The *refdbms* bibliography database user guide and reference manual

John Wilkes

Concurrent Computing Department
Hewlett-Packard Laboratories

HPL–CSP–91–11 20 May 1991

This is the user guide and reference manual for *refdbms*, a scheme for maintaining a database of bibliographic references and retrieving them for citation. This paper documents the *refdbms* facilities, commands and data formats.

Contents

1 Introduction

The *refdbms* reference system helps you add the bibliographic citations that traditionally clutter the ends of papers and other scholarly works. It also helps you find papers to cite, or even read.

The primary purpose of this document is to introduce *refdbms* and to act as a user guide for it. A secondary purpose is to describe the internal workings for people who wish to delve more deeply.

To get started right away, see the Quick Reference Guide on page 53. To find out more about what *refdbms* can do, continue reading this section, and the ones that describe the database structure and the public commands. To build and maintain your own database, or make more than cosmetic changes to the *refdbms* defaults, read the appendices. If you want to delve into the innermost implementation details, you can read the *refdbms* source code.

Required caveat

This announcement does not constitute any warranty, express or implied, and all that jazz. *Refdbms* continues to evolve; suggestions for improvements are very welcome, as are notifications of any defects you find.

Purpose

Refdbms provides services to authors of papers that cite other people's work. It includes commands that find, select, and format references for use. You can use *refdbms* as a convenient indexing and retrieval system for papers, and even build and maintain your own private reference database. Your site may also have a database of references ready for citation (see the Local Guide starting on page 43 for details).

To illustrate the use of *refdbms*, suppose you were writing a paper about the growth of marsupial populations in the barren desert areas of central Australia. You vaguely remembered reading a paper a year or so ago that had direct bearing on your thinking, but can't quite place your hands on it now.

A quick refsearch *marsupials* gives you a list of the related references. But this turns out to be too long, and you can't remember which of the set was the one you cared about. So you narrow down the search by remembering that both *marsupial* and *desert* occurred in the reference: refsearch *marsupials desert* restricts the search to those references that contained both, and now three references result. The output from refsearch is a list of the *tags* (internal names that uniquely identify references); you convert these into into full references by piping them into refget. (Alternatively, you can use the reflook command to do both operations together.) The paper Stagg87a by Amos Alonzo Stagg, Jr is the one you want, and so you embed the reference to it in your latex document: `\cite{Stagg87a}`.

When you've finished writing your document (or when you want to print a draft to look at), you run refbibtex *marsup.tex* to build the marsup.bib file, ready for latex and bibtex. Your references will come out in the paper's bibliography, sorted and nicely formatted.

Refdbms is not intended as a substitute for libraries, which offer very extensive cataloging and searching facilities that it would be foolish to try and duplicate. Rather, it acts as a useful supplement to such resources for common cases, frequent use, and the mechanical process of producing papers with citations in them.

Alternatives to *refdbms*

Why use *refdbms* instead of one of the existing "standard" bibliography maintenance systems? Here are a couple of factors that might influence your choice:

- *Speed*: refsearch can find 300 references in 2.3 seconds, and refget can give you the text of those references (about 0.25MBytes) in 6.9 seconds—1.4 seconds if the data are already in the file buffer cache.[1]
- *Simplicity*: despite the perhaps daunting thickness of this guide, there are only a couple of commands that need to be mastered to use *refdbms*.
- *Completeness*: citations made with *refdbms* include everything needed to track down the item being referenced, unlike many bibtex styles.
- *Expandability*: any combination of private and centralized databases is possible, so your personal collection of early twentieth century communist propaganda doesn't have to be visible to your colleagues.
- *Customizable*: you can change the way *refdbms* displays references using all the facilities of bibtex, and a few more besides.
- *Correctness*: entries in the communal reference database have been carefully, if not lovingly, checked for correctness: one person's efforts frequently cross-check another's. In the unlikely event that an error is found, it can be corrected easily.

Before you become too enamoured of these facilities, you should be aware of the competition for *refdbms*. There are basically four main packages: refer [Lesk78], Scribe [Reid81], bibtex [Lamport85,Patashnik88] and Tib [Alexander87]. Each has some advantages and disadvantages for producing bibliographies and for managing databases of reference entries, summarized here.

1. refer, which supports the troff family of formatters [Kernighan78,Kernighan81], was the original inspiration for both *refdbms* and Tib. The packages use similar, although not identical, formats for their reference files. The main functional differences are that (1) the type of each reference is made explicit in *refdbms*, but left implicit in refer and Tib; and (2) refer reference entries don't have a unique tag by which they can be identified: instead, "sufficiently many" keywords have to be given to identify a single reference. Clearly, both decisions can result in ambiguities.
2. Tib, which supports TEX, was derived from an earlier similar formatter called bib which worked with the troff family. The reference file format is almost identical to refer, except that many field types have been added to support documents that have

[1]Timings from an HP 9000/845 system.

been translated from other languages, and the system comes with control files that describe many, many different styles of bibliography and citations.

3. Scribe was developed as a research vehicle to demonstrate the practicality of separating document content from form. A part of that demonstration was a nicely designed bibliography package. The Scribe program is now commercially available, at a not inconsiderable cost. However, no support is provided for searching Scribe bibliography files: their purpose is to supply data for pagination, not to act as a database. And although the format is elegant, it is not easy to use text-processing tools like grep and awk with it.
4. bibtex is an add-on package and program for the latex formatting system. It uses (essentially) the Scribe format for its references. Like Scribe, it comes with no support for searching for references by keyword for exploratory purposes, and the bibtex program is severely limited in the size of the files that it can handle successfully. bibtex is rather slow, and designed only for use with latex. Together, these make bibtex unsuitable for managing a large database.

   The bibtex style files, which describe how references and citations are to be formatted, are complicated programs in a special interpreted language (compared to the much shorter declarative form used by Tib). In addition, the standard style files take many liberties. For example:

   - they change the case of words in titles;
   - if the proceedings of a conference is published as a journal issue, the journal name, volume, and issue number are ignored;
   - ignored fields generate no error messages to warn you that information is being discarded.
5. Making a minor change to a bibtex style requires generation of a complete new style file: this can easily lead to a profusion of different variants corresponding to minor differences in personal taste.

Refdbms was inspired by refer and Scribe, and makes use of bibtex to do some of its formatting. It reaps the advantages of fast indexes that make refer so useful, while avoiding the ambiguities of that system, and extends the reference format to store more information. Although *refdbms* does not yet support the wealth of formatting options available through Tib, it achieves a respectable subset via use of bibtex, but avoids the latter's indiscretions by providing *refdbms*-specific style files that take full advantage of the available data. It also provides some support for FrameMaker documents.

Refdbms has resulted from a long period of growth, tinkering, rationalisation, and more tinkering. It doesn't yet provide all the facilities that might be wished, but, nonetheless, it does seem to have reached the point where it is usable. I hope you find it useful, too.

Acknowledgments

David Jacobson contributed the reference-mode package for GNUemacs. Richard Golding has been instrumental in developing the multiple-database management code. Diana

Roberts provided some fine feedback on the documentation. They and many other colleagues at HP Laboratories have contributed bugs reports and fixes, ideas for improvements, and many reference entries.

Highlights of this edition

Here is a list of the major things that have changed with this edition of the user guide.

- New commands: refsubmit, reflook, refmaker, refmatch
- Modified commands:
  1. refbibtex sends it output to stdout, rather than trying to deduce the filename to use;
  2. all the keys to refsearch need to hit before a reference matches: before, only one need match.
  3. the expandrefs *-F expandfile* option became *-E expandfile.*
- Authors' and editors' full forenames are now kept by default, rather than being converted to initials. The *-I* option forces conversion to initials, if desired.
- Improvements to multi-user use of the commands (better locking to prevent conflicting updates, new refsubmit command).
- Error messages are now uniformly written to stderr, rather than a mixture of stderr and stdout.
- Database changes:
  1. the new field type %I [ISBN/ISSN] is used to record ISBN and ISSN data;
  2. the %X [extract type] field has been eliminated;
  3. the format of titles has changed to eliminate the last vestiges of Upper and lower case except for Proper Names;
  4. minor formatting-rules have changed in the %C [conference name] field;
  5. the %R [report number] should no longer record that a PhD thesis has been published as a technical report.
- Large- and small-formats for databases have been defined, and a template Makefile for managing them supplied.
- There is an initial stab at FrameMaker support.
- (Hopefully) better documentation, including a separate Local Guide that isolates all the site-specific bits.

Things in the works

Refdbms is a living piece of software. Your suggestions, advice and contributions are requested to help it improve.

By way of a preview, here are some of the features that are being considered for inclusion in future versions. Doubtless their form will change before they arrive; equally, several features may never see the light of day. Such is the nature of experimental work.

- Better support for multiple databases, including an explicit *database:tag* format.
- Support for groups of databases that span multiple file systems (e.g. replicated databases).
- Explicit inter-reference linkages through a dedicated field.
- Better FrameMaker support.
- Filters to import data held in refer, bib, Tib, and bibtex format.
- ACM Computing Reviews category support.
- Escapes to allow you to enter all fields from newref.
- Better documentation (suggestions welcome!)

2 Database structure

This section of the document provides an overview of the way a reference database is put together. A detailed description may be found in chapter 5.

Terminology used in *refdbms*

There are a number of common items that crop up in describing the bibliography and citation process. Here is how they are referred to in the *refdbms* system:

- *Reference*: an entry in the database describing a paper, book, etc.
- *Citation*: a pointer to another work from the body of a paper. This pointer is typically indirect, since it refers to an entry in a bibliography at the end of the paper.
- *Bibliography*: a collection of descriptions of other works, embedded in a paper. A bibliography is constructed by extracting a set of references and massaging their format to conform to that needed by the document processor being used for the paper. (A common such format is that used in bibtex .bib files.)
- *Tag*: the internal, unique name of a reference in the database (e.g. "wilkes83a"). No two references in a *refdbms* database may have the same tag. Tags are formed from the last name of the primary author, plus two digits for the year of publication, followed (only if needed) by a letter to disambiguate multiple publications from the same year.
- *Keyword*: a word associated with a reference (e.g. "marsupials"), by which it may be retrieved. A reference will have many keys: the words of the title and the authors' names are included automatically, and you can supply others when the reference is entered into the system. "Noise" words such as "the" and "or" are discarded, as are single letters and non-alphanumeric characters.
- *Database path*: a list of directories containing reference databases. The *refdbms* databases on a database path are searched in turn, so this allows you to treat two or more databases as if they were one larger one.

References

A reference in a *refdbms* database consists of a contiguous set of lines. Each line commences with a percent sign (%) and a letter. All the lines tagged with the same letter are said to be of the same *type*, and consecutive lines of the same type are called a *field*. Every reference in a file—including the last one—is ended by a blank line.

The first field (line) of a reference must always identify the type of the reference; the second its tag. These are the %z [reftype] and %K [tag] fields respectively. The reference type indicates the nature of the item being described, such as a book, a journal article, or a technical report.

