hardcover, large print, electronic, microfiche, . . .) receives a different book number, because each has separate cataloging, marketing, pricing, and stocking requirements.
- The fourth field is a base-11 check digit chosen from the set `[0-9Xx]`, with `x` equivalent to `X`.

By the early 2000s, it was clear that the supply of 10-digit ISBNs would soon be exhausted, so a new 13-digit format based on the *European Article Numbering (EAN)* scheme [**?**] was introduced, and became official in January 2007. Fields 2 and 3 of the ISBN-10 format are copied without change to fields 3 and 4 of the ISBN-13 format. Publishers may continue to use their existing 10-digit numbers, but must convert to the 13-digit form when they get a new publisher number assignment. A few years before 2007, many publishers began to print both numbers on back covers and in the front matter, and the book covers have separate barcodes for each.

The new form has five fields. The first is normally `978`, but that will become 979 when the supply of unique values is exhausted for at least one publisher. The remaining fields are as for 10-digit ISBNs, and the check digit in general differs between the two forms. As long as the new value begins `978-`, it has a unique old value as well, so many BIBTEX entries record both:

```
ISBN =    "0-88275-642-7",
ISBN-13 = "978-0-88275-642-4",

ISBN =    "0-8186-8857-2, 0-8186-4857-0 (microfiche),
           0-8186-8857-2 (casebound)",
ISBN-13 = "978-0-8186-8857-7, 978-0-8186-4857-1 (microfiche),
           978-0-8186-8857-7 (casebound)",
```

While library-catalog software transitions to support both 10-digit and 13-digit ISBN values, it is possible that one catalog recognizes or records only one form, and another catalog, only the other. Thus, humans find it useful to have both available for searching.

So far, only *two* entries in the archives mined in **Table 1** have been found with the new `979-` ISBN-13 prefix. Their existence, however, indicates that both 10- and 13-digit ISBN data *must* be supported by bibliography styles. For completeness, we include the two with `979-` prefixes in our reference list [**?**, **?**].

4 Modernizing standard BIBTEX styles

Many publishers now strongly encourage the incorporation of DOI data in publication reference lists, because it makes it easy to enhance their publication archives on the Web with live hyperlinks to cited

documents, and also to validate author-supplied reference-list data.

LATEX users can themselves provide live hyperlinks in their own documents if they have a `\usepackage{url}` command in their document preambles, they wrap Web addresses in `\url{...}` macros, and they typeset with **`pdflatex`** instead of **`latex`**.

This author produced a new set of BIBTEX style files in Spring 2017 to further modernize the basic four with support of DOI and ISSN-L data. They are the main focus of the remainder of this document.

DOI data *could* be recorded in **`URL`** fields, which are already supported by the `is-*.bst` styles, and a few others. However, that is a serious abuse of logical markup, and both the DOI registration organization at `http://www.doi.org/`, and the 16th (2010) and 17th (2017) editions of the famous *The Chicago Manual of Style* [**?**, **?**], widely used in North America, recommend a different approach. The DOI should appear *absolutely last* in the formatted reference-list item, prefixed by the string *doi:*, and with the protocol and host portion, if any, stripped from a DOI in URL form. In addition, no punctuation should follow the DOI value, to avoid confusion with that value. Thus, a BIBTEX entry with the assignment

```
DOI =          "http://dx.doi.org/10.1109/40.540",
```

should produce in the `.bbl` file that is read by the `\bibliography{...}` command a reference-list item that, when typeset, ends with

```
doi:10.1109/40.540
```

For roughly the first fifteen years of use of DOIs, the mapping of DOI value to Web address was consistently easy: just prefix `http://dx.doi.org/` to the part that *always* begins with `10.`, usually followed by four or five decimal digits, a slash, and a publisher-dependent string.

However, recently new Web address forms of DOIs are showing up in publisher metadata, including at least these:

```
http://doi.acm.org/                    https://doi.acm.org/
http://doi.ieeecomputersociety.org/    https://doi.ieeecomputersociety.org/
http://doi.org/                        https://doi.org/
http://dx.doi.org/                     https://dx.doi.org/
http://www.pnas.org/cgi/doi/           https://www.pnas.org/cgi/doi/
```

The `s` in the protocol prefix indicates a *secure* (encrypted) network connection.

In 2017, the DOI agency recommended switching to `https://doi.org/` as the standard Web prefix, and our bibliography archives have all been changed to use that form. Some large organization, including ACM, IEEE, and the US *National Academy of Sciences*, supply their own Web hostname, instead of that of the DOI agency. That host serves as a redirection site, ensuring that even if the publication owner changes, forcing the URL to change as well, the DOI must remain intact, because it is a *persistent* identifier that should always lead to the document, independent of who owns it today, and where on the Internet the definitive copy of the document is currently stored. We recommend replacing all publisher-specific Web prefixes with that of the DOI agency.

In the hundreds of existing BIBTEX styles without DOI support, the contents of any **`note`** field value are what normally appears *last* in a reference-list item. The recommended DOI treatment changes that long-standing practice, but standardizing the location of Web addresses in reference lists seems worthwhile of adoption by everyone, even if a few publishers might occasionally recommend different positioning, as the ACM has done (but I've strongly urged them to follow the DOI and CMS recommendations).

5 Other supported fields

Besides the already-discussed **CODEN**, **ISSN**, **ISBN**, and **URL** BibTeX field names, the 1999 `is-*.bst` family provided support for a few additional fields: **day**, **bookpages**, and **price**,

The `is-*.bst` styles can be generated with special code controlled by the preprocessor symbol `_NUMERIC_SUFFIXES` to handle the case of matching citation labels, replacing the default of supplying single-letter disambiguating suffixes to using numeric suffixes, `-1`, `-2`, `-3`, ... That allows handling of bibliographies with more than 26 label collisions. In the absence of numeric-suffixes code selection, the label suffix code was also changed to switch from alphabetic to numeric after the 26 letters are used.

The new styles for 2017

`x-abbrv.bst` `x-alpha.bst` `x-plain.bst` `x-unsrt.bst`

are supersets of the `is-*.bst` family. The new files recognize **DOI** fields whose values are always formatted last in reference-list items, **ISBN-13** fields whose values are typeset following any 10-digit ISBN data, and **ISSN-L** fields.

The latter are handled slightly differently than the **ISSN** fields: an ISSN-L value is suppressed when it is identical to the ISSN value. Compare these two references: [**?**] where they differ, and [**?**] where both are identical.

The new styles also handle **articleno** and **pagecount** fields that are described on page 8.

Table 2 shows how commonly used are the extended fields in our bibliography archives, and Table 3 gives similar data for the document classes known to BibTeX.

6 Problematic DOIs and URLs

The Digital Object Identifier agency assigns the initial `10.nnnn/` string to a publisher. However, when the DOI system was first introduced, its documentation was vague about the format of the DOI contents that follow the publisher prefix. The expectation was that publishers would supply relatively short document-specific identifiers, so that DOIs could be manually typed, or dictated over the telephone, and would occupy minimal space in reference lists.

