

Originally published in
the Journal of the
Evangelical
Theological Society,
46:3 (Sept 2003),
485-495.

Windows Software for Bible Study
Van Parunak
4/4/2003

1 Introduction

Bible software has come a long, long way. The procedure for doing computerized biblical studies in the late 1970's began something like the joke about the recipe for elephant soup: "First, catch an elephant." There were no publicly available texts of the Bible, so one had first to devise a coding scheme and type in the data to be searched. There were no standard search mechanisms, so one had to code the search program in a language such as Fortran, C, or Pascal. Many computer systems were batch processors; running a query meant loading a stack of punched cards into the hopper and going out for lunch while waiting for the program to reach the top of the queue and execute. This process required prolonged residence at the computer center, and turn-around was often measured in hours. The field was accessible only to scholars who combined philological and computer skills, or who could form a close team to bring together the needed capabilities.

Today, the standard biblical texts are widely available, along with programs that offer a wide range of search capabilities. The user confronting this embarrassment of riches faces a challenging decision. In comparison with the state of the field twenty-five years ago, there are no bad decisions today. We would have given our eye teeth for any of the offerings now available. Still, a Bible student contemplating the expenditure of several hundred dollars for one of the leading commercial packages may hesitate among the alternatives, or wonder whether the difference in functionality over a freeware package is worth the price. This review is intended to help in such decisions.

Due to the hardware environment available to me, this review is limited to packages for Microsoft Windows. Unfortunately, this restriction means that it cannot include the highly acclaimed Accordance package from Project Gramcord. In addition, the Gramcord package itself is in the midst of a major technical migration from a 16-bit implementation to a state-of-the-art 32-bit implementation, and at the time of writing is not available to include in the review. (Disclosure #1: I am a long-standing fan of Gramcord, since learning of it in 1980. The first installation of the mainframe version of Gramcord outside of Paul Miller's personal environment was at the University of Michigan under the auspices of my postdoctoral work there. Paul and I spent many evenings together translating from the native Univac Pascal in which the program was originally written to the compiler available on the Amdahl processor at UM, and I implemented the first extensions of Gramcord to search Hebrew texts. I have been an eager user of Gramcord in its various incarnations, including the MS-DOS version running on a 4.77 MHz PC-2 and the now aged 16 bit version. Paul's pathbreaking vision pointed the way for all of the packages described here, and continues to be the background against which I view other packages. When the new Gramcord is available, I'll be eager to compare it against the baseline in this review. Disclosure #2: I have never been an employee or paid consultant of any publisher of Bible software, including Gramcord.)

The focus of this review is on Bible study, which I understand in the sense of studying the biblical text (as opposed to reading books about the Bible). I thus concentrate on software functions that support reading, searching, and displaying the text, and consulting reference works

that are oriented to a particular passage (such as lexical, grammars, and commentaries), rather than more general digital library capabilities. The area of digital libraries is important, particularly for people with limited shelf space or financial resources, and merits a separate review. Still, to some extent every package reviewed here is a “library” that helps the user interact with multiple electronic “books” (including biblical texts and reference works). In this discussion, the word “resource” refers indiscriminately to any such electronic book.

The review is organized as follows. Section 2 gives summary information about the six packages reviewed. Section 3 discusses the heart of any Bible software, the texts of the Bible that it makes available for study. Sections 4, 5, and 6 discuss three categories of functionality that are important for Bible study: searching the biblical text, consulting reference works related to a passage being studied, and integrating the student’s own notes and observations with the system. Section 7 discusses how users can get help in using the packages. Section 8 concludes.

2 The Packages

This review covers six software packages that are currently available, summarized in Table 1.

Table 1: Packages Reviewed in this Article

| Abbr. | Name | Version | Interface | Free? | URL | Cost |
|-------|---|--------------------------------------|---------------|-------|---------------------|--|
| BWk | BibleWorks for Windows | 5.0.037q | Control Panel | No | www.bibleworks.com | \$299.95, + \$197 for HALOT 4 and BDAG 3 |
| BWn | Bible Windows | 7.0 | Desktop | No | www.silvermnt.com | \$195.95 |
| eSw | e-Sword | 7.0.0 | Control Panel | Yes | www.e-sword.net | \$15 for CD |
| Lib | Logos Bible Software Series X—Scholar’s Library | Libronix Digital Library System 1.1a | Desktop | No | www.logos.com | \$599.95 |
| OLB | On-Line Bible | 1.20 | Control Panel | Yes | www.onlinebible.net | \$34.95 for CD; \$64.95 with NIV, NAS, NRSV, and NLT, \$109.95 including Scourby narration |
| Swo | Sword | 1.5.5a | Control Panel | Yes | www.crosswire.org | \$3 for CD |

Here’s what each of the fields in this table describes.