The other fields in a reference are a function of its type (e.g. articles published in journals usually have a volume number and a page range, whereas a technical report will usually have neither), and the zealousness with which the person who entered the reference into the database did so. A complete list of the individual field types can be found on page 22.

Files

A reference database consists of one or more files containing the references themselves, together with two auxiliary files that hold a pair of inverted indices over the references. The two auxiliary files are always called:

Tags	maps a tag to the file and byte offset of its reference
Keys	lists the tags of references containing each keyword

Figure 1 shows the relationship between keys, tags and references.

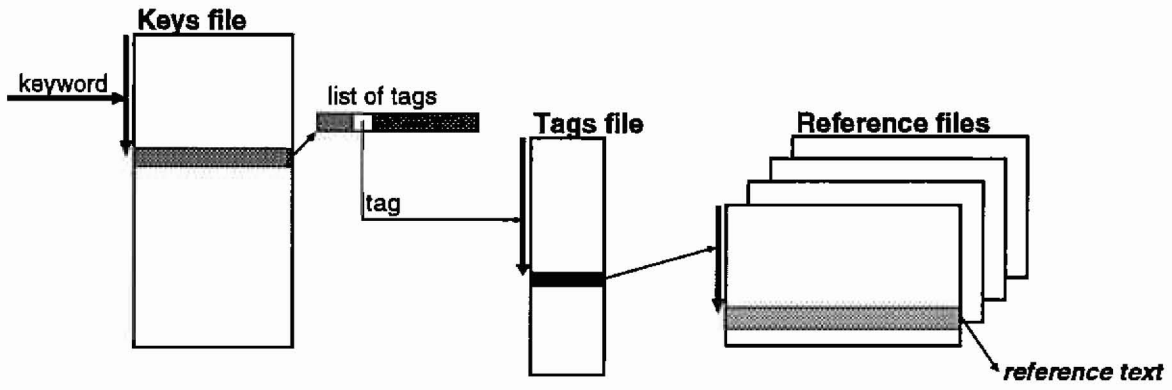

Figure 1: How keywords map to tags, which map to references. Keys are looked up in the Keys file to produce a list of tags. These are in turn looked up in the Tags file to find where the reference is in the database.

All files in a *refdbms* database contain only straightforward printable ASCII data, with white space separating items within a line. More information on the detailed file structures can be found in chapter 5.

References can be kept in any convenient set of files: the files don't even all have to be in the same directory (although this is usually what happens). There always is just one Tags and one Keys file per database.

There is a Makefile that you can use as a template for maintaining a private database. The local guide (p. 43) indicates where this template file can be found. The Makefile will automatically rebuild the Tags and Keys files if needed. It expects the references to be in a file called References.

2.1 Abbreviations

The idea behind *abbreviations* is that a short, concise, standardized form of a longer name can be expanded automatically, rather than have to be entered by hand each time it is encountered. This has several benefits:

- less typing
- more accurate references
- different expansions for different uses: you might choose the fully-expanded form "IEEE Transactions on Software Engineering" for a tutorial article, but a condensed version "IEEE Trans. Softw. Eng" for a paper in a journal short on space; both would come from the initial abbreviation "IEEESE."
- consistency in the expanded form, both in terms of length and correctness

Refdbms provides support for such abbreviations, and their use is actively encouraged. A *refdbms* abbreviation is a contiguous set of letters *terminated by a period*. The case of an abbreviation is significant. The expanded forms that abbreviations map into are described by expansion control file; each such control file defines one set of expansions.

Some abbreviations are specific to just a few types of fields: for example, conference name abbreviations are only expanded if they occur in the conference name field. Such abbreviations are described with those field types in chapter 5. (The newref command (p. 14) will provide an up to date list of them as part of its help information.) No expansions are done on AEkos fields: authors' and editors' names, keywords, private notes, and the submitter field. The standard abbreviations applicable to all the other fields are as follows:

Abbreviation	*Expansion*	*Abbreviation*	*Expansion*
AI.	Artificial Intelligence	Applic.	Applications
Arch.	Architecture	Archit.	Architectural
Assoc.	Association	Co.	Company
Comm.	Communication	Comp.	Computer
Compt.	Computing	Conf.	Conference
Corp.	Corporation	Dept.	Department
Depts.	Departments	Dev.	Development
Distrib.	Distributed	Elec.	Electrical
Eng.	Engineering	Eur.	European
Grp.	Group	Inc.	Incorporated
Inf.	Information	Inst.	Institute
Intl.	International	J.	Journal
L.	Letters	Lab.	Laboratory
Labs.	Laboratories	Lang.	Languages
Mach.	Machinery	Natl.	National
O-O.	Object-Oriented	Obj.	Object
Op.	Operating	Orien.	Oriented
Princ.	Principles	Proc.	Proceedings of
Prog.	Programming	Res.	Research
Sci.	Science	Soc.	Society
Softw.	Software	Spec.	Specification
Symp.	Symposium	Sys.	Systems
Syst.	Systems	Tech.	Technical
Theor.	Theoretical	Trans.	Transactions
Univ.	University		

There are currently two standard expansion control files. Both live in $REFDIR/Source:

- Expand-long does a complete expansion, removing all abbreviated forms; it is useful when clarity is more important than space (e.g. in technical reports).
- Expand-short does a partial expansion; it is for use where space for the bibliography is at a premium (e.g. in papers published in a journal).

Eventually there will be other abbreviation styles, such as Expand-ACMComputingSurveys, to accommodate the whims and dictates of particular target publications.

You can override the standard abbreviations by supplying your own expansion control files to be used in place of the standard ones. The expansion control file format is described in chapter 2.1. You can also run the expandrefs command directly (see p. 12).

2.2 Style files for bibtex

Three bibtex styles are provided for use with *refdbms*:

- refalpha – alphabetic keys sorted on author, e.g. [Fruitfarm89, Gerbil77a]
- refplain – numeric keys sorted on author, e.g. [2, 13]
- refunsrt – numeric keys in citation order, e.g. [23, 24]

These are direct analogues of the standard bibtex styles, except that they have been rewritten to work with references extracted from *refdbms*. The recommended default is refalpha.

Regrettably, *refdbms* bibliography files are not directly compatible with the standard bibtex style files, since the latter ignore so much of the information stored in a *refdbms* entry.

2.3 Multiple databases and the database path

Refdbms provides support for multiple databases: that is, you can create, search, and retrieve references from multiple separate sets of files. Each database has its own Keys and Tags files (which also means that there can be no more than one database per directory).

All relevant *refdbms* commands take a *-P dbmspath* option to specify the list of databases. Alternatively, if the environment variable REFPATH is set, it will taken as the database path. Finally, if no *-P dbmspath* option is given and REFPATH isn't set, then the environment variable REFDIR is used as a single-element database path. The phrase "database path" in this document means the list of databases obtained by one of these means.

Both the *-P path* option and the REFPATH environment variable use the same syntax to provide the list of databases: a colon-separated list of directory names.

3 Public commands

This section of the paper describes the commands that normal use of *refdbms* will require. These are the so-called *public commands*:

- reflook, refsearch, and refget look references up and retrieve them;
- newref and refsubmit add new references;
- refbibtex and refmaker build bibliographies for bibtex and FrameMaker respectively;
- expandrefs converts abbreviations into their longer forms;
- refstrip summarizes references.

All the other *refdbms* commands are documented in chapter 6.

3.1 Finding and retrieving references

refsearch *[-P dbmspath] [-a] key . . .*

Looks for references that match one or more of the given keys, and writes a list of their tags to stdout.

The case of keys *is* important: keys that are all in lower case are passed through a "word stem" algorithm before being looked up, and the result will match any leading substring stored in the Keys file. (For example, mach will successfully match mach, Mach, machine, and so on.) Both behaviours can be prevented by putting one or more of the letters in the key into upper case. (This is particularly useful for authors' names.)

A reference is considered to *match* the search only if *all* of the given keys are matched.[2] That is, the effect is to "and" together the searches for each key. You can get the effect of an "or" by using the *-a* (for *any*) flag. In this case, a reference will match if any of the given keys are found.

The database path can be supplied as a list of directories in the *-P dbmspath* option (see p. 9), or in the environment variable REFPATH. If neither are given, the environment variable REFDIR is used as a single-element database path.

The normal exit code from refsearch is 0 (zero); 1 is returned if some required key couldn't be matched or if no matches at all could be found for a search; 2 if some more serious error occurred.

Bugs: No attempt is made to suppress duplicate tags from multiple databases. (Worse still, at present there is no way to identify which tag came from which database. This will be fixed when the database path code is upgraded.) The -a option is a hack: full tree search expressions should be supported in the style of find.

[2]This is a change from previous editions of this guide.

refget *[-k] [-P dbmspath] [-f inputfile] [tags . . .]*

Retrieves references from the database given their tags. Output is sent to stdout.

One or more tags can be supplied on the command line. If none are supplied, the list is read from the file *inputfile*, if it is present, or from stdin otherwise. In both cases, the input should have one tag on each line. Using a minus sign (-) for *inputfile* will cause the command to take its list of tags from stdin.

The case of tags is irrelevant and white space is silently discarded. Tags can include restricted regular expressions in the style of *regcmp(3)*. In particular, the wildcards period (.) and period-asterisk (.*) are supported, meaning the same as the shell's '?' and '*' respectively.

> The usual idiom is to use patterns like tag83.* to match tag83, tag83a, tag83b, and so on. Don't forget to escape the wildcards with quotes or backslashes (as here) to stop the shell interpreting them.

The database path can be supplied as a list of directories in the *-Pdbmspath* option (see p. 9), or in the environment variable REFPATH. If neither are given, the environment variable REFDIR is used as a single-element database path. By default only the first reference with a given tag is returned, ignoring any others in subsequent databases on the database path. The *-k* option retrieves all references with the given tag from the list of databases on the database path.

The normal exit code from refget is 0 (zero); 1 is returned if a tag couldn't be looked up; 2 if some more serious error occurred.

reflook *[-aks] [-P dbmspath] [keys . . .]*

Does a combined refsearch and refget. The *-s* option pipes this through refstrip. Output is piped to stdout through more (or the command specified in the environment variable PAGER, if it is defined).

refstrip *[-K] [-k fields | -d fields] [files . . .]*

This command reformats the references it is given for ease of reading; it is typically used as a filter to peruse the output from refget. Its default is to output just the tag and title information in the following format:

```
[Gnasher88a]
  Fruit farming on the mountainous slopes of Southern Mongolia
[Gnarled67]
  The economics of wombat farming in tropical rain forests
```

The following options are available to control its processing further:

- *-K* suppresses the special handling of %K [tag] lines and the subsequent indentation
- *-k* specifies which fields should be kept (syntax is a list of letters, such as AEO, or a range such as *A-Z*)

-d specifies which fields should be deleted (the remainder will be kept; syntax is the same as for *-k*).