Unfortunately, some publishers did not behave as expected, and produced long DOIs containing punctuation and nonalphanumeric symbols. Such DOIs can pose difficulties for document formatters, such as TeX, that support programmatic manipulation of text. For example, troublesome 8-bit characters in URLs can be replaced by a percent and two hexadecimal digits that represent the position of the character in the computer character set. However, in TeX documents, a percent is the normal comment-start character. The percent can be escaped with backslash to mean a literal percent, but that only works for inline use: a similar string passed through one or more levels of macro calls is almost certain to be corrupted.

Figure 1 shows how prevalent long DOIs are in a large corpus of bibliographic data.

Fortunately, there are at least four online services for shortening long URLs and DOIs:

`http://bit.ly` `http://goo.gy` `http://tinyurl.com` `http://shortdoi.org`

The first three work well, but their longevity is questionable, and they only produce URLs.

The fourth, from the DOI agency itself, is recommended for DOI (but not URL) values, because a short DOI that it issues is as persistent as the original.

Table 2: Usage of extended fields in the bibliography archives.

Field	Count	Field	Count
articleno	15775	**ISSN**	1145638
bookpages	1121	**ISSN-L**	1052134
CODEN	1050203	**LCCN**	45750
day	175255	**pagecount**	1076033
DOI	622564	**price**	9244
ISBN-13	80770	**URL**	709599
ISBN	80737		

Table 3: Document classes and their use in the bibliography archives.

Class	Count	Class	Count
`@Article{...}`	1140618	`@MastersThesis{...}`	1425
`@Book{...}`	49016	`@Misc{...}`	5885
`@Booklet{...}`	65	`@Periodical{...}`	440
`@InBook{...}`	117	`@PhdThesis{...}`	996
`@InCollection{...}`	5745	`@Proceedings{...}`	17356
`@InProceedings{...}`	41205	`@TechReport{...}`	11091
`@Manual{...}`	1912	`@Unpublished{...}`	2598

For example, consider this 56-character DOI assigned to a Wiley journal article:

```
10.1002/(SICI)1097-461X(1996)57:1<3::AID-QUA1>3.0.CO;2-1
```

From its Web-address form, the first three services produce the URLs

```
http://bit.ly/2mH3Zl3  http://goo.gl/7uHJMG  http://tinyurl.com/k4y4pz7
```

and the DOI-agency service reports

```
shortDOI&reg;
Your request was processed. The previously created shortcut for
10.1002/(SICI)1097-461X(1996)57:1<3::AID-QUA1>3.0.CO;2-1 is the
handle:

    10/b8sr3k

The shortcut HTTP URI is:

    http://doi.org/b8sr3k

This shortcut will return the same results as
    http://dx.doi.org/10.1002/(SICI)1097-461X(1996)57:1<3::AID-QUA1>3.0.CO;2-1,
```

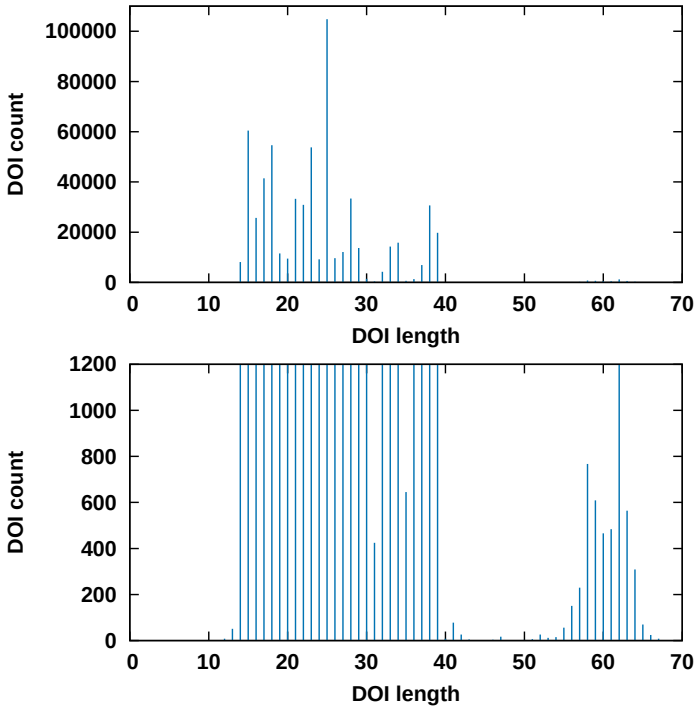

Figure 1: Counts of DOI lengths (excluding the Web-address prefix) from 612 737 entries with recorded DOIs in the combined BibNet Project and TEX User Group bibliography archives. The lengths range from 12 to 70 characters, and the most common length is 25.
The lower plot shows the same data, but on a truncated vertical scale.

```
and
    doi:10/b8sr3k
can be used in place of
    doi:10.1002/(SICI)1097-461X(1996)57:1<3::AID-QUA1>3.0.CO;2-1.
```

Later documentation from the DOI agency improved the situation, and most publishers quickly changed their DOI-identifier algorithms to produce shorter ones. However, the already-issued long ones are 'permanent', and are not retroactively remapped to short DOIs, except on a case-by-case basis, as we did here.

Should you encounter a DOI or URL containing a vertical bar, the delimiter character used in the output `.bbl` file to mark `\url|...|` command arguments, just replace it by `%7C`, the hexadecimal representation of its position in the ASCII and Unicode character sets. In the more than 600 000 DOIs in our bibliography archives, no such case has been recorded. Among about 710 000 URL fields, only four with vertical bars were found, all for Harvard University's Hollis Library catalog, and they were easily rewritten.

7 Article numbers and page counts

For hundreds of years, journals traditionally have been published in volumes, usually divided into separate issues, with page numbers increasing uniformly from 1 through the volume, or else reset to 1 at each issue.

Volumes and issues might be regular, with a new volume each year, and issues appearing half-yearly, quarterly or seasonally, monthly, or weekly. However, that regularity is broken by many journals, so it is important to record in BibTeX entries values for the six fields **`volume`**, **`number`**, **`pages`**, **`day`**, **`month`**, and **`year`**. A typeset reference reporting those values might then contain a string like **123**(7):723–752, July 4, 1976.

When page counts increase monotonically through volumes, it is possible for software to look for gaps in coverage in journal-specific bibliographies, and such checks have often uncovered errors and holes in publisher metadata. When page counts are reset in each issue, there are many more opportunities for lost publications.

Sometime after the year 2000, several publishers in computer science and the physical sciences changed to a new style of article identification, where there are now two new fields supported in the `x-*.bst` styles: **`articleno`** and **`pagecount`**. In computer-science journals, the article number usually increases from 1 in each volume, but in the physical-science journals, article numbers are 4- to 6-digit values of uncertain origin, and that do not necessarily increase uniformly across volumes. Page-gap checks are then impossible, making the new style a serious mistake in this author's view.