Abbr.—I’ve assigned a three-letter abbreviation to each package to simplify comparisons later in the review. For ease of reference, I’ve ordered all tables by these abbreviations.

Name.—This is the name by which the package is commonly known. It’s the name on the box, and usually appears on the splash screen when the package launches.

Version.—All of these packages are constantly being upgraded. The designator in this field (accessible in the “Help/About” selection from the menu bar on any Windows program) shows the specific version on which my comments are based. An older or newer version of a program may have different features or performance I report. (Usually, things get better over time, but not always.)

Interface.—The interface to a package can be classified as either “control panel” or “desktop.” In a “desktop” layout, each open resource appears in a window that can be positioned freely on the computer screen. Dialog boxes supporting functions such as search definition or opening new resources appear only when invoked from the menu bar. In a “control panel” layout, resource windows and selected dialog boxes are arranged in predefined positions on the screen. A common control panel layout is to display the Bible text in one window, dictionaries and other word-based information in a second, and commentaries or other reference-based resources in a third. BWk and OLB are hybrids. BWk presents the basic information (e.g., biblical text, selected lexicon, search results, user notes) in a predefined layout, but most resources appear in separate windows. Until recently, OLB used the desktop model, and now it presents resource windows (including both the biblical text and commentaries) in a “mini-desktop,” with others (dictionary information) in the fixed panes of the control panel.

Free?—Three of the packages can be downloaded with no charge from the web, while the other three must be purchased. Free packages can have fees associated with them. A typical download is tens of megabytes, so users without a high-speed connection will want to order a CD, for which a distribution fee is usually charged. These packages must also charge significant license fees, imposed by publishers, for access to more recent Bible translations. Though distributed freely, these packages are not in the public domain, and users can copy and redistribute them only under the terms of the software licenses included with each package.

URL.—Each package has a website, providing a variety of features. The free packages can be downloaded from these locations. A package’s website also may provide supplementary materials contributed by users or offered for sale, technical support information, user groups, and similar material.

Cost.—The prices listed in Table 1 are from the programs’ respective websites. For the commercial programs, significant discounts are often available from online booksellers.

3 Texts

The heart of any Bible software package is the text of the Bible—or texts, since all of the packages reviewed here contain multiple versions. In addition to the basic texts offered, packages differ in the supplementary information (e.g., morphology) encoded with each text, and in how they enable comparisons across versions.

3.1 Availability

Many packages include a wide selection of versions in both English and other languages that will be of little interest to the majority of users. If you need a version such as Esperanto, Bulgarian, or Tagalog, chances are one of these packages offers it (in these specific cases, Swo), but most readers would not be served by collating these here. Table 2 summarizes the versions that most readers will find of interest for serious study. Any entry in a cell means that the version represented in the row is available for the package represented in the column. “X” simply

indicates availability, while other characters indicate the presence of supplementary information, described in the following paragraphs.

Table 2: Available Texts and Supplementary Information

| | BWk | BWn | Lib | eSw | OLB | Swo |
|-------------------------------|------|------|-----|-----|-----|-----|
| KJV | ST | X | ST | S | ST | SM |
| ASV | X | X | X | X | X | X |
| NIV | X | | X | | X | |
| NASB | S | | S | S | X | |
| BHS | KVAM | KVAM | KVM | X | X | KVA |
| LXX | AM | AM | AM | | X | SM |
| NA27 | AM | | AM | V | | |
| Friberg | AM | AM | | | | |
| WH | AM | X | X | | | SM |
| MT (Robinson- Pierpont) | AM | | X | | | |
| MT (Scrivener) | AM | X | A | AS | ST | SM |

English Texts.—In addition to the basic text, English versions often include two valuable categories of linguistic information: Strong’s numbers (indicated by “S” in the table) and Tense-Voice-Mood information for verbs in the original languages (“T” in the table). Strong’s numbers are derived from the numbering scheme for Hebrew and Greek lemmata in Strong’s Concordance. This supplementary information can be a great help for a user whose facility with Greek and Hebrew is not great enough to work directly with the original language text, but who recognizes the lexical skewing that occurs in every major translation. Swo includes a complete morphological analysis (“M”) for the Greek words underlying the NT.