Only the rightmost *-k* or *-d* option takes effect.

expandrefs *[-E expandfile] [files . . .]*

This expands abbreviations in references into longer forms (see p. 7 for more details on abbreviations, p. 36 for information on the format of expansion control files). This command is run automatically by ref2bibtex (page 12). Input is taken from the list of files given, or stdin; output is to stdout.

The optional *expandfile* parameter directs the command to use a specific expansion template; the default is to expand names in full. (The standard values for *expandfile* are long and short.) If *expandfile* is itself a readable file, then it is used, otherwise it is taken to be a suffix to the string $REFDIR/Source/Expand-.

Bugs: expandrefs is rather slow. It should be possible to provide database-specific expansion files, rather than be restricted to a single global one.

3.2 Text processor support

refbibtex *[-E expandfile] [-P dbmspath] [-la] file1.tex file2.tex . . .*

Generates a bibtex .bib file from one or more latex input files containing \cite and \nocite commands.[3]

The refbibtex command looks at the given .tex file(s) and at the associated .aux files. It scans these for citations embedded in the latex file(s) directly, and in references pulled in by any of the cited references. This means that it can be used before latex has been run, although it should be re-run once the .aux file has been created to make sure that all the cross-references are correct.

The *-P dbmspath* option is passed on to refget; the other arguments are given to an internal ref2bibtex. The output is written to stdout.[4]

Bugs: a defect in the scanning algorithm means that even citations that have been commented out will be included. This will doubtless be fixed at some stage.

ref2bibtex *[-E expandfile] [-laA] [files . . .]*

Converts references retrieved by refget into the format used by bibtex .bib files. The input can be from the files given on the command line, or on the standard input. Output is to stdout.

By default, abbreviations are expanded to their longest form, first names are shortened to their initials; and %x [extract], %o [private note], and %k [keyword] matter are suppressed

[3]Important safety tip: don't put spaces inside the arguments to either of these LaTeX commands.

[4]This is a change from previous editions of this guide.

because they are rarely useful for bibtex citations. These defaults can be overridden as follows:

-E behaves just like the *-E expandfile* option to the expandrefs command (p. 12)

-I will cause forenames to be abbreviated to initials

-a retains the abstract, private note and keyword matter

Notes:

1. The generated bibtex files are designed to be used with one of the refalpha, refplain or refunsrt style files for best results.
2. Since bibtex limits the amount of text it is willing to handle, you may find that the *-a* option generates too large a .bib file. If this happens, you can use the *-A* option instead of *-a*. Instead of including the abstract and private note matter into the .bib file, it writes each abstract and each private note to a separate file, and puts \input commands in the generated bibtex ABSTRACT and PRIVNOTE entries in the .bib file. The names of the files are derived from the tag for the reference.

Bugs: since the bibtex format is not perfectly compatible with that of the reference database, minor editing may still be required on the ref2bibtex output.

refmaker *[-E expandfile] [-P dbmspath] [-I] files . . .*

Generates a FrameMaker bibliography on stdout. The output is in .mif file format. Input is one or more FrameMaker files containing tags enclosed in [square brackets].

The *-P dbmspath* option is passed on to refget; the *-E expandfile* to expandrefs. The *-I* causes forenames to be shortened to initials.

Bugs: The algorithm for finding tags considers *anything* enclosed in square brackets a potential tag, which means that you'll get lots of "unable to find a match to . . . " messages if you have regular text bracketed. Also, there is no way to suppress the trailing letter of the tag if you only cite (for example) Grimblethorpe91g in your document, but no others by the same author.

3.3 Adding new references

The "standard" way to add new references to a *refdbms* database is to use the newref command. This uses a prompt/response format to query you for information about one or more new references. Some of the prompts you get depend on the kind of reference; some are common to all reference types. Along the way newref performs several checks on the new reference, such as whether it already exists, and whether its syntax is roughly correct.

For GNUemacs users, there is a package available for constructing references from inside the editor. It is documented in section 3.5 on page 18. Nonetheless, you will still probably find it easiest to start out with the newref command.

If you are maintaining a small private database, you can simply keep all the references in a single file, and add to it with GNUemacs or the editor of your choice. Using the filename

References will ease use of the standard tools. A shared database is best updated by use of the newref, refsubmit and mergenewrefs commands.

To make reference databases as useful as possible, it is important to maintain a high quality for the entries in them. The standard to strive for is *extremely* high: zero defects in content, typography, spelling, page numbers, etcetera. The goal is that citations from a *refdbms* database can be assumed correct and not have to be checked when they are used. As Donald Knuth observed:

> ... people have a great tendency to copy citation information blindly into their own papers, and so errors propagate unchecked. When Elwyn Berlekamp wrote his book on coding theory, he found that nearly half the information in bibliographies of papers was wrong. [Knuth88, p. 30]

Addressing this problem in her *Handbook for scholars*, Mary-Claire van Leunen states:

> To write a reference, you must have the work you are referring to in front of you. Do not rely on your memory. Do not rely on your memory. Just in case the idea ever occurred to you, do not rely on your memory.
>
> ... If you must not rely on your own memory, even less should you rely on someone else's. If your only access to a reference is through a secondary source, then you must refer to the secondary source as well as the primary one. [vanLeunen78, pp 139,142]

The information stored in a reference should be as complete as possible. By way of a small motivational example, the following passage concerns a paper being edited by ACM after acceptance for publication:

> The publishers also insisted on more details in [Knuth's] bibliography. They wanted to know, for example, exactly where and when a conference had taken place. Someone in the class pointed out that Mary-Claire van Leunen recommends omitting the location of conferences. Don replied that libraries often nowadays index conferences by city for those poor souls who can remember nothing else about them; so such information was useful. [Knuth88, p 30]

Even the most careful individuals make mistakes; fixing them is the subject of the section that starts on page 18.

newref *[-o output]*

Add new references to a *refdbms* database. The default output is the file Newrefs in the database directory $REFDIR, but this can be overridden by specifying an explicit *-o output* option, which can be the name of a file, or of a database directory in which there is a Newrefs file.

The newref command prompts for all its input; the reply question-mark (?) to a prompt will get help, in the form of a more wordy description of what is wanted—please ask for such help frequently at first, until you understand *all* of the subtleties of the data formatting conventions.

Use latex forms for em-dashes, en-dashes, accents, etc, and *please* follow the suggestions given by '?' as to how to format entries: a little work early on saves enormous hassle later!

Whenever you can, take advantage of the abbreviations provided for common journals, institutions, and the like. As well as saving you typing, this will allow different amounts of expansion to be provided in different bibliography styles. Abbreviations are partly field-specific (e.g. the names of journals), and partly common to all fields (e.g. various abbreviations from common words such as "Department", "International", "Systems").

When newref starts up, and each time it is ready to describe a new reference, it presents the menu shown in figure 2.

```
Type of reference: one of

      1  Article          in journal
      2  InProceedings    an article from a conference
      3  TechReport       technical report, but not a manual
      4  Book             all of it
      5  InBook           a chapter, or a range of pages
      6  Manual           about a product, program, etc
      7  PhD thesis       (masters' theses are TechReports)
      8  Proceedings      whole of conference proceedings
      9  UnPublished      not formally published
     10  Miscellaneous    unclassifiable

  AGAIN  Another entry from the same issue or book
   EDIT  Edit the output file
  blank  Exit

Please select a type of reference: _
```

Figure 2: The **newref** main command menu.

The following paragraphs describe how to map the kind of reference you have in hand to one of the types that *refdbms* expects.[5]

Most items fall naturally into the first three categories (articles in journals, papers in conference proceedings, and technical reports); a few fit the next couple (books, manuals and PhD theses); and a very small minority need more careful thought. The best rule is to start at the top of the list, working your way further down only if the document doesn't fit into one of the more common categories.

Each type of reference is likely to have a different set of descriptive information. The list of likely items is provided for each reference type. Some items are required; some are expected (i.e. please try to determine them); some are optional. By all means add more information if it is available to you.

All references are required to contain %z [reftype], %K [tag], and %s [submitter] fields. It is always beneficial for them to contain %L [location], %k [keyword], %x [extract], and %o [private note] fields.

[5]Much of this description was inspired by the one in the Scribe users' guide [Reid80, pp. 97–102].

The reference types are as follows:

- **Article**: a individual paper published in a journal or magazine, unless the entire issue of the journal is a conference proceedings (in which case use InProceedings).

  Required fields: %T [title], %D [date].
  Expected fields: %A [author], %J [journal], %V [volume], %N [number], %P [pages], %k [keyword].

- **InProceedings**: a paper published in a collection that is the proceedings of a conference. A single paper in a journal summarizing the entire conference (e.g. a report on a workshop) is an Article.

  Required fields: %T [title], %D [date].
  Expected fields: %A [author], %P [pages], %k [keyword], %C [conference name], %c [conference location]. If the conference proceedings are published as an issue of a journal: %J [journal], %V [volume], %N [number]. Otherwise: %p [publisher].
  Desired fields: %E [editor].

- **TechReport**: almost any document published by a university or company for internal use or wider dissemination, *unless* it is a full-fledged Book, a PhD thesis (even if it is published as a TechReport), or a Manual.

  Required fields: %T [title], %D [date].
  Expected fields: %A [author], %p [publisher], %k [keyword].

- **Book**: a work emitted by a (commercial) publishing house.

  Required fields: %T [title], %D [date], %p [publisher].
  Expected fields: %A [author], %k [keyword].
  Desired fields: %S [series].

- **InBook**: a chapter or section within a Book. (Don't use this for a paper in the proceedings of a conference: use InProceedings instead.)

  Required fields: %B [book title], %T [title], %D [date], %p [publisher].
  Expected fields: %A [author], %S [series], %P [pages], %k [keyword].

- **Manual**: instructions, or technical documentation explaining how to use something. If there were no Manual category, a manual would get classified as a TechReport or a Book.

  Required fields: %T [title], %D [date], %p [publisher].
  Expected fields: %A [author], %k [keyword].

- **PhDthesis**: the text submitted to a University by a doctoral degree candidate. This is frequently published as a TechReport, in which case say so in the %R [report number] field:

  ```
  %R PhD thesis; published as Technical Report CMU--CS--83--124
  ```

  A master's thesis is not considered worthy of its own category: treat it just like a TechReport.

  Required fields: %T [title], %D [date], %A [author], %p [publisher]—the university or school where it was submitted.
  Expected fields: %R [report number], %k [keyword].

- **Proceedings**: an entire volume or journal issue dedicated to the record of a conference; used only when referring to the issue as a whole. Usually the publisher is the only named entity, although some proceedings have editors.

Required fields: %T [title], %D [date].

Expected fields: %E [editor], %p [publisher], %k [keyword], %C [conference name], %c [conference location]. If the conference proceedings are published as an issue of a journal: %J [journal], %V [volume], %N [number]. Otherwise: %p [publisher].

- **UnPublished**: these are documents that have only been made available to a select audience, and have not seen wide circulation. Examples include: work in progress; slide presentations not available as technical reports; personal communications. A %O [public note] field must be supplied to describe the item.

  Required fields: %T [title], %D [date], %O [public note].