Here is an example for one journal that does not reveal page counts in its metadata:

```
journal =      "Theoretical Chemistry Accounts",
volume =       "131",
number =       "8",
articleno =    "1257",
```

Here is another that does:

```
journal =     "Physical Review A (Atomic, Molecular, and Optical
               Physics)",
volume =       "85",
number =       "3",
articleno =    "034501",
pagecount =    "4",
```

The new numbering conventions are a problem for all existing bibliography styles and databases, because they cannot readily handle them.

A reasonable solution that has been widely used in our bibliography archives is to reencode the two values as a compound page range, like these assignments for our two examples:

```
pages =        "1257:1--1257:??",

pages =        "034501:1--034501:4",
```

However, because the physical-science journal production is large, with some journals appearing weekly, and with 20 000+ pages per year, bibliography styles really need to be adapted to the new system.

Examination of current citation practices for the **articleno** and **pagecount** fields shows that ACM computer-science journals format output like this:

> . . . , *International Journal of Reconfigurable Computing* 2012, Article 12 (Jan. 2012), 1 page.
>
> . . . , *ACM Transactions on Reconfigurable Technological Systems* 8, 4, Article 23 (Sept. 2015), 22 pages.
>
> . . . , *ACM Computing Surveys* 39, 4, Article 11 (Nov. 2007).

By contrast, an APS physics journal might have this more compact form:

> . . . , Rev. Sci. Instrum. 82, 073109 (2011).
>
> . . . , Phys. Rev. A 87, 062327 (2013).

Notice that the **day**, **month**, **number**, and **pagecount** values are omitted, and that dropping issue numbers produces serious location ambiguities for journals that begin each issue on page 1.

The `x-*.bst` files have therefore been adjusted to produce something like

```
\showVOLUME{8}(\showNUMBER{4}), \showARTICLENO{23}\showPAGECOUNT{22}
```

in the output `.bbl` file, but only when the **pages** field is empty. Notice the absence of space between the last two macros. Otherwise, they output the **pages** values, ignore the **articleno** and **pagecount** values, and issue a warning that they have done so.

The wrapper macros can then be defined with their default values

```
\def \showARTICLENO    #1{Article #1}
\def \showPAGECOUNTONE #1{, #1 page}
\def \showPAGECOUNT    #1{, #1 pages}
```

for the computer-science style shown in the bibliography [**?**, **?**, **?**], or redefined by the user to

```
\def \showARTICLENO #1{#1}
\def \showPAGECOUNT #1{}
```

for the compact physics style.

There are, of course, other differences in field order among the sample journals, and supporting their formats properly would require adding new style files, readily derivable from the `x-*.bst` family.

8 Implementation considerations

BibTeX styles are written in a markup language expressed in reverse-Polish form: operands (function arguments) appear *before* operators (functions) on a dynamic call stack, and are placed there anonymously, and possibly, not even by code close to the point of the stacking of the operator name. BibTeX has no style-file debugger, and extremely limited string processing operations. Output is asynchronous, and may or may not happen when an output operator is executed. Debugging BibTeX style files is therefore painful, and frequently frustrating, because there is no operator that can print the call stack without destroying it! Neither is it possible to save and restore the stack, nor to determine its depth, nor to display a function-call traceback. Also, when the stack has an incorrect number of entries, BibTeX's error message reports the line number of the `ITERATE` command at the end of the `*.bst` file, rather

than the function location where the error happened. Such errors are easy to make with a missing, or extra, asterisk (BIBTEX's string-concatenation operator), and are often difficult to find in the style file because of the lack of precise error-location diagnostics.

BIBTEX users are encouraged to record DOI values as Web addresses, rather than as bare strings beginning `10.`..., because other people are then more likely to recognize that the address can be pasted into a Web browser to find the document online. Although publishers are enthusiastic about the benefits of DOIs, it is unlikely that more than a small minority of human readers of documents know what a DOI is, and how to convert it to a Web address. Thus, it should be the job of the BIBTEX style file to strip the protocol and hostname from the DOI value, and format the remainder for use in the reference list.

In a language with good string-processing facilities, such as `awk`, the prefix-stripping job is a trivial function call:

```
gsub("https?://[^/]*/", "", doi)
```

That does a global substitution of the prefix that matches the regular-expression pattern up to the slash before the DOI, replacing the matched string with an empty string.

Life is much harder in the BIBTEX style-file language. Here is a fragment of the function common to the new `x-*.bst` files that formats the DOI value:

```
FUNCTION {format.doi}
{
  %% For clarity, we assign the DOI value, or an empty string,
  %% to the temporary variable t, then strip common prefixes.
  doi empty$
    { ""  }
    { doi }
  if$ 't :=

  %%-------------------123456789.123456789.123456789.123456789.
  t #1 #28 substring$ "http://www.pnas.org/cgi/doi/" =
    { t #29 t text.length$ #28 - substring$ }
    { t }
  if$ 't :=

  ...

  %%-------------------123456789.123456789.123456789.123456789.
  t #1 #16 substring$ "https://doi.org/" =
    { t #17 t text.length$ #16 - substring$ }
    { t }
  if$ 't :=

  doi empty.or.unknown
    { "" }
    {
      newline$
```

```
        new.block
        "\newblock \ifshowDOI {\showDOI \href {https://doi.org/" t *
                   "} {" * t * "}}\ifshowDOIPERIOD . \fi \fi " *
      }
    if$
  }
```

The elided block of `if$` statements has similar four-line constructs for each of the patterns listed on page 4 of this document.

The first `if$` statement is executed like this:

- Push the value of the variable `doi` onto the call stack.
- Push the `empty$` operator onto the stack.
- Execute the top operator, popping its single operand, and pushing a value *1* (for true), or *0* (for false) back onto the stack.
- Push the next 12 tokens (from the first open brace to the second close brace) onto the stack.
- Push the `if$` operator onto the stack.
- Execute that operator, popping three groups of items: the test outcome, the first braced group, and the second braced group. If the test value means *true*, execute the first braced group (the *then* part); otherwise, execute the second braced group (the *else* part).

The first line of the second statement block matches the first 16 characters of the DOI value against a known prefix string, and the equality test operator pushes *true* or *false* onto the stack. The *then* part pushes onto the stack the substring of the DOI that follows the matched prefix; the *else* part pushes the full DOI value. The final `'t :=` assigns the pushed string value to the temporary variable `t`.

Such programming is tedious and verbose, and much less flexible than the `awk` one-liner, which matches *any* reasonable protocol and hostname prefix.

While it was intended by the DOI designers that a given document should have only a single unique DOI value, in practice, a few hundred documents each year in our bibliography archives are found to have two DOI values, and one or two per year have three or more DOI values. Once again, the `awk` one-liner would handle those multivalued cases properly, but our BibTeX style-language code only reduces the first DOI found.

The test entries in the reference lists at the end of this document exercise all of the prefix variants, as well as cases with two or more DOI values, and with and without prefixes. However, they also exhibit a small flaw in that handling: they enclose the entire DOI field with a TeX `\url|...|` macro, and that macro has been coded to discard spaces, so the semicolon-space separators are reduced to bare semicolons. With more style-file programming, that could be fixed, because we already handle URL field values one wrapped URL at a time. However, the reference lists for most journals would be expected to have only a single value following a final `doi:`, and we recommend therefore that the DOI field value be restricted to a single object identifier in bibliographic data expected to be used for journal publication.