Hebrew.—The Hebrew text in all cases is based on the encoding of BHS that we produced at the University of Michigan in 1980-81, as subsequently refined by the CATSS project at the University of Pennsylvania and Westminster Theological Seminary. The original encoding included vocalization, accents, and Kethiv-Qere readings, but some of this information has been stripped out in some packages. The codes in Table 2 indicate whether the package includes Kethiv-Qere (“K”), vocalization (“V”), accents (“A”), and morphology (“M”). Morphology information (in both Hebrew and Greek texts) includes the Tense-Voice-Mood information offered by the “T” option in English texts. While both BWn and BWk include accents, BWk’s coding includes the useful classifications in J. Price’s system [7].

Display of pointed Hebrew text is a nontrivial problem. All of the displays are legible, but Lib’s display shows several anomalies. A spurious space often appears between bound prepositions, articular *he*, and sometimes *waw*, and the word to which they are attached. Sometimes *waw* is too close to the first consonant of the word. Consonants are sometimes cut off at the beginning of the line.

LXX.—Several packages include the Rahlfs Septuagint, sometimes with Strong’s numbers (“S”), accents (“A”), and morphology (“M”).

Greek New Testament.—Both the Nestle-Aland 27th edition and the Robinson-Pierpont edition of the majority text are available, sometimes with Strong’s numbers (“S”), Tense-Voice-Mood (“T”), variant readings (“V”), accents (“A”), and morphology (“M”). The Friberg text has the same underlying Greek text as NA27, but has a distinctive morphological code that includes functional and discourse information not represented in a traditional encoding, as documented in [2]. BWk also provides a morphologically analyzed UBS3 text.

3.2 *Linked Versions*

It is often useful to display two versions of the same text at the same time, to compare them with one another. Various packages offer three different approaches to this functionality: linked windows, interverse, and interlinear arrangements.

Linked windows allow the user to open two different translations, each in its own window, to the same passage, and then link the windows so that the translations scroll together as the user moves from one verse to the next. Linked windows are supported by BWk, Lib, and eSw. OLB offers a version of linked windows: each window that is open to a particular passage offers a set of tabs for all available versions. Pressing the tab for a version shows that version, starting at the first verse displayed in the window. While the different versions are not concurrently visible, the user can shift rapidly among them.

Interlinear arrangements allow the user to specify a set of versions, whose verses are then aligned word-for-word in a single window. Such a display is extremely difficult to generate automatically for an arbitrary pair of versions. For any of its original language texts, BWn can generate an interlinear of that text aligned with the English translations of each word. Lib offers an interlinear version of NA27 as a separate volume in the collection, not a view dynamically generated from an original language text.

An interverse arrangement is an approximation to an interlinear, consisting of a single verse presented in each of several selected versions. Although the individual words are not aligned, most verses are short enough that the user can quickly relate the different versions to one another. BWk and eSw offer interverse arrangements. Lib has a tool to line up multiple versions side-by-side, verse-by-verse, visually highlight the differences between them, and report the statistical deviation from a user-specified base version.

4 Search

Most users of Bible software want the ability to search the biblical text for passages that satisfy certain descriptive criteria. The available packages differ in the criteria that can be specified in searches, the interface by which a user specifies a search, the form in which the results can be presented, and how rapidly a search is performed.

4.1 *What can I search for?*

Programs differ in the kinds of patterns they can find in the biblical text. Table 3 summarizes various levels of search capability.

Table 3: Search Capabilities

| | BWk | BWn | Lib | eSw | OLB | Swo |
|----------------------|-----|-----|-----|-----|-----|-----|
| String of Characters | R | ? | R | | | R |
| Single Word | X | X | X | X | X | X |
| Phrase | X | X | X | X | X | X |
| Boolean | X | X | X | LY | L | L |
| Proximity | X | | C | | | |
| Strong's | X | | X | | X | X |
| Morphology | X | X | X | | | |
| Agreement | MSL | M | | | | |
| Alignment | X | | | | | |

String of Characters.—The simplest search to implement is one that specifies a string of characters, perhaps with wild cards (symbols that can match any character, or a string with specified characteristics). Many word processors support at least a subset of a class of patterns known as “regular expressions.” For example,

- a set of characters in brackets means “any character in the set,” so ‘[aeiou]’ would match any vowel;
- an asterisk ‘*’ after a pattern matches zero or more instances of that pattern, so ‘c*’ would match “c”, “cc”, “cccccc”, and so forth;
- a plus sign ‘+’ after a pattern matches one or more instances of the pattern;
- two patterns separated by ‘|’ indicate that either of the two may occur;
- parentheses around a string indicate that the string is to be treated as a whole for applications of special characters such as ‘*’, ‘+’, and ‘|’, so that “(man)|(men|)” will match either “man” or “men”, while “man|men” would match “manen” or “mamen”.