  Expected fields: %A [author], %k [keyword].

- **Miscellaneous**: Pretty much everything else – use only as a last resort. Examples include: Usenet articles, patents, publicity brochures, items published in an unusual form. A %O [public note] field must be supplied to describe the item.

  Required fields: %T [title], %D [date], %O [public note].

  Expected fields: %A [author], %k [keyword].

Besides the different types of references, three other command options are available at the main prompt from newref:

- **AGAIN**: makes it easy to add another reference from the same journal or conference proceedings as the last. Type the first reference normally, and then use AGAIN for each of the remaining ones: it will ask you only about those fields that are different.

  Be particularly careful to get the first one right. Any errors you make will be faithfully propagated to subsequent references generated with AGAIN. (*Warning:* because of the way that the option works, it will ignore corrections you make with the EDIT option.)

- **EDIT**: lets you edit the references that newref has collected so far in this session. The value of the EDITOR environment variable is used to start up the editor.

- **(RETURN)** or **(blank)**: submit the references to the database and exit newref.

If you want to enter a field for which newref doesn't prompt, such as %a [author note], the easiest way is to enter the bulk of the reference in the normal way, and then use the **EDIT** option to add the field carefully(!) by hand.

Just before it submits the new references to the database, newref runs a set of simple checks over its input file using checknewrefs (p. 39). If any problems are found, it will complain, and return to the main loop. You can then correct the problem (using the **EDIT** option) and try again.

If you want to experiment without making any lasting changes, try the newref command using a dummy output file.

Should newref abort catastrophically for some reason (e.g. you accidentally kill it), all is not lost. It puts its partial output in a file with a name of the form /tmp/newref9999.ref, where the 9999 is the process ID of the newref command. You can recover this file by hand, clean it up, and then use refsubmit to append it to the Newrefs file.

refsubmit *[-q] [-o output] [files . . .]*

If you already have a collection of references in the *refdbms* format, this command will add them to a database for you. The default output is the file Newrefs in the database directory $REFDIR, but this can be overridden by specifying an explicit *-o output* option, which can be the name of a file, or of a database directory in which there is a Newrefs file. Some simple syntax and conflict checks are applied to the incoming references; failures result in a return code of 1; if all is well, the return code is 0 (zero).

The *-q* option causes the reference syntax checks to be applied quietly.

In the case of shared databases, references submitted by newref and refsubmit are collected up and added to the database once each night: the new references do not immediately become part of the database.

3.4 Correcting mistakes

Once a reference gets into the database, it may be found to be in error—either by the original perpetrator of this miserable deed or by some helpful colleague. In either case the fix is simple:

1. retrieve the offending reference with refget
2. edit it to correct the blunder
3. add an exclamation point (!) at the end of the %K [tag] line – this will cause the old reference to be replaced with the new contents
4. re-submit the reference to the database

If it is the *tag* that is in error, then proceed as follows:

1. retrieve the offending reference with refget
2. edit it to correct the blunder
3. construct a dummy reference to cause the old erroneous one to be deleted:

    ```
    %z Delete
    %K Oldtag78
    ```
    (blank line)

4. re-submit both references to the database

3.5 GNUemacs reference mode

In addition to the newref command described above, reference entries can easily be created and submitted from the GNUemacs editor running in *reference-mode* (in addition to its editing capabilities, this provides extensive on-line documentation on *refdbms* fields and their formats). By convention, reference-mode is entered automatically for filenames ending in .ref.

A tabular summary of the commands available in reference mode can be found on page 53.

A reference can be created by visiting any file (or buffer), entering reference-mode (either automatically for .ref files, or by hand with M-x reference-mode), then inserting a template with new-reference, which is bound to C-c C-n. This will ask you for the reference type. Type the reference type followed by the RETURN key. (You can use auto-completion here (the SPACE bar). On-line help is available by typing ?.) The package will insert a template containing the appropriate fields for the chosen class of reference into the buffer.

You advance from field to field with TAB, and insert text as usual. If you want to continue a field onto the next line type RETURN (this only succeeds if the field you are editing is allowed to have multiple lines). Typing TAB on a blank entry kills (deletes) the whole line. Typing it at the end of the tag field checks the validity of the tag. Expect a slight delay for this operation.

A field can be justified with M-q. This works even if only the first line is introduced with "%(field-letter)", which is useful when pasting in copy from another source. If the field does not allow multiple lines, it will be made into one line. Frequently references from commercial bibliography services will be in all caps. Providing a prefix argument (C-u M-q) will downcase everything but the beginning of sentences, as well as justifying.

If you are editing multiple references, C-c C-c will replace the line that you are on with the first block of lines in the previous entry of the same type. Thus, if you are entering a whole pile of things from the same journal or proceedings, you can save a lot of typing.

The entire buffer is submitted to the reference system with C-c C-s. This also performs a syntax and validity check on the buffer.

On-line help is available for most fields by positioning the cursor on the line in question and typing C-h r (i.e. reference mode has added an r option to the usual on-line help).[6]

General on-line help about reference mode is available by typing C-h m when in reference mode. More detailed information about each command is available by typing C-h f *command-name*.

The list of abbreviations can be viewed by typing C-c C-a; as an abbreviation's final period is typed, the status line will show you its expansion.

Normally the template you get with new-reference contains only fields appropriate to that type of reference. You can force a template to be generated that contains all possible fields with C-u C-c C-n.

To install the necessary software and control files, consult the Local Guide.

3.6 How it all works

Figure 3 provides a pictorial overview of how all the components of the *refdbms* system play together in the preparation of a bibliography for a latex paper. (The picture looks

[6]**Note:** it is common to remap HP keyboards to reverse the roles of the C-h and DEL keys. In keeping with the original documentation, we retain the C-h form here.

much more frightening than the process actually is!)

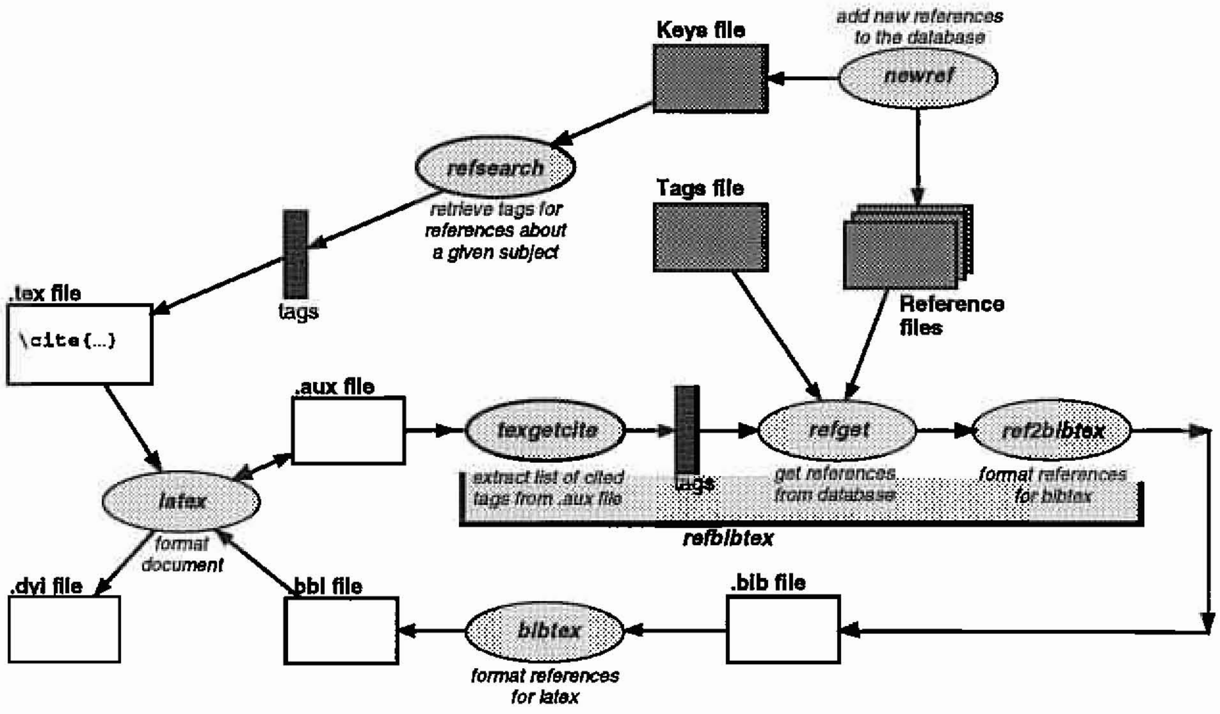
Keys file
add new references to the database
newref
refsearch
retrieve tags for references about a given subject
Tags file
tags
.tex file
\cite{...}
Reference files
.aux file
texgetcite
tags
refget
ref2bibtex
extract list of cited tags from .aux file
get references from database
format references for bibtex
refbibtex
latex
format document
.dvi file
.bbl file
.bib file
bibtex
format references for latex

Figure 3: The overall flow of information in a *refdbms* system being used to build a bibliography for a latex document.

The path starts with new references being added by the newref command (top right hand corner of the figure), and proceeds counter-clockwise through:

- looking for the references to be used with refsearch;
- putting citations into the .tex document with the latex \cite command;
- running latex to make a .aux file
- running refbibtex to: extract the citations (texgetcite), retrieve them (refget), and convert them to bibtex format (ref2bibtex);
- running bibtex to convert them into a form that latex can use to generate the final printable form.

Because there is a loop encompassing latex, refget and bibtex, these commands may have to be executed several times to reach convergence—e.g. if a cited reference contains a citation, which cites another reference . . .

4 Formatting rules for references

This chapter describes the overall structure of references and details about each of the field types. The following guidelines apply to all field types used in *refdbms*:

1. Don't put extra punctuation around items or at the ends of lines. In general, the only acceptable punctuation at the end of a line comes from an abbreviation with a trailing period (which Jr, 1st, 2nd and 3rd do not have).
2. There are two types of field that can have multiple lines: repeatable fields (e.g. %s [submitter]), where each line is a separate entry, and continuable fields (e.g. %x [extract]), where the text just flows from one line to the next. For the latter, editing is simplified if the lines are formatted to be 80 characters or less in length.
3. If only a single line is allowed for a field, simply allow the line to get long if it needs to: although it may wrap around on the screen when you display it, it will still print correctly.
4. Please use the standard abbreviations wherever possible.
5. Use latex forms for accents and unusual punctuation. Use \em for emphasis, rather than \it. Otherwise, leave out formatting commands since they will be inserted later.
6. There are three kinds of dashes:
   - The long or em dash is written ---. It is usually used—without surrounding space—to introduce a parenthetical remark.
   - The medium or en dash is written --. It is used in number ranges (e.g. 6--9), between portions of a technical report number (e.g. HPL--90--27), or as a separator – again, of parenthetical remarks – when surrounded by white space.
   - The hyphen - is used to link words together.
7. Capitalization: basically, don't.
   - *Don't* use Upper and Lower Case for titles—if the original author did so, now is the time to fix it.
   - *Don't* capitalize a word just because it follows a colon.
   - Do capitalize proper names and acronyms. Conference and journal names count as proper ones, as do the names given to research projects and pieces of software. The correct way to capitalize the name of your favourite operating system is "UNIX".