Another possible way to fix the disappearing-space problem is to supply an option to the `url` package:

```
\usepackage[obeyspaces]{url}
```

However, that solution is imperfect, because BibTeX introduces additional line wrapping and indenting spaces in the generated `.bbl` file, and each of them is then preserved in the typeset bibliography; the extra space is noticeable, and likely, unwanted.

9 Formatting a BibTeX entry for a reference list

Only a few of the sample reference-list items are explicitly cited in this document: we include them all with a simple `\nocite{*}` command that creates implicit citations for every entry in the BibTeX database files listed in the `\bibliography{...}` command. Most refer to real documents, but a few are obviously fictitious, and included just to demonstrate how their data are formatted. Here is just one of those entries from four similar ones [**?**,**?**,**?**,**?**]:

```
@Article{Such:2099:FTc,
  author =       "None Such",
  title =        "Fake title with multiple standard {DOIs}",
  journal =      "Bogus Journal",
  volume =       "3",
  number =       "4",
  pages =        "5--6",
  year =         "2099",
  CODEN =        "YYYYY",
  DOI =          "http://doi.org/10.1109/XX.2099.56a;
                 http://doi.org/10.1109/XX.2099.56b",
  ISSN =         "8888-8889 (print), 8888-8888 (electronic)",
  ISSN-L =       "8888-8889",
  bibdate =      "Thu Apr 06 11:58:55 2017",
  price =        "US\$33.00",
  URL =          "http://users.example.com/~such/XX.2099.56",
  acknowledgement = ack-nhfb,
}
```

Its formatted item in the generated `.bbl` file for the x-plain style looks like this:

```
\bibitem{Such:2099:FTc}
  \ifshowBIBTYPE \showBIBTYPE{article}{Such:2099:FTc} \fi \showAUTHORRAW{None
  Such}\showAUTHOR{None Such}.
\newblock \showTITLE{Fake title with multiple standard {DOIs}}.
\newblock {\em \showJOURNAL{Bogus Journal}}, \showVOLUME{3}\penalty 0
  (\showNUMBER{4}):\penalty 0 \showPAGES{5--6}, \showYEAR{2099}. \ifshowCODEN
  {\showCODEN{YYYYY}}. \fi \ifshowISSN {\showISSN{8888-8889 (print), 8888-8888
  (electronic)}}. \fi \ifshowISSNL {\showISSNL{8888-8889}}. \fi \ifshowPRICE
  {\showPRICE{US\$33.00}}. \fi
\newblock \ifshowURL {\showURL
  \url{http://users.example.com/~such/XX.2099.56}}. \fi
\newblock \ifshowDOI {\showDOI \href {https://doi.org/10.1109/XX.2099.56a;
  http://doi.org/10.1109/XX.2099.56b} {10.1109/XX.2099.56a;
```

```
http://doi.org/10.1109/XX.2099.56b}}\ifshowDOIPERIOD . \fi \fi
```

The typeset reference-list item then looks somewhat like this:

[**?**] None Such. Fake title with multiple standard DOIs. *Bogus Journal*, **3**(4):5–6, 2099. CODEN YYYYY. ISSN 8888-8889 (print), 8888-8888 (electronic). ISSN-L 8888-8889. US$33.00. URL `http://users.example.com/~such/XX.2099.56`. doi:10.1109/XX.2099.56a; http://doi.org/10.1109/XX.2099.56b.

Let us analyze the TEX markup in the `.bbl` file item:

- The `\bibitem` command receives the citation label as its mandatory single argument, and expands it to a bracketed item number (in the numbered styles) for typesetting.
- The next line, with the `\showBIBTYPE` macro, is normally discarded by the default setting of the conditional. However, on occasion, it can be made to prepare something useful — an annotated typeset bibliography that identifies the type and label for each entry, as we do in the reference list on **page ??** with these definitions:

```
\newlength  {\bibrightmargin}
\setlength  {\bibrightmargin} {25ex}
\newcommand {\showBIBTYPE} [2]
{%
    \marginpar{%
                  \highlightcolor
                  \kern 0.4\baselineskip
                  \mbox{}%
                  \kern -\bibrightmargin
                  \parbox{0pt}{{\small #1}\\{\footnotesize \bf #2}}%
              }%
}
\showBIBTYPEtrue
```

  The right-margin adjustment is saved in a length variable, because it is needed later when the bibliography is typeset, like this:

```
\begin{list}{}{\leftmargin = 0ex \rightmargin = \bibrightmargin}
    \item
    \bibliography{bst}
\end{list}
```

- The action of the `\newblock` commands depends on the document style; they generate additional rubber space (TEX glue), or else paragraph breaks.
- The `\penalty 0` commands tell TEX that a line break at that point adds no penalty to the paragraph badness, offering more places for line breaking in material that is otherwise rather dense.

- Every output field value is wrapped in a macro that identifies its origin from its BIBTEX entry. Together with the `\showBIBTYPE` command, there is now sufficient information in the `.bbl` file to reconstruct a BIBTEX database file that can later be used to create a new `.bbl` file, *possibly in a different style*.

  Had this capability been present from the beginning of BIBTEX, then it would have been easy for publishers to accept author-provided bibliographies in any available BIBTEX format, then automatically convert them to the style preferred for the journal of publication. Incompatibilities in reference-list formatting among journals have long been a nuisance for authors, editors, and journal production staff. Perhaps these new styles might encourage journal publishers to recommend their use in article submissions, making life easier for all involved.

  The availability of identifiable DOI data in entries would also allow automated checking by publisher software of author-supplied reference-list items, both for completeness, and for errors. It also facilitates adding hyperlinks to online reference lists. Some publisher make those lists freely available at journal Web sites, just as they often do with article abstracts, even when access to complete article content may require a journal or database subscription, or online payment.

- Each optional output field value is set as a sentence, wrapped like this:

  ```
  \ifshowXXX {\showXXX{xxx-value}}. \fi
  ```

  Thus, the complete sentence disappears when the selector is false. Otherwise, the sentence appears, but the embedded `\showXXX{...}` that controls the formatting is braced, so that any style changes that it makes, such as in color or font, are limited to the braced group. The period is intentionally outside that group.

  There is no punctuation *between* the conditionals, and if they are false, they produce no output space.

  The definitions inside the `.bbl` file are written in plain TEX commands, rather than with LATEX equivalents, because BIBTEX output must be usable for all TEX variants.

- The final `\ifshowDOI` ... `\fi` wrapper allows the user to control DOI output with the simple commands `\showDOItrue` and `\showDOIfalse`, *without* having to change either the human-generated BIBTEX database file, or the reference list that BIBTEX automatically formatted according to the specified style and database sources.