Technically, it’s relatively easy to write a computer program to find regular expressions, which is why they’re so commonly available. Microsoft Word can perform such searches in a generic text of (say) the KJV, downloadable from any one of a number of on-line sources for free. At first glance, regular expressions seem to be very powerful, but a little experimentation shows their inadequacy for exegetical work. They are at too fine a level of granularity. The expositor typically needs to work at the level of the word, not the character, and regular expressions by themselves are too cumbersome. For example, the pattern “man” will lead one to passages in the text that include the word “men”, but also to those including “commandment,” “Manasseh,” and a host of other instances of the string “men” that have nothing to do with the word “men”. Generating a search that yields only the word “men” requires specifying explicitly all the features that can indicate word boundaries, including not only spaces and punctuation, but also line boundaries. In addition, string searches are case sensitive, so that “men” does not match “Men”. “R” in the table indicates support for regular expressions in searching.

Single Word.—The most common search capability is for single words, with the program taking the responsibility of recognizing word boundaries. This functionality provides the electronic

equivalent of a concordance. The simplest versions require the user to specify differences such as singular or plural separately, so that “tent” would not retrieve “tents,” but Lib, given the singular of a noun, will attempt to find its plural as well (unless the “nostem” operator is specified). A search for “tent” will also find “tents”, and one for “mess” will find “messes”, but a search for “mouse” won’t find “mice.” All packages provide some mechanism to extend the search to include simple affixes, such as the use of a wild card like “?” to match any single character or “*” to match any string of characters, so an attentive user can include suffixed plurals in searches in the other packages as well, if desired.

Phrase.—A phrase search looks for a specified string of words. In a phrase search, “Lord God” would return all verses in which the phrase “Lord God” appears, but not those including “God the Lord.”

Boolean.—A Boolean search allows the user to specify words and then find verses that contain all, any, or none of the specified words (in any order). For example, one might want to find all verses in a book that do not refer to God, or all that contain either “Lord” or “God”. Some programs offer only partial Boolean capability. For example, the interface may support finding verses with all specified words (“L”), but not those with any (“Y”) or with none (“N”).

Proximity.—A proximity search allows the user to find patterns that are not confined to the same verse, but appear in different verses within so many characters, words, or verses of one another. This capability is important for topical studies, in which one typically wishes to find paragraphs rather than isolated verses. Proximity searches in Lib are restricted to a single chapter (“C”). That is, they cannot extend across chapter boundaries.

Strong’s.—An extremely helpful feature for the student with limited Hebrew and Greek skills is the ability to search on Strong’s numbers, thus retrieving all instances of the same original word regardless of how it is translated. Strong’s searches are only possible in texts that include Strong’s numbers, but the mere presence of these numbers does not guarantee that a program supports searching for them.

Morphology.—Packages that include morphological coding in original language texts also provide a way to search this coding, typically with the option of leaving some fields unspecified. Thus, for example, one can search for masculine plural nouns, or for *Qal* verbs in the prefix conjugation. The richer the morphological coding, the richer the search possibilities: only BWk and BWn encode Hebrew accents, so only these packages support searches on accents.

Agreement.—An important extension to morphological searching is the ability to constrain two words to agree with one another. For example, one might want to find all the verses in the Greek New Testament that have a noun modified by the adjective *kalos*. The user does not care about the number, gender, or case of the construction, so long as the noun and the adjective agree. BWk and BWn allow the user to specify that all the words in a search must agree with one another morphologically (“M”). In addition, BWk has two agreement features that other tools do not offer. First, users can specify agreement at the level of lexical items (“L”), not just morphology, for instance, “give me all the places where the Hebrew particle *gam* joins two forms of the same verbal root” (as in Gen. 27:33). Second, its advanced search engine (ASE) can specify agreement among different subsets of the words in a construction (“S”). For instance, in Greek, one might be looking for participles separated from their articles by a long string of words, in which the article and the participle agree in number, gender, and case, but none of the intervening words agrees with them.

Alignment.—An important clue to the meaning of many Greek words in the New Testament is their usage as translations of Hebrew terms in the LXX. Ideally, one would like an interlinear Hebrew-LXX text that could be searched to determine the concordance between Greek and Hebrew vocabularies in different grammatical contexts. None of the products offers such a tool, but BWk’s ASE can identify verses between two different texts that contain specified patterns in each text. Thus it can quickly generate possible alignments for subsequent checking, a great improvement over the manual process of flipping back and forth between the body and the index of Hatch-Redpath while leafing through Rahlfs and BHS.