   To quote Jan White:

   > A subset of the capitals-and-lowercase problem is the decree that the first letters (initials) of important words in [titles] be capitalized. This practice evolved in U.S. newspapers in the last century for technical reasons: they ran out of capital letters for headlines and had to invent some alternative means to distinguish headlines from text. With today's technology, such shortages cannot happen ... Nonetheless, this outmoded typographic habit continues in unquestioned use (although only in the United States).

> … Our eyes recognize words as letter-groups by scanning the upper part of the word. Capital Initials Impede and Retard Reading Speed Because They Disturb the Natural Patterns and Relationships of Letters to Each Other. tHIS iS jUST aS sILLY bUT fORTUNATELY wE dON'T sEE iT tOO oFTEN.
>
> To make matters worse, an Up-and-Down Style prevents the reader from noticing proper names and acronyms, both of which use capital letters as distinguishing characteristics. Instead of being visible as the vital references they are, their presence is camouflaged by neighboring words that receive the identical typographical treatment without deserving it.
>
> If you want your product to read smoothly, look contemporary, and be logically crafted, become aware of the dead hand of tradition and get rid of the Up-and-Down Style. Instead, start your [titles] with a capital letter and continue in lowercase (downstyle), as if it were a normal sentence that happened to be important and therefore deserved a bigger and bolder setting. [White88, pp. 34–5]

4.1 Summary of field types

The letters used for the different types of line are shown below. Fields marked with a plus sign (+) can span multiple lines; ones marked with an asterisk (*) can occur several times (as separate fields); the others may occur at most once.

*	%A	author(s)
*	%a	notes about the author(s)
	%B	book title
	%b	bibtex key
	%C	conference name
	%c	conference location and date
	%D	date of publication
*	%E	editor(s)
*	%e	notes about the editor(s)
+	%I	ISBN/ISSN number
	%J	journal name
	%K	tag **(this must be the second field)**
+	%k	keywords for searches
+	%L	known location of document
	%N	part (of a volume or series), or edition
+	%O	public notes (will be printed in some reference styles)
+	%o	private notes (for the delectation and edification of future retrievers)
	%P	page range or list
	%p	publisher or organization
	%R	report number (and/or type of document)
	%S	series title
+	%s	submitter: the person who added the reference to the database
	%T	title
	%V	volume

+ %x	text of extract (abstract or introduction or ...)	
* %y	organizational affiliation of preceding author(s)	
%z	type of reference	**(this must be the first field)**

Notes:

- The %z [reftype] and %K [tag] fields always occur as the first and second fields in the reference.
- Multi-line fields may appear in any order, although all their lines must be together (e.g. you can't have two %k [keyword] lines separated by a different field type).
- The only kinds of fields that can occur multiple times (%A [author], %a [author note], %E [editor], %e [editor note], and %y [organizational affiliation] fields) are designed to allow interleaving of information about authors and editors with their names; the order of the fields is thus significant.

4.2 Formats of the fields in a reference

This section contains a detailed description of each field type.

Field: %A — author

Purpose: Each author has a separate %A field. Choose the longest form of the name available: sometimes names printed on an article itself are longer than those in the table of contents for the journal or proceedings. However, don't add information that isn't on the reference, even if do you happen to know what the person's full name is.

Formatting details: Each initial should have a period and a blank after it; any multicomponent last names should have a backslash (\) before the spaces separating their parts. Be particularly careful with Jr. not everybody precedes it with a comma; in any case, it should not have a terminating period. The de, van, von etc. of French, German, Dutch and Flemish names are included in the surname only if they are in lower case; otherwise they are treated like forenames. If in any doubt, try to see how a professional librarian has treated the name.

If there are any special comments (e.g. translator) put them on a separate %a [author note] line after the last %A line to which they apply. Do not enclose them in parentheses. And *never* resort to et al: always enter all the authors' names.

Examples:

```
%A Andrew J. Wombat\ Jr
%A Catheter de\ Morton-Smythe
%a translator
%A A. B. See\ III
%a and 93 others    — NO! Wrong! Don't do this! Bad idea!
```

Field: %a — author note

Purpose: Comments about the immediately preceding author(s), such as translator.

Formatting details: Do not enclose the text in any punctuation or parentheses. Do not use et al. See %A [author] for examples.

Field: %B — book title

Purpose: The name of the book in which a chapter or segment (e.g. a range of pages) appears. (If the reference was for the whole book, then its title would go in a %T [title] field.)

Formatting details: Do **not** use Upper and Lower case in book titles,[7] and *don't* capitalize small words immediately following a colon.

Examples:

```
%B The wind in the willows
%B A handbook for scholars
```

Field: %b — bibtex key

Purpose: The tag for bibtex to use in preference to the one stored in the reference database. It is only for use with automatically-created databases, where the citation tag needs to be unique (e.g. DSD memo numbers), but the preferred display form is derived from the author's name.

Formatting details: Follow the rules for the %K [tag] field, but don't add a trailing letter to make the name unique.

Examples:

```
%K HPL--CSP--90--1
%b Wilkes89
```

Field: %C — conference name

Purpose: The name of the conference at which the associated paper was presented.

Formatting details: Abbreviations are particularly useful here, because conference names tend to be very long. Don't spell out numbers (e.g. fourteenth): use the numeric form instead (14th). Don't keep in small words like "of the", since the expansion of Proc. will provide them.[8]

Some conferences title their proceedings explicitly as such (e.g. Proc. 7th SOSP.); others do not (e.g. IJCAI, COMPCON). Follow the original on this. If the conference proceedings was also published as an issue of a journal, record that information in the %J [journal], %V [volume], and %N [number] fields as appropriate.

Examples:

```
%C Proc. 7th SOSP.
%C FJCC.
%C 15th Annual Ball-Bearing Convocation
%C Fall COMPCON'80
%C Proc. 1986 SIGMOD Conf. on Management of Data
%C Proc. 14th VLDB.
```

Abbreviations:

ASPLOS.	Intl. Conf. on Architectural Support for Prog. Lang. and Operating Sys.

[7]This is a change from previous editions of this guide.

[8]This is a change from previous editions of this guide.

EUUG.	European UNIX Systems User Group
FJCC.	Proc. AFIPS Fall Joint Comp. Conf.
ICDCS.	Intl. Conf. on Distrib. Computing Sys.
ICSE.	Intl. Conf. on Softw. Eng.
IJCAI.	Intl. Joint Conf. on Artificial Intelligence
IWSSD.	Intl. Workshop on Softw. Specification and Design
NCC.	Proc. AFIPS National Comp. Conf.
OOPSLA.	Object-Oriented Programming Sys., Lang. and Applications Conf.
PODC.	Proc. of the Princ. of Distrib. Computing Conf.
POPL.	Annual Symp. on Principles of Prog. Lang.
SJCC.	Proc. AFIPS Spring Joint Comp. Conf.
SOSP.	ACM Symp. on Operating System Principles
VLDB.	Intl. Conf. on Very Large Data Bases

In addition, all the abbreviations for organizations, schools, and publishers available in the %p [publisher] field are valid here.

Field: %c — conference location

Purpose: Where and/or when a conference was held.

Formatting details: Include the significant parts of an address: usually the town and state or country. Separate the parts by commas. Dates should be as in the %D [date] field (i.e. in European style—note the use of the en dash '--'); include them only if they provide additional information beyond the publication date for the proceedings. If the conference happened in the USA, use the two-uppercase-letter postal abbreviations for the state (e.g. CA); otherwise include the country unless there is no possibility of ambiguity. Abbreviations are the same as for the %C [conference name] field.

Examples:

```
%c Bretton Woods, NH, May 1985
%c Paris, TX, 11--23 Oct. 1983
%c Bristol, MA
%c West Berlin
```

Field: %D — date

Purpose: The date of publication of the reference.

Formatting details: Always abbreviate month names to their first three letters and a period (except for May, June and July). Year numbers should be written out in full. Only rarely should the day of the month be included; if it is (e.g. in a date for a conference or a specific draft of a document), format it in European style (day month year), without any commas.[9]

Examples:

```
%D 1983
%D Dec. 1983
%D 14 Aug. 1925
%D 11--14 June 1925
```

[9]Even though this may not be your preferred choice, adhering to this convention will make it possible to convert back and forth between American and European styles automatically.

Abbreviations:

Jan.	January
Feb.	February
Mar.	March
Apr.	April
May.	May
Jun.	June
Jul.	July
Aug.	August
Sep.	September
Sept.	September
Oct.	October
Nov.	November
Dec.	December

Field: %E — editor

Purpose: The names of the editor or editors of the collection in which the item appears. Each editor is put in a separate %E field.

Formatting details: The same as for %A [author] fields.

Examples:

```
%E Peter Rabbit\ Jr
%e panel moderator
```

Field: %e — editor note

Purpose: Comments about the immediately preceding editor(s), such as session chair.

Formatting details: The same as for %a [author note] fields. See %E [editor] for an example.

Field: %I — ISBN/ISSN

Purpose: The ISBN (or ISSN) number for the item, or document in which the item appeared.

Formatting details: The ISBN or ISSN number, with en-dashes "--" between the components of the number. Be sure to include the "ISBN" or "ISSN". Multiple numbers (e.g. one for a hardback, one for a paperback) can be supplied on separate lines.

Examples:

```
%I ISSN 0--394--40904--3
%I ISBN 0--19--861121--8
%I ISBN 0--19--861122--6 (paperback)
```

Field: %J — journal

Purpose: The name of the journal in which the article appeared.

Formatting details: Since journal names are proper names, the first letters of Significant Words in Journal Names are Capitalized. Abbreviations are strongly encouraged here because different bibliography styles have very different degrees of expansion for their journal names.

Abbreviations:

ABLTJ.	AT&T Bell Labs. Tech. J.
ACTA.	Acta Informatica
BSTJ.	The Bell Sys. Tech. J.
CACM.	Communications of the ACM
COMPJ.	Comp. J.
COMPSURV.	ACM Compt. Surveys
HPJ.	Hewlett-Packard J.
IBMJRD.	IBM J. of Res. and Dev.
IBMSYSJ.	IBM Sys. J.
IEEECOMM.	IEEE Trans. on Comm.
IEEECOMP.	IEEE Trans. on Computers
IEEESE.	IEEE Trans. on Softw. Eng.
IEEESOFT.	IEEE Trans. on Softw. Eng.
IPL.	Inf. Processing L.
JACM.	J. of the ACM
JCSS.	J. of Comp. and Sys. Sciences
JSS.	J. of Sys. and Softw.
OSR.	Op. Sys. Review
SCP.	Sci. of Comp. Prog.
SEN.	Softw. Eng. Notes
SICOMP.	SIAM J. on Comp.
SIGPLAN.	SIGPLAN Notices
SPE.	Softw.—Practice and Experience
TCS.	Theor. Comp. Sci.
TOCS.	ACM Trans. on Comp. Sys.
TODS.	ACM Trans. on Database Sys.
TOGS.	ACM Trans. on Graphics
TOMS.	ACM Trans. on Mathematical Softw.
TOOIS.	ACM Trans. on Office Inf. Sys.
TOPLAS.	ACM Trans. on Prog. Lang. and Sys.