- The `\showDOI` portion expands to the default value `doi:\penalty 0`, allowing another opportunity for a line break. If no line break is needed, then because TEX ignores whitespace after a command name or its numeric argument, no space occurs between the colon and the reduced DOI value `10.1109/XX.2099.34a`.

- The prefix reduction happens only on the first DOI value, so the second is preserved intact, and as noted earlier, the space after the semicolon disappears in the typeset output because of a decision by the author of the now widely used, and thus, immutable, `\url{...}` command. We describe on **page 11** why the `obeyspaces` option for the `url` package is probably not an acceptable fix.

- The 16th edition of *The Chicago Manual of Style* (2010) shows a period following each DOI, even though that introduces unnecessary confusion. If you really want that terminal period, just put the command `\showDOIPERIODtrue` before the bibliography, or `\showDOIPERIODfalse` if you do not want it.

10 Comparison of the extended styles

To give a flavor of the reference-list formatting in each of the four extended styles, here is an example of an `@Article{...}` entry in each, with the extended fields suppressed:

x-abbrv

[27] C. D. Linkletter, P. Ranjan, C. D. Lin, D. R. Bingham, W. A. Brenneman, R. A. Lockhart, and T. M. Loughin. Compliance testing for random effects models with joint acceptance criteria. *Technometrics*, **54**(3):243–255, Aug. 2012. See erratum [**?**].

x-alpha

[LRL+12b] Crystal D. Linkletter, Pritam Ranjan, C. Devon Lin, Derek R. Bingham, William A. Brenneman, Richard A. Lockhart, and Thomas M. Loughin. Compliance testing for random effects models with joint acceptance criteria. *Technometrics*, **54**(3):243–255, August 2012. See erratum [**?**].

x-plain

[28] Crystal D. Linkletter, Pritam Ranjan, C. Devon Lin, Derek R. Bingham, William A. Brenneman, Richard A. Lockhart, and Thomas M. Loughin. Compliance testing for random effects models with joint acceptance criteria. *Technometrics*, **54**(3):243–255, August 2012. See erratum [**?**].

x-unsrt

[38] Crystal D. Linkletter, Pritam Ranjan, C. Devon Lin, Derek R. Bingham, William A. Brenneman, Richard A. Lockhart, and Thomas M. Loughin. Compliance testing for random effects models with joint acceptance criteria. *Technometrics*, **54**(3):243–255, August 2012. See erratum [**?**].

11 Missing values

Extensive experience with the bibliography archives cited in **Table 1** shows that there is a definite need to distinguish between an empty field value, and one that should not be empty, but whose value is still unknown. The convention adopted in those archives is to mark the gaps in knowledge with assignments like these:

```
volume =       "??",
pages =        "123--??",
publisher =    "????",
address =      "????",
month =        "????",
```

The `x-*.bst` files support such markup by treating any field value that begins with *two question marks*, or contains only whitespace, as if the field assignment were omitted entirely, or were present, but with an empty string value. Of the given samples, only the **pages** value would be preserved in the output `.bbl` file.

That proves convenient, because the bibliography entries are then usable, even though they are known to be incomplete.

A careful user should search the `.bbl` and `*.bib` files to find any instances of consecutive question marks, and then make reasonable efforts to find the missing data, and repair the BibTeX database files.

A set of fictitious entries [**?**, **?**, **?**, **?**, **?**, **?**, **?**, **?**, **?**, **?**, **?**, **?**, **?**, **?**] in the reference list shows what happens when fields are omitted, or their values are either empty, or else begin with two or more question marks that indicate unknown, and still-to-be-found, values. Careful examination of those entries show that they are all well-behaved, and except for their titles, give no indication that expected data are missing.

Note to the author: Perhaps I should prepare a set of companion bibliography styles, `x-debug-*.sty`, that output distinctive phrases where expected data are absent. A sample entry from such a style might then look like this:

```
\def \showMISSING #1{{\color{red} \large \bf [??#1??]}}
...
\bibitem{Zucchina:2050:FTP}
  \ifshowBIBTYPE \showBIBTYPE{article}{Zucchina:2050:FTP} \fi
  \showAUTHORRAW{Asparago Zucchina}\showAUTHOR{Asparago Zucchina}.
\newblock \showTITLE{Fake title with pages unknown}.
\newblock {\em \showJOURNAL{Bogus Journal}}, \showVOLUME{1}\penalty 0
  (\showNUMBER{2})\showMISSING{pages}, \showMONTH{December} \showDAY{31},
  \showYEAR{2050}.
  \ifshowCODEN {\showCODEN{ZZZZZ}}. \fi \ifshowISSN {\showISSN{9999-9998
  (print), 9999-9999 (electronic)}}. \fi \ifshowISSNL {\showISSNL{9999-9998}}.
  \fi
\newblock \ifshowURL {\showURL
  \url|http://docs.example.com/zanetti/bogusj.1.2.3.4|}. \fi
  \showNOTE{This is a note about this sample article.}
\newblock \ifshowDOI {\showDOI \href {https://doi.org/10.9999/bogusj.1.2.3.4}
          {10.9999/bogusj.1.2.3.4}}\ifshowDOIPERIOD
  . \fi \fi
```

With a suitable definition of the `\showMISSING` command, it would typeset like this:

[**?**] Asparago Zucchina. Fake title with pages unknown. *Bogus Journal*, **1**(2) **[??pages??]**, December 31, 2050. CODEN ZZZZZ. ISSN 9999-9998 (print), 9999-9999 (electronic). ISSN-L 9999-9998. URL `http://docs.example.com/zanetti/bogusj.1.2.3.4`. This is a note about this sample article. doi:10.9999/bogusj.1.2.3.4

Such a display could be of great value to careful authors, especially those with large and complex bibliographies.

---

12 Deviations from the four standard styles

The new 4100-line `xbtxbst.doc` file from which the four new extended styles are extracted is a heavily edited copy of the original 2500-line `btxbst.doc` file that produced BibTeX's original four sample styles. Output field order is reasonably similar, except for the addition of the 11 new optional fields.

However, there are some significant changes in field handling in the four new styles:

- All fields are wrapped in distinct macros to allow further customizations of typeset appearance, automated data extraction, and BibTeX file reconstruction.
- Several document types that previously ignored **pages** values now output them.
- With preprocessor-time definition of the symbol that enables recognition of code for handling **bookpages** value, that useful field is supported, and optionally output to `.bbl` files. For example, it is convenient to record in the `@InXXX{...}` entries both **bookpages** and **pages** values. The first provides a value for the complete volume, and the second for the document within the volume. Having both makes it easy later to construct separate `@Book{...}` or `@Proceedings{...}` entries for the complete volume, allowing it to be cited separately, as is often desirable.
- Formatting of the 14 standard document types, and the new 15th `@Periodical{...}` type, employs a common output function, so they all recognize the extra field names, and format their values in the same order. In most existing BibTeX style files, the particular selection of output fields is eclectic, and hard for users to predict without consulting documentation, or worse, style-file source code.
- The original four sample styles treat the **address** fields for `@Proceedings{...}` and `@InProceedings{...}` differently from all other document types, and output the address before the publisher. That practice was based on an unfortunate idea that the **address** field for conference papers should reflect the meeting location, rather than the publisher address. That historical mistake is rectified in the new styles, and the markup in our bibliography archives has about 65 000 entries where the **address** field refers to the publisher, just is it does for all other document types that support **institution**, **organization**, **publisher**, or **school** fields. The conference location is generally recorded in the volume title, or about 5% of the time in a **note** field, as in these examples:

```
title =         "{2016 IEEE 23nd Symposium on Computer Arithmetic
                (ARITH 2016), Santa Clara, California, USA, 10--13 July
                2016}",

title =         "Numerical methods for partial differential equations",
note =          "Papers from the International Congress held in
                Marrakech, September 14--18, 1998.",

title =         "Preconditioning techniques for large sparse matrix
                problems in industrial applications",
note =          "Papers from the International Conference (SPARSE '99)
                held at the University of Minnesota, Minneapolis, MN,
                June 10--12, 1999, Numer. Linear Algebra Appl. {\bf 7}
```

```
                (2000), no. 7--8.",
```

- The **type** field value in `@MastersThesis{...}` and `@PhdThesis{...}` is no longer titlecased. In this author's view, that was a serious flaw in the original four styles, and has led to many downcasing errors for academic degrees in published reference lists because the BibTeX file had something like

  ```
  type =          "Ph.D. thesis",
  ```

  instead of the brace-protected variant

  ```
  type =          "{Ph.D.} thesis",
  ```

  resulting in the incorrect output *Ph.d. thesis*.

  BibTeX's case-changing code is hard-coded into the program, rather than being implemented in the style-file language. Many errors of bibliography markup, and typeset reference-list formatting, would have been eliminated if BibTeX had been written to refuse to change lettercase in words containing an embedded uppercase letter, and also, in all words inside math mode. That way, titles like

  ```
  title =         "Adobe's InDesign hits the market",

  title =         "Anomalous $\delta$ functions, ordinary $\Delta$
                  operators, and protected $\Delta$ operators
                  in Bose--Einstein statistics",
  ```

  would not have required the protecting braces that are currently needed, as in

  ```
  title =         "{Adobe}'s {InDesign} hits the market",

  title =         "Anomalous $\delta$ functions, ordinary {$\Delta$}
                  operators, and protected {$\Delta$} operators
                  in {Bose--Einstein} statistics",
  ```

  The first word of titles is titlecased in many style files, and so, for our two examples, that word is unchanged, but we have made it an easy-to-remember rule that *all* proper nouns are braced in titles. That way, should a title begin with *McTavish*, it will not become the erroneous *Mctavish* in the reference list.

  Similarly, all titles in German, and pre-1948 Danish, languages where nouns are always capitalized, have easily supplied protecting outer braces, as in these titles of famous doctoral theses (with helpful language identification, and English translation):

  ```
  author =        "Albert Einstein",
  title =         "{Eine Neue Bestimmung der Molek{\"u}ldimensionen}.
                  ({German}) [{A} new determination of molecular
                  dimensions]",
  ```

Table 4: Selection macros for output control of extended and fields DOI punctuation. The highlighted first pair are special purpose, and rarely needed.

\showBIBTYPEfalse	\showBIBTYPEtrue
\showBOOKPAGESfalse	\showBOOKPAGEStrue
\showCODENfalse	\showCODENtrue
\showDOIfalse	\showDOItrue
\showDOIPERIODfalse	\showDOIPERIODtrue
\showISBNfalse	\showISBNtrue
\showISSNfalse	\showISSNtrue
\showISSNLfalse	\showISSNLtrue
\showLCCNfalse	\showLCCNtrue
\showPRICEfalse	\showPRICEtrue
\showURLfalse	\showURLtrue

```
year =          "1905",

author =        "Niels Bohr",
title =         "{Studier over Metallernes Elektronteori}. ({Danish})
                [{Studies} on the electron theory of metals]",
year =          "1911",
```

13 Customizing the reference list

The `x-*.bst` styles have been carefully written so that inclusion of values for the several new field names remains under user control. *Neither the* BibTeX *databases, nor the reference list itself, need manual editing to achieve that goal.* Instead, the user simply selects the visibility of any particular extended field by executing one or more of the TeX selection macros in **Table 4** somewhere before the reference list is output. The default value of all of them is the `true` variant, except for the highlighted `showBIBTYPE`, which is `false`, because it is intended for producing specially marked bibliographies, such as that on **page ??** of this document. Good places for such settings are the document preamble, or just before the `\bibliography{...}` command that typesets the references.

For user convenience, the commands

`\hideOPTIONAL` `\showOPTIONAL`

can be used to select all of the false, or all of the true, variants. The special purpose, and rarely used, `showBIBTYPE` variants are not affected by those commands.

Here are the relevant portions of a LaTeX file that uses one of the extended styles, and turns off output of one class of field values:

```
\documentclass {article}
```

```
\usepackage {x-bst}

\begin {document}

...

\bibliographystyle {x-plain}
\showPRICEfalse
\bibliography {myrefs,herrefs,hisrefs,ourrefs,theirrefs}

\end {document}
```

The `\usepackage{x-bst}` command has the job of ensuring that all of the wrapper conditional macros needed in the bibliography are properly defined before typesetting begins. The x-bst package automatically includes the url package if it has not yet been loaded.

Each of the extended fields is formatted by a simple macro that, if it is not already known, is defined near the start of the `.bbl` file output by BIBTEX. That gives the user additional opportunities for customization. For example, you might wish to highlight DOI values with color, or change their font family or size. Here is how to do that in the document preamble:

```
\usepackage {color}
\newcommand {\showDOI} { doi:\penalty 0 \small \color{blue}}
```

DOIs and URLs are output inside braced groups in the `.bbl` file, so that any formatting changes are limited to their group. In our example, the color change affects only the DOI value.

Similarly, other fields can be given personal definitions, such as these:

```
\newcommand {\showISBN} [1] {ISBN {\bf #1}}
\newcommand {\showLCCN} [1] {LCCN \texttt{#1}}
```

14 Using the x-*.bst family in plain TEX

We noted on page 14 that BIBTEX bibliography style files should be written to require only plain TEX markup, so that they can be used with all TEX variants. Here is what a plain TEX file might look like to typeset just the bibliography of this document:

```
\input x-bst
\input citesort.sty  % optional to sort and range numeric references
... optional commands and prose ...
\nocite{*}
\bibliographystyle{x-plain}
\bibliography{bst}
\vfill
\eject
\bye
```

The `x-bst.tex` file automatically inputs the `btxmac.tex` file if the command `\bibliographystyle` is not already defined; that standard file provides the necessary support for using BIBTEX in plain TEX.

Plain TEX cannot use the LATEX url package, so instead, `x-bst.tex` includes a simpler package that is usable with all TEX variants: `path.sty`. It then issues a `\let \url = \path` command that gives `\url` the definition of `\path`, a command that works like the LATEX `\verb` command, except that it permits hyphenless line breaking after text ending with any of a user-redefinable set of discretionary break characters.