Field: %K — tag

Purpose: The tag by which this reference will be uniquely identified. It must always be the second field in a reference.

Formatting details: The tag should be the surname of the senior (first) author concatenated with the last 2 digits of the year of publication. If there is already a different reference with the same tag in the database, resolve the ambiguity by appending a letter ('a', 'b', ...) to the result. (The first such tag in the database should have no trailing letter.)

A tag may contain only letters and digits; it should start with a capital letter unless the primary author's name does not. Spaces and punctuation in an author's name should be omitted in multi-part last names (e.g. van Jacobson becomes vanJacobson). Don't include a trailing Jr or similar in the tag.

If there is no author's name to use for the tag, use the primary editor's name (if there is one), or the publisher's name instead. In the latter case, use the obvious short form of the name, if there is one (e.g. IBM for International Business Machines).

Examples:

```
%K Wombat83
%K MortonSmythe37a
%K Wilkes82a
%K IBM83f
%K vanWijngaarden68
```

Field: %k — keyword

Purpose: Keywords (index entries) that will be helpful in finding this reference again in the future.

Formatting details: The keywords need not be on separate lines: they can be separated by white space and/or punctuation. Words in the title and the author's names are automatically included, and need not be entered again (unless they are Proper Names, or not in the preferred forms shown below).

Only letters and digits will be retained in the index; words that occur more than once, and "noise" words (e.g. the, and, it) will be suppressed. Upper and lower case are *not* equivalent in keywords: an upper case letter will prevent the keyword from being fed through the wordstemming algorithm, so if a Proper Name occurs in the title, please re-include it in a %k line with at least one capital letter.

Above all, be generous in allocating index terms. Put far too many in rather than not enough. Provide lots of different levels of abbreviations (e.g. database *and* dbms), and American spelling. Remember that these are to help you find this reference when you can no longer remember its existence, not simply to help you locate something whose name you have temporarily forgotten.

Examples:

```
%k uniting frenzies, Australian sociology
%k SouthEast Asia mammals
%k marsupials, herbivores
```

All for an article entitled "Wombats of the world – unite!".

Field: %L — location

Purpose: The physical location of a copy of the document. Include this if there is any chance at all that it might be tricky to find in the future.

Formatting details: A single line of text saying where a copy may be found. Please be as specific as possible.

Examples:

```
%L John Wilkes' reference filing cabinet.
%L HPL corporate library, catalogue number ...
%L New York Metropolitan Museum has the only extant copy.
```

Field: %N — number

Purpose: The part or sub-part of a book, volume, etc, in which the reference appears. Use this field to record the edition or version (in which case, use numerical form (2nd) rather than spelling it out). If you want to supply an ISBN number, it goes in the %I [ISBN/ISSN] field, rather than here.

Examples:

```
%N 6
%N 123, part B
%N 2nd edition
%N 7th edition, virtual VAX--11 version
```

Field: %O — public note

Purpose: Comments that will be printed *every* time this reference is included in a bibliography. Typical comments include: a forward reference to an erratum published later; the language if other than English; a \cite{...} pointer to the source for which this is a secondary citation or a translation, restrictions on the availability of the original.

Observations or summaries don't belong here: they go in the %o [private note] portion. A citation for a conference proceedings that is also a journal issue does not belong here either: use the %J [journal], %V [volume], and %N [number] fields instead. Unpublished items are required to have a %O field to say what they are.

Citation strings placed here will cause the associated references to be extracted in bibliography styles that include the text from %O fields.

Formatting details: There may be as many lines as you like, each begun with a %O. End the whole thing with a period.

Examples:

```
%O See \cite{Hoare74a} for an erratum.
%O HP Internal Use Only.
%O In French.
%O Private communication.
%O Cited from \cite{HPphonebook88}.
```

Field: %o — private note

Purpose:

Notes about the item that don't get printed out in any normal bibliography style. The most valuable use of this field is to provide a capsule summary (precis) of significant findings or results in the article—this is often much more useful than typing in the whole of a rather unenlightening abstract. (Imagine other database users reading it, as well as you, a few months from now.) Value judgements are particularly helpful.

Formatting details: The notes can be on as many lines as desired, provided each commences with a %o. Remember that they should observe latex formatting conventions.

Examples:

```
%o The standard coroutine scheme, with justification and
%o performance comparisons---as implemented on the
%o Cambridge IBM/370 and in Tripos. Nicely presented.
```

Field: %P — pages

Purpose: The page numbers that this reference spans.

Formatting details: Use a space between numbers. Do *not* elide leading digits in the second number: write them out in full. Use a + after a page number to indicate that following pages are non-contiguous (e.g. in a magazine article "continued on page 278"). If there are several page number ranges, separate them with a comma and a space.

Examples:

```
%P 10 23
%P 324 326
%P 76 87, 278+
```

Field: %p — publisher

Purpose: The name of the publisher or institution that put out the document.

Formatting details: You should normally enter at least a minimal form of their address. Abbreviations are encouraged.

Examples:

```
%p CSDEPT., Univ. of Wisconsin
%p Xerox PARC.
%p McGraw-Hill, New York
%p Stanford Univ., CSDEPT.
%p Sun Microsystems Inc., 2550 Garcia Ave, Mountain View, CA 94043
```

Abbreviations:

ACM.	Assoc. for Compt. Machinery
AE.	American Elsevier, New York
ANSI.	American Natl. Standards Inst.
AP.	Academic Press, London and New York
AW.	Addison-Wesley, Reading, Mass. and London
BBN.	Bolt, Beranek and Newman Inc.
BCS.	British Comp. Soc.
BSI.	British Standards Inst.
CHI.	Comp. and Human Interaction Conf.
CMU.	Carnegie-Mellon Univ., Pittsburgh, PA
CSDEPT.	Comp. Science Dept.
DECSRC.	DEC. Sys. Res. Center, Palo Alto, CA
DECWRL.	DEC. Western Res. Lab., Palo Alto, CA
DECWSL.	DEC. Western Softw. Lab., Palo Alto, CA
DEC.	Digital Equipment Corp.
DEPTCS.	Dept. of Comp. Science
DEPTCSE.	Dept. of Comp. Science and Eng.
DEPTCSEE.	Dept. of Comp. Science and Elec. Eng.
DEPTEECS.	Dept. of Elec. Eng. and Comp. Science
ECMA.	European Comp. Manufacturers Assoc.
EE.	Elec. Eng.
HPL.	Hewlett-Packard Labs.
IBM.	Intl. Business Machines Corp.
IEE.	Inst. of Elec. Engineers
IEEE.	Inst. of Electrical and Electronics Engineers
IRISA.	IRISA (Institut de Recherche en Informatique et Systèmes Aléatoires)

JW.	John Wiley, New York
MCG.	McGraw-Hill, New York
MITLCS.	Lab. for Comp. Science, MIT.
MIT.	Massachusetts Inst. of Technology, Cambridge, MA
NBS.	Natl. Bureau of Standards
NH.	North-Holland, Amsterdam
PARC.	Palo Alto Res. Center, CA
PH.	Prentice-Hall, Englewood Cliffs, NJ
PHI.	Prentice-Hall Intl., London
SV.	Springer Verlag, Berlin
UCBCS.	Comp. Sci. Div., Dept. of Elec. Eng. and Comp. Sci., UCB.
UCB.	Univ. of California at Berkeley
UCCL.	Univ. of Cambridge Comp. Lab.

Field: %R — report number

Purpose: The technical report number, ordering number, ISBN number or the Library of Congress catalog number for this document.

Formatting details: Include the type of number being inserted, e.g. Technical report *unless* the field contains only a technical report number made up solely of digits, capital letters and punctuation.[10]

Use an en dash '--' between numbers or pairs of capital letters, not just simple hyphens. Theses are often published as technical reports: don't do anything special beyond noting the report number.[11]

Examples:
```
%R HPL--CSP-91--14
%R 11
%R Order number GC28--0629
%R NTIS number 1245--5678
```

Field: %S — series

Purpose: The name of the series of which the book (etc.) is a part.

Formatting details: Formatting is as for the title of a book. Do *not* use Upper and Lower Case (see the %T [title] field description for more details).[12]

Examples:
```
%S Lecture notes in computer science
```

Field: %s — submitter

Purpose: The electronic mail address of the person who submitted the reference for inclusion in the database. If the data in the reference was obtained from anything other than a copy of the reference itself (i.e. a secondary citation – see p. 14), this is the place to record that: the reference should have one field for the original author, and another for

[10]This is a change from previous editions of this guide.
[11]This is a change from previous editions of this guide.
[12]This is a change from previous editions of this guide.

you. (The reference must also have a %O [public note] field recording this as a secondary citation with a \cite command referring to the secondary citation source.)

Formatting details: A valid electronic mail address, usually of the form loginname@address. It may include additional information in brackets after the name, such as the date. Multiple entries may occur if one person submitted the entry and another has modified it.

Examples:

```
%s wilkes@hplabs.hp.com
%s jacobson%cello@hplabs.hp.com [Mon Jan 29 18:55:46 PST 1990]
```

Field: %T — title

Purpose: The title of the object being referenced.

Formatting details: Titles should be converted to lower case everywhere except in Proper Names, even if the original used Upper and lower case (see p. 21 for reasons). There are no exceptions.[13]

Do not capitalize the first word after a colon just because of its position. If a title has two parts not otherwise separated, put a colon between them. Be especially careful about spaces around dashes, and about the difference between a medium, or en dash '--' and a long, or em dash '---' (p. 21.)

Examples:

```
%T Horus---a remote procedure calling system
%T Distributed systems -- an advanced course
```

Field: %V — volume

Purpose: The volume name or number in which the reference appears. This may be a journal volume, or a volume in a series of books.

Formatting details: If the item has two or more parts, separate them with commas. Many IEEE publications have volume names with both letters and digits, as in the last example here.

Examples:

```
%V 7
%V 12, part B
%V SE--12
```

Field: %x — extract

Purpose: An extract quoted directly from the abstract of the referenced item itself.

The idea is to give a feeling for what the item is about, using the authors' own words. You don't have to include the entire abstract, or even contiguous sentences—try instead to hit the key points. (A full abstract is often tedious to read, while a couple of short sentences can get the main points across much more effectively.)

[13]This is a change from previous editions of this guide.

Occasionally you can give a better idea of the reference by using some text from the body text (e.g. the conclusion) rather than the abstract. If so, go ahead, but also include an annotation about the source, as in the examples here.

Formatting details: The notes can be on as many lines as desired, provided each commences with a %x. Don't add text of your own: that belongs in a %o [private note] field.