Both `\path` and LATEX `\verb` expect their arguments to be surrounded by identical delimiters that are not used in the arguments. The `x-*.bst` files take care to use only the command form `\url|...|`, which otherwise in LATEX also accepts a braced argument.

15 Incompatibility with the hyperref package

The LATEX hyperref package provides an easy way to get hypertext links within, and from, a typeset document output in PDF from. Alas, while it knows about the URL markup style `\url{...}`, it fails to handle the delimited style `\url|...|`. However, we need the latter to support plain TEX via the path package.

There appear to be several solutions, in order of decreasing desirability:

- Repair the hyperref package so that it recognizes the delimited URL macro, which it should have done in the first place, because both braced and delimited styles have always been supported by the url package.
- Extend the path package to handle braced arguments, and have the `x-*.bst` files wrap DOIs and URLs in that form. The possibility of special characters that have other meanings in TEX than literal text characters means that such arguments cannot be passed to other macros, but instead must be typeset at their first occurrence.
- Produce companion style files named `p-*.bst` that differ trivially from the `x-*.bst` files, the first producing vertical-bar delimiters where the second have brace delimiters.
- Drop support for plain TEX, and output only braced DOI and URL values.

Resolution of this serious problem is best done after seeking the sound advice of recognized TEX and LATEX experts.

In the meantime, two lines in `xbtxbst.doc` have been temporarily reset to generate `\url{...}` instead of `\url|...|`, so that this document can be typeset with hypertext links.

16 BIBTEX limits

Old versions of BIBTEX had hardcoded sizes of several internal arrays, likely unchanged from its origins in the mid-1980s when most machines had at most a few megabytes of memory. In 1994, this author revised the BIBTEX source code to use dynamic allocation, and reallocation, of those arrays, banishing messages like this one seen when the new `x-*.bst` style files are used with old BIBTEX executables:

```
Sorry---you've exceeded BibTeX's single function space 100
(That was a fatal error)
```

The benefits of the new styles are more important than maintaining compability with an 'ancient' version of BIBTEX, so we simply record that limitation here. Users of modern TEX distributions, such as em-TEX, MiK-TEX, PC-TEX, TEX Live, and others, will not encounter such obstacles.

17 Reconstructing a BIBTEX file

The new bibliography style files described in this document intentionally wrap every output field value in a distinct macro, allowing users to easily alter the formatting of any particular field, and permitting other software to identify and extract selected data from the `.bbl` file. There is sufficient detail to reliably recover all of the fields and values from the input BIBTEX database files that were actually written to the `.bbl` file.

To demonstrate that useful capability, the distribution of the new styles also contains this document and its companions for other styles, and an awk program to reverse BIBTEX's operations. You can use it like this:

```
% nawk -f xbbl-to-bib.awk myfile.bbl > new.bib
```

The awk language is one of the simplest of the many scripting languages that have been developed in the Unix world, and it is available for all common desktop software platforms, including the popular, and GUI rich, but tool poor, Microsoft Windows system. The syntax of awk somewhat resembles that of the C language, and this author has taught it to students several times in one-hour courses.

The awk language is rigorously defined by the IEEE POSIX Standards. Importantly, there are at least *three* completely independent implementations of the language: GNU **gawk**, Michael Brennan's **mawk**, and the original Bell Laboratories **nawk** described in a book [**?**] that is short enough to be read and understood in an evening or two. The latter version replaced an older prototype implementation called **awk**, prefixing it with **n** for *new*. The old language is of historical interest only, and all modern systems support the newer definition in the book.

The recovery of a BIBTEX file from a `.bbl` file is not perfect, because all of the field assignments that BIBTEX ignored in processing the selected bibliography style are missing. Nevertheless, for the purposes of turning reference lists into reusable BIBTEX data, the tool works well. As we noted earlier, the tool could be helpful in journal-production environments to automatically convert author-supplied bibliographic data into any desired house style. Apart from possible lettercase changes, BIBTEX does not alter field values, so little is lost, and importantly, citation labels, and protecting braces in **title** values, are preserved.

As an example of how the program works, here is an original BIBTEX entry

```
@Article{Zucchina:2050:FTA,
  author =       "Asparago Zucchina",
  title =        "Fake title with all fields set",
  journal =      "Bogus Journal",
  volume =       "1",
  number =       "2",
  pages =        "3--4",
  day =          "31",
  month =        dec,
  year =         "2050",
  CODEN =        "ZZZZZ",
  DOI =          "http://dx.doi.org/10.9999/bogusj.1.2.3.4",
  ISSN =         "9999-9998 (print), 9999-9999 (electronic)",
  ISSN-L =       "9999-9998",
  bibdate =      "Thu Apr 06 11:58:55 2017",
```

```
  note =         "This is a note about this sample article.",
  URL =          "http://docs.example.com/zanetti/bogusj.1.2.3.4",
  acknowledgement = ack-nhfb,
}
```

that produces in the x-plain style the `.bbl` entry

```
\bibitem{Zucchina:2050:FTA}
  \ifshowBIBTYPE \showBIBTYPE{article}{Zucchina:2050:FTA} \fi
  \showAUTHORRAW{Asparago Zucchina}\showAUTHOR{Asparago Zucchina}.
\newblock \showTITLE{Fake title with all fields set}.
\newblock {\em \showJOURNAL{Bogus Journal}}, \showVOLUME{1}\penalty 0
  (\showNUMBER{2}):\penalty 0 \showPAGES{3--4}, \showMONTH{December}
  \showDAY{31}, \showYEAR{2050}. \ifshowCODEN {\showCODEN{ZZZZZ}}. \fi
  \ifshowISSN {\showISSN{9999-9998 (print), 9999-9999 (electronic)}}. \fi
  \ifshowISSNL {\showISSNL{9999-9998}}. \fi
\newblock \ifshowURL {\showURL
  \url{http://docs.example.com/zanetti/bogusj.1.2.3.4}}. \fi
  \showNOTE{This is a note about this sample article.}
\newblock \ifshowDOI {\showDOI \href {https://doi.org/10.9999/bogusj.1.2.3.4}
          {10.9999/bogusj.1.2.3.4}}\ifshowDOIPERIOD
  . \fi \fi
```

from which our awk program reconstructs this BIBTEX entry:

```
@Article{Zucchina:2050:FTA,
  author =       "Asparago Zucchina",
  title =        "Fake title with all fields set",
  journal =      "Bogus Journal",
  volume =       "1",
  number =       "2",
  pages =        "3--4",
  day =          "31",
  month =        dec,
  year =         "2050",
  CODEN =        "ZZZZZ",
  DOI =          "http://dx.doi.org/10.9999/bogusj.1.2.3.4",
  ISSN =         "9999-9998 (print), 9999-9999 (electronic)",
  ISSN-L =       "9999-9998",
  bibdate =      "Sat Apr 15 10:19:25 MDT 2017",
  note =         "This is a note about this sample article.",
  URL =          "http://docs.example.com/zanetti/bogusj.1.2.3.4",
}
```

The input and output BIBTEX entries are almost identical, differing only in the **bibdate** and **acknowledgement** fields, neither of which is present in the formatted list. Had the fields been longer, there would likely be more differences, because the output strings are not wrapped across lines. Processing input and output BIBTEX entries with this author's **bibclean** and **biborder** tools would prettyprint them and standardize field order, making them more similar.