Examples:

```
%x In this paper, we show that tadpoles can never, for
%x purely arithmetic reasons, grow up to be frogs.

%x (Editorial note.)
%x Because of the controversial nature of their findings,
%x the authors have chosen to remain anonymous.

%x (From the conclusion.)
%x In summary: the idea was not a success.
```

Field: %y — organizational affiliation

Purpose: Organizational affiliations of the author(s) as recorded on document being referenced. (This usually means at the time of the document's creation.)

Formatting details: A single line of text. Abbreviations are as for %C [conference name] and %p [publisher] fields. %y lines aren't printed in most reference styles, whereas %a [author note] lines are.

Place a %y field after all the authors affiliated to one institution. One such line will cope with several authors. There can be more than one affiliation line if authors come from multiple places – in such a case, insert the %y lines between the %A [author] lines, much as they appear in the document itself.

Examples:

```
%A Fred Bloggs
%A Andy Capp
%y CMU.
%A Joe Somebody
%a translator
%y Univ. of Hamburg, West Germany
```

Field: %z — reftype

Purpose: The kind of reference. This must always be the first field in a reference.

Formatting details: Only one of the following keywords may be used. Case is significant.

Article	in a journal
InProceedings	an article from a conference
TechReport	technical report, but not a manual
Book	all of it
InBook	a chapter, or a range of pages
Manual	about a product, program, etc.
PhDthesis	doctoral theses only: Master's theses are TechReports
Proceedings	the whole of a conference proceedings

UnPublished	not formally published
Miscellaneous	unclassifiable
Delete	never occurs in the database: used to remove a reference

5 Database file formats

This chapter documents the formats of the various control files used in *refdbms* databases. Chapter 4 (on page 21) explains how the references themselves are formatted.

5.1 Tags files

The Tags file is an inverted index on the reference files. It contains tuples, one per line, of the form: tag, file-number, byte-offset. The tuples are sorted by their tags; fields are separated by white space.

Instead of storing the full filename in each tuple, an index into a table of filenames at the beginning of the Tags file is used. This table has one filename per line; the file-number used in the tuples is the line number containing the filename. The list of filenames is terminated by a blank line.

Filenames are treated as relative to the directory in which the Tags file resides, to simplify *refdbms* use from a remote machine (e.g. via an NFS mount). For example, the first few lines of a large database Tags file might look like this:

```
refsA
...
refsY
refsZ

abbott81 1 43274
abdelhamid89 1 34279
abelson85 1 27494
...
```

5.2 Keys files

The Keys file is a set of tuples, one per line, of the form: keyword, tag, tag, ...

To avoid very long lines that would break commands such as sed, keywords that map to many tags are given multiple lines, each looking like a complete tuple. (This also allows fast incremental update of the Keys file when new references are added.) The tuples are sorted by keyword.

A random sample of lines from a Keys file might look like this (the funny spelling of "accelerator" is a side-effect of the automatic word-stem algorithm in use):

```
academia brassard87
accelerater arnould89 huffer87
accent baron85 fitzgerald85 fitzgerald86 jul85 myers86 perq84 rashid80 rashid81
accent rashid81a rashid86 spector86 wendorf87 zayas87
accept liskov84 wegman86
...
```

5.3 Expansion control files

An expansion file contains three columns of information, separated by white space:

1. a list of field letters that are valid for this abbreviation (e.g. Ccp); an asterisk (*) in this column means "all valid fields";
2. the abbreviation (with no following period);
3. the expanded version of the abbreviation.

Expansions are processed in order, so that expansions defined near the beginning of the control file may be further modified by later expansions. A comment line is introduced by "#" at the beginning of the line. For example:

```
# Conference names
Cc   EUUG    Eur. UNIX Sys. User Group
Cc   FJCC    Proc. AFIPS Fall Joint Comp. Conf.

# Journals
J    ABLTJ   AT\\\&T Bell Labs. Tech. J.
J    ACMCS   ACM Computing Surveys
J    ACTA    Acta Informatica

# Long-form names of the months
Dc   Jan     January
Dc   Feb     February
Dc   Mar     March

# Common expansions; also be used in lines above.
*    Arch    Architecture
*    Comm    Communication
*    Conc    Concurrent
*    Comp    Computer
```

Be careful with expansions that contain characters special to sed, such as "\" and "&". Prefix any such character with one or three "\" characters to protect them—first against sed, then (if needed) against latex.

The two standard expansion control files files (Expand-long and Expand-short) are derived from a common source file, Expand.cpp, via the C preprocessor /lib/cpp.

5.4 The "wordsToIgnore" file

The wordsToIgnore file is used to provide a list of "noise" words that are discarded when building the keyword index (the Keys file). Its format is simple: an exclamation point in column 1, followed by white space, and then the word to ignore. There is one word per line; case is not important. A "#" in column one causes the rest of the line to be treated as a comment. For example:

```
# A sample from the standard wordsToIgnore file
! a
! about
! an
! and
! as
! at
! be
```

5.5 Help files

The directory $REFDIR/Source/Newrefs contains a number of prompt files that are used to provide help information in newref and GNUemacs reference-mode. The filename extension is always .prompt; the contents are the help text. The text is displayed when a question mark is given in response to a newref prompt.

```
# A sample from the standard wordsToIgnore file
! a
! about
! an
! and
! as
! at
! be
```

5.5 Help files

The directory $REFDIR/Source/Newrefs contains a number of prompt files that are used to provide help information in newref and GNUemacs reference-mode. The filename extension is always .prompt; the contents are the help text. The text is displayed when a question mark is given in response to a newref prompt.

6 Private commands

The following commands help maintain *refdbms* databases, or are used internally by the public commands. They are documented here for completeness; most users can simply skip this entire chapter.

The phrase "this binary" implies that the command being described is a compiled executable rather than a shell script.

6.1 Database maintenance commands

The following commands are used for updating and maintaining the reference database, and various sundry other tasks.

make

The file $REFDIR/Makefile is a control file for the make command to perform the following functions:

- creating a Newrefs file if one doesn't exist;
- if Newrefs is non-empty, running mergenewrefs on it, and then emptying it;
- rebuilding the Tags and Keys files if needed.

Invoking make is the normal way to handle all of these functions; checks are built in to try to make sure that no information is overwritten by accident.

Don't use this file if you want to build yourself a private *refdbms* database: there is a better template file to start from (consult the Local Guide for where it can be found).

mergenewrefs *[-BqeD] [-o dbms] [-m maillist] files ...*

This command adds new references into a database, and replaces and deletes references already there. The *-o dbms* option names the directory holding the database to be used: the default is the file References in the current directory.

If the *-B* flag is used, the database is assumed to be in "big" format, which is designed for large numbers of references. In this kind of database, references are divided into 26 different files: one for each letter of the alphabet, corresponding to the first letter of their tags. The files are called refsA, refsB, and so on. The *-o dbms* option names the *directory* in which these files are to be found. The default value is the current directory if *-B* is used.

The *-q* option suppresses reassuring messages. The *-e* option causes the input files to be emptied (truncated to zero length) when mergenewrefs completes successfully. The *-D* option is for debugging: it prevents any lasting updates to the reference database.

The first step in the merge is to run checknewrefs (p. 39) to perform some simple syntax checks on the new references. If all is well, the references are added to the database and the Keys file updated incrementally. Mail announcing the new references is sent to the distribution list *maillist* if supplied. (Items in the list should be separated by commas.)

If mergenewrefs finds any errors it exits with status code 1.

To edit or replace a reference, put an exclamation point "!" onto the end of the tag field, and then resubmit the reference to the database: the old version of the reference will then be excised before the new one is added.

References can be completely deleted from the database by specifying a reference type of **Delete** in the %z [reftype] field. This is normally only used if a tag on a reference in the database turns out to be erroneous and needs to be completely deleted. (Hint: this should never happen!)

The Tags file should be rebuilt after running mergenewrefs to reflect the new entries. In addition, the Tags file has to be up to date *before* running mergenewrefs because it is needed by the internal checknewrefs command.

The Keys file only needs to be rebuilt if deletions have occurred: otherwise, it will already have been updated (incrementally) by mergenewrefs.

mergenewrefs takes out a lock to prevent simultaneous updates to the database. The file is called mergelock, and is put in the directory of the database being updated. It may be necessary to delete this if it gets accidentally left around from a run of mergenewrefs that dies a horrible death for some reason.

checknewrefs *[-aq] [-o output] files . . .*

This command is used to check that the syntax of the to-be-added references is correct and that there are no duplicate tags. (For example, it is invoked by the mergenewrefs command (p. 38 before it tries to merge in the new references.) Warning: its checks are by no means complete!

Input is taken from stdin if no files are given on the command line. The *-q* option suppresses reassuring messages about successful progress.

The *-o output* option names the directory or file against which tag conflict checks are performed. (The default directory is the value of the environment variable REFDIR). The conflict checks look for duplicate tags in the input files, and between the input files and the output database. The Tags file should be up to date before running checknewrefs, because it is used in the check for tag conflicts.

By default, checknewrefs checks to see whether there are any tag conflicts between the input files and the output. (If no explicit output file is specified, Newrefs is used.) This check can be suppressed with the *-a* option, which allows you to check references in such output files themselves.

If checknewrefs finds any errors it exits with status code 1, or 2 for more serious trouble. Otherwise it emits status code 0 (zero).

buildrefkeys *files . . .*

Builds a new Keys file (on stdout) from the given reference files. Keys are derived by applying a word-stem algorithm to words in the %B [book title], %k [keyword], and %T [title] fields. The %A [author] and %E [editor] fields are also used, but these are not subjected to the word stem algorithm.

buildreftags *files . . .*

Makes a Tags file (on stdout) from the input files given. These filenames should be relative to the directory in which the Tags file is to reside, so that access across a network mount point (e.g. NFS) will be possible.

sortrefs *files . . .*

The input files should contain references. The output from the command (on stdout) is the references sorted alphabetically by their tags. stdout must *not* be redirected to any of the input files. If two references with the same tag occur in the input, the command exits with status code 1.

refmatch *-e pattern . . . files . . .*

This command is used to scan files for references that match one or more of the given patterns. The patterns should be in the style of awk line-matching commands: if they are regular expressions, don't forget the leading and trailing / characters. The input files should contain references. The output from the command (on stdout) is a list of tags that match the patterns given. For example:

```
        refmatch -e '/%z Book/' -e '/%z/ && $2 == "InBook" '
```

will find (in a slightly convoluted way) all references that have a %z [reftype] field of *Book* or *InBook*.

6.2 Internal commands

Internal commands are those that are called from within the other *refdbms* commands. They should not normally be invoked directly.

texgetcite *files . . .*

Given one or more latex .tex or .aux files (or its standard input), this binary will extract a list of citations (from bibtex \cite and \nocite commands) and write the list to its standard output, one per line, ready to sort and feed to refget.

texgetcite is used by refbibtex (p. 12).

refinitials *[-b] [files . . .]*

This binary shrinks authors' and editors' first names to their initials. It handles hyphenated surnames, hyphenated first names, TEX accents, and all sorts of other goodies. Spaces preceded by a '\' are considered part of a name, so that Roland Hedley\ Jr abbreviates to R. Hedley\ Jr. Input is taken from the list of files given, or stdin. The *-b* option causes last names to be enclosed in curly braces to protect them from bibtex.

refinitials is run automatically by ref2bibtex (p. 12) and refmaker (p. 13).

Ref2bibtex

This binary does the massaging of references in the *refdbms* format into the format used by bibtex .bib files. It is used by ref2bibtex and refbibtex. (Note: it can be used only as a filter, since it takes no command-line arguments.)

Ref2bibtex is used by ref2bibtex (p. 12).

maker2tags *files*

This scans one or more FrameMaker files and generates a list of tags. maker2tags is used by refmaker (p. 13); see the description of that command for restrictions and bugs in the scanning algorithm.

ref2mif *[-D]*

This command takes in *refdbms* format data and generates FrameMaker .mif on stdout. It uses bibtex and a variant of the regular refalpha style file to generate the formatted data in FrameMaker .mml format, and then converts that to .mif. This command requires that perl be installed.

ref2mif is used by refmaker (p. 13).

refwordstem

This binary takes in a list of words (one per line), and applies a word stem algorithm to them, in an attempt to reduce them to their shortest form. White space is deleted along the way. Words that contain an upper case character or a non-alphanumeric are passed through unchanged, except that they are translated to lower case. (Note: it can be used only as a filter, since it takes no command-line arguments.)

refwordstem is used by buildrefkeys (p. 39) and by refsearch (p. 10).

Refsearch ...

This binary takes the same arguments as the refsearch command, and does almost all the real work of looking up keys. The difference is that Refsearch does not apply the word stem algorithm to its arguments.

Refsearch is used by refsearch (p. 10).

checkrefsyntax *files . . .*

Used by the checknewrefs command (p. 39) to do some simple format checks on its input files. Input is taken from stdin if no files are given on the command line. If it finds any errors it exits with status code 1 for less severe infractions, or 2 for must-be-fixed errors; both types of error should be corrected.

checkrefsyntax is used by checknewrefs (p. 39).

checkreftags *inputfiles . . .*

The *inputfiles* should be tag files; this command looks to see whether there are multiple references with the same tag. If it finds any, it exits with status code 1. It is used by the checknewrefs command (p. 39).

analyzereftypes *[-i letters] files . . .*

This command is used to generate a torture test for the ref2bibtex command (p. 12).

The input files should contain references. The output from the command (on stdout) is an analysis of the references by reference type (the %z [reftype] field), further broken down by the combination of fields they contain. Fields may be removed from consideration by including them in the list given with the *-i* option.

7 Local guide

This local guide has been written for Hewlett-Packard Laboratories in Palo Alto, CA. Much of what it covers may apply to your site, too, but several parts are going to be site-specific.

7.1 Software installation

To install *refdbms* and start using it on your own system:

1. Become super-user and run:

   ninstall *-vh hplacs1 refdbms*

   You should redo this occasionally to take advantage of the latest *refdbms* improvements – once a week is probably adequate.

   The ninstall package puts the *refdbms* commands in /usr/local/bin/, and the GNUemacs files in /usr/local/emacs/lisp/CSPlocal/. Some of the private commands are not distributed since they only apply to large-database builders.

   At some point in the future, it will be possible to ninstall both the private commands and the master sources. If you need access to them in the meantime please contact the author.

2. Set the environment variable REFDIR to point to the reference database sources and communal database (these instructions work for the C-shell):

   netunam */net/cello rfa*
   setenv *REFDIR /net/cello/users/wilkes/lib/references*

   Note 1: Soon, this will change to be an NFS mount point.
   Note 2: Users of the ACSadmin.group ninstall package have REFDIR set for them in the /etc/csh.login file. If you aren't an ACSadmin subscriber, you can do this in your own .login file.

3. Send mail to *john_wilkes@hplabs.hpl.hp.com* to let him know that you have started to use the database. That way, you can be notified when:

   (a) a new software release occurs, or if the guidelines change
   (b) new versions of the documentation come out
   (c) additions are made to the communal database.

4. To enable automatic loading of the GNUemacs package, add the following lines to your .emacs file:

```
(autoload 'reference-mode "ref-mode"
 "Establishes a mode for editing refdbms references."
 t nil)

(autoload 'new-reference "ref-mode"
 "Inserts a template for refdbms.  Prefix arg means all
```

```
         possible template lines, not just those appropriate
         for the type." t nil)

        (fset 'Reference-mode 'reference-mode)
        (fset 'ref-mode 'reference-mode)
```

If you are using *refdbms* on Cello, and have the standard ACSadmin environment set up, then most of the above has already been done for you. In particular, the environment variable REFDIR is already set to point to the reference database sources and communal database, and the auto-startup GNUemacs code is already installed in the GNUemacs startup file.

The master copies of the *refdbms* system sources live on the machine Cello in the directory /users/wilkes/lib/references. In addition to the communal database and the internal scripts and prompt files, this contains:

Commands	holds *refdbms* shell scripts
Commands300	holds *refdbms* executable binaries for Series 300 systems
Commands800	holds *refdbms* executable binaries for Series 800 systems
Source	auxiliary files (e.g. awk scripts) used by the commands

The private database Makefile template is the file /users/wilkes/Source/Makefile.template. (The one in /users/wilkes/lib/references/Makefile is specialized for the communal database, and less generally useful.)

7.2 Local commands

memonum2ref *files . . .*

Given a set of memo-number files (e.g. as generated by the cspmemo or salmemo commands), reformats them into a *refdbms*-format file of references on the standard output. For details on the specific format that is accepted, please see the memonum2ref script itself.

7.3 The communal database

Part of the local *refdbms* system is an established database of bibliography entries. This *communal database* represents a few years' worth of reference collecting. Its entries describe papers, conference proceedings, books, and technical reports. Most of them are in the fields of distributed operating systems, computer graphics, software engineering, databases, and Egyptology. At the time of writing, the database contains roughly 3000 references (about 2.4 Mbytes of data).

Because the communal database began life as a personal one—and to a large degree still is—there are certain idiosyncratic properties that are imposed on it, and on the behaviour of anybody who chooses to add to it. Primary amongst these are rigid adherence to the entry quality guidelines on page 14. If you find these restrictions overly irksome (some do!) you are welcome to use the communal database in a read-only fashion, and create a private one of your own. Of course, this means that your colleagues cannot then benefit so readily from all your hard work in assembling references.

It is important that no Hewlett-Packard company confidential material be added to the communal database. This allows it to be exported freely to researchers outside HP (including me, should I ever leave the company).

End of sermon.

The communal database is stored in the "large database" format (see p. 38 for details). Its references are stored on Cello in the files $REFDIR/refs[A-Z].

When references are added to the communal database with newref, refsubmit, or GNUemacs reference-mode, they aren't immediately added to the database: instead, they accumulate in the file $REFDIR/Newrefs, and are merged into the main database late at night by an automated daemon. If you want to force an immediate update, change directory into $REFDIR and type make.

Mail for updates to the communal database is sent to the mailing list ref-updates@cello.

References

[Alexander87] J. C. Alexander. *Tib: a TeX bibliographic preprocessor.* Department of Mathematics, University of Maryland, 1987. Version 2.1.

[Kernighan78] Brian W. Kernighan and Lorinda L. Cherry. A system for typesetting mathematics. *Communications of the ACM*, **18**(3):151–6, March 1978.

[Kernighan81] Brian W. Kernighan. A typesetter-independent TROFF. Computing Science technical report 97. Bell Laboratories, Murray Hill, NJ, 1981.

[Knuth88] Donald E. Knuth, Tracy Larrabee, and Paul M. Roberts. Mathematical writing. Technical report STAN–CS–88–1193. Department of Computer Science, Stanford University, January 1988.

[Lamport85] Leslie Lamport. *LaTeX: a document preparation system.* Addison-Wesley Publishing Company, Reading, MA, 1985.

[Lesk78] M. E. Lesk. Some applications of inverted indexes on the UNIX system. Computing Science technical report 69. Bell Laboratories, June 1978.

[Patashnik88] Oren Patashnik. Bibtexing, 8 February 1988. Overview document distributed with Bibtex.

[Reid80] Brian K. Reid and Janet H. Walker. *Scribe introductory user's manual*, 3rd edition. Unilogic Ltd., 605 Devonshire St., Pittsburgh, PA 15213, May 1980.

[Reid81] B. K. Reid. *Scribe: a document specification language and its compiler.* PhD thesis, published as Technical report CMU–CS–81–100. Carnegie-Mellon University, Pittsburgh, PA, 1981.

[vanLeunen78] Mary-Claire van Leunen. *A handbook for scholars.* Alfred A. Knopf, New York, 1978.

[White88] Jan V. White. *Graphic design for the electronic age.* A Xerox Press Book, Watson-Guptill Publications, New York, 1988.

Index

Quick reference guide

1. Install *refdbms* and set the environment variable REFDIR to the name of the directory in which the *refdbms* system lives: see the Local Guide for details (page 43).

2. If you have more than one *refdbms* database, set the environment variable REFPATH to list them (see p. 9).

3. To use the reference database:

command	*function*
refsearch *[-a] keyword* ...	Find tags given keywords
refget *tag* ...	Get references
reflook *[-a] keyword* ...	Get references given keywords
refstrip	Filter references down to tag+title
latex *file*.tex ⇒ *file*.aux refbibtex *file*.tex ⇒ *file*.bib bibtex *file* ⇒ *file*.bbl latex *file*.tex ⇒ *file*.dvi	Make a bibliography for a latex document called *file*.tex
newref	Add references
refsubmit *file*	Add an existing file of references
refget *tag* > file.*ref* – *edit* file.*ref* – *fix tag:* %K tag! refsubmit file.*ref*	Correct a reference

4. With latex, use the following:

   - \bibliographystyle{refalpha}
     \bibliography{file}
   - \cite{Tag89} or \nocite{Tag89,Tag91a} to cite references, using the *refdbms* tag (don't put any spaces in the \cite command's argument).

   Note: you may need to run latex a couple of times because of the way that latex handles forward references—and citations are a form of forward reference. If any of your cited references themselves cite other references, you may also need to re-run refbibtex.

5. With FrameMaker, put the tags in [square brackets].

6. The following table summarizes the GNUemacs reference mode commands and their default key bindings.

command	*binding*	*notes*
reference-mode	*(none)*	
new-reference	*C-c C-n*	*C-u* ⇒ all fields
next field	*TAB*	
continue field	*RETURN*	
justify field	*M-q*	*C-u* ⇒ downcase too
abbreviations	*C-c C-a*	shows current list
copy field	*C-c C-c*	from last reference
refsubmit	*C-c C-s*	
help on fields	*C-h r*	
help on pkg	*C-h m*	
help on command	*C-h f*	*command* as argument