18 Anatomy of BIBTEX formatting

Although we have shown examples of BIBTEX input and output, apart from the fragment of the `format.doi` function shown on page 10, we have said little about BIBTEX style-file language programming.

It is now time to delve deeper into the internals of that language. Here is how a BIBTEX `@Article{...}` entry is processed in `x-plain.bst`:

```
FUNCTION {article}
{
  output.bibitem
  format.authors "author" output.check
  new.block
  format.title "title" output.check
  new.block

  crossref missing$
    {
      journal "journal" wrap.required emphasize "journal" output.check
      format.vol.num.pages output
      format.date "year" output.check
    }
    {
      format.article.crossref output.nonnull
      format.pages output
    }
  if$

  write.others
  finish.entry
}
```

In historical styles, the critical output operations are relegated to `output` and `output.*` functions. They do not write their current arguments, but instead, write the most-recent string given to one of the family. Their current arguments remain on the stack, unwritten.

The reason for that peculiar behavior is that BIBTEX wants to be able to supply punctuation between strings, and to do so, it has to track transitions into, inside, and out of, period-ending sentences. Also, if punctuation is omitted at the end of a field value that should have it, such as the **`note`** field, BIBTEX silently supplies it.

That is all clever and useful, but it makes programming BIBTEX style files difficult, because the programmer is often unsure about what is on the pending output stack, and because of the language limitations, is unable to safely inspect stack contents, or the pending strings, or even test for their presence or absence. Worse, if a `write$` primitive function is called to output the current top of stack, when there is a pending string below it, output strings are in the wrong order!

After several unsuccessful tries at properly supporting the several new field names that way in `xbtxbst.doc`, this author gave up and adopted a simpler and cleaner approach.

Each of the new field values is output as a complete sentence, so there is really no need to track sentence state, nor to have confusing hidden pending output strings. Thus, the historical style

```
format.XXX output
```

is replaced with

```
write.XXX
```

The only precaution needed is to call the function `write.string.with.period` to transition from delayed output to immediate output, and that needs to be done in only one place.

All of the BIBTEX document types have the same extended field ordering in the `.bbl` file, produced with this function:

```
FUNCTION {write.others}
{
  write.string.with.period
  write.coden
  write.isbn
  write.issn
  write.issnl
  write.lccn
  write.price
  write.url
  write.note
  write.doi
}
```

Here is a typical function called there:

```
FUNCTION {write.issn}
{
 issn empty.or.unknown
    { "" }
    { "\ifshowISSN {" issn "issn" wrap * "}. \fi " * }
  if$
  write.string
}
```

The empty string output in the *then* part of the conditional could easily be replaced by a wrapper `\showMISSINGISSN` to provide the diagnostic capability that we discuss in the note-to-the-author on **page 16**.

The particular syntax `value "field" wrap` is a function call that returns the string `"\showFIELD{value}"` on the stack, and receives extensive use in the `x-*.bst` files.

Data are finally written to the `.bbl` file with this function:

```
FUNCTION {write.string}
{
  duplicate$ empty$
    'pop$
    'write$
  if$
}
```

It discards an empty argument string, and returns immediately. Otherwise, it pops and writes out that string.

19 Remarks on the reference list

This section is followed by four sections with the same reference list presented in different formats: annotated, wide one-column extended, narrow two-column, and wide one-column abbreviated. In each case, the *same* input `.bbl` file is read, but changes in the wrapper and conditional macros allow variations in the displayed material.

Most of the entries in the reference list have been chosen from existing ones in our bibliography archives. The selections for this document represent all of the document types supported by all existing BIBTEX styles, as well as the additional `@Periodical{...}` type recognized by the `is-*.bst` and `x-*.bst` styles.

Many entries include CODEN, ISBN, ISBN-13, ISSN, ISSN-L, and LCCN data, and some have **`bookpages`**, **`day`**, and **`price`** fields. The **`day`** (and **`month`**) fields are essential in references to newspaper articles, because newspapers rarely carry volume and issue data that could be used to identify document locations more precisely.

The **`day`** field is also required for weekly journals, of which *Nature*, *Science*, *The BMJ* (formerly, *The British Medical Journal*), *The Journal of the American Chemical Society (JACS)*, and *The New England Journal of Medicine (NEJM)* are among the most prestigious. About 15% of the `@Article{...}` entries in our archives have **`day`** values.

There are also examples of a single article from one conference proceedings [**?**], and multiple articles from another conference proceedings [**?**, **?**, **?**]. In the former, the proceedings data are included with the article data, whereas in the latter, each list entry has an automatically generated cross reference to a separate entry for the proceedings [**?**].

One entry is an example of a paper [**?**] that was later corrected by a short erratum. The entry contains a cross reference to another with the erratum, so the writer only needs to cite the main paper, but can be assured that any related comments, corrigenda, debates, discussions, errata, notes, rebuttals, remarks, and responses are automatically incorporated in the reference list.

The possibility of chains of cross references within entries means that the traditional command sequence

```
latex myfile.ltx ; bibtex myfile ; latex myfile.ltx
```

may need additional pairs of **`bibtex`** and **`latex`** commands to resolve them completely. Both programs make only one pass through their input files, so a newly cited document that BIBTEX mentions in the text of its output `.bbl` file is not noted until LATEX next typesets the bibliography, producing a new citation request in the `.aux` file that BIBTEX reads when it is next run.

Collectively, the chosen entries provide a comprehensive set of examples of the formatting of BIBTEX entries that are likely to contain more metadata than most bibliographers in the past have bothered to record. However, remember that reference lists are provided in most academic publications to guide the reader to prior work, and to learn more about the document's subject area. Thus, writers should strive to help their readers by supplying as much information about each reference as can reasonably be found, and CODEN, DOI, ISBN, ISBN-13, ISSN, ISSN-L, LCCN, and URL data are particularly useful.

The historical practice of providing highly abbreviated reference list items reflects concern for the large effort needed by authors to manually prepare reference lists, for human typesetters to then

reformat them according to the journal style, and for publishers to reduce costs, rather than to make life easier for the more numerous humans who later read and use the document.

With the help of BIBTEX, LATEX, and TEX, much of the past tedium of reference-list preparation and formatting is eliminated, and the resulting lists can be much more useful to readers. In electronic documents, DOIs and URLs can be wrapped with hypertext links, such as provided by the LATEX `hyperref` package.

Importantly, the work of generating a BIBTEX entry for a given document really only needs to be done well *once*; the result can then be shared electronically with everyone who needs to cite the same document. The cited bibliography archives are a significant step in that direction, because they are actively curated and maintained, are public domain, and are easily available on the Internet.

Two-column typesetting

One-column typesetting