4.2 Interface

The most powerful search capabilities are useless if a user cannot access them in a straightforward, intuitive way. Programs provide a variety of means for users to formulate a search, summarized in Table 4.

Table 4: Search Interfaces

| | BWk | BWn | Lib | eSw | OLB | Swo |
|-----------------|-----|-----|-----|-----|-----|-----|
| Cursor | X | X | X | | X | |
| Dialog Box | MC | MC | MC | C | C | C |
| Graph Structure | X | | | | | |
| Command Line | D | | D | | | |
| Refinements | C | B | N | | | |

Cursor.—A search is often motivated by the exegetical principle that usage determines meaning. Confronted with a phenomenon in the text, the student would like to examine other instances to see what they may have in common. In this setting, the most natural action is to put the cursor on the word or phrase in the text and, with a mouse click, say, “Please find me more like this.” BWk and OLB do not even require the user to click on the word of interest, but display relevant information (lexical, morphological, and TSK references for BWk, dictionary entries for OLB) in a supplementary window as the user moves the cursor over the text.

Dialog Box.—A dialog box search interface presents the user with a field in which one or more words can be typed, and sometimes with several pre-defined options. Two common sets of options are those defining constraints on the Cooccurrence of the listed words (“C”; for example, must all of the listed words appear, or only some? Must they occur in the order listed? May other words intervene?) and those specifying Morphological features (“M”). The dialog box can be used alone, or with a cursor interface to modify the specific form found in one location in the text.

Figure 1: A Dialog Box Interface from BWN

For example, Figure 1 shows BWN's dialog box for grammatical searches in Hebrew. The user can identify a series of words, each one described morphologically, and define the Boolean relation between each pair of successive words, the number of intervening words allowed, and where the first word must fall in a verse.

Graph Structure.—Even a command line can become unwieldy if users wish to specify complicated patterns of agreement, or alignment between different texts. For such applications, a more natural representation is a graph in which the nodes represent individual words and the edges represent relations among those words. Figure 2 shows an example of a graphical search specified in BWk, exploring an extended participial construction.

- The central horizontal line requires an article, a preposition, a noun, and a participle in that order, with optional separating words.
- The two boxes above the central line of boxes require that all four of these elements must appear in a verse for a match to succeed, and that at least seven words separate the article from the preposition.
- The three boxes extending vertically below the central line specify case agreements, among the four words and (referencing the lower-left box) among any other words that intervene between the article and the preposition.

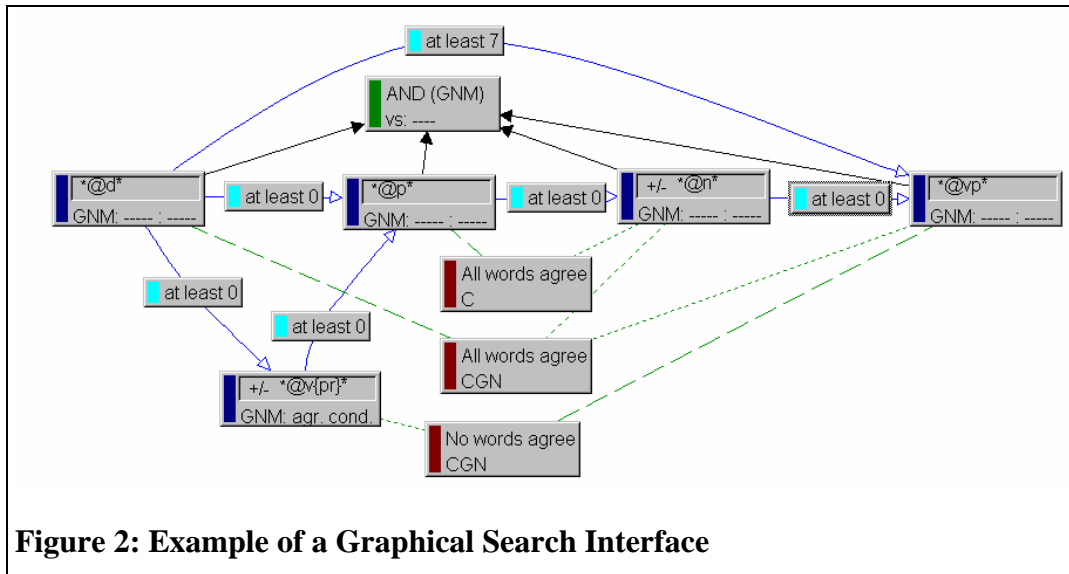


Figure 2: Example of a Graphical Search Interface

Command Line.—Dialog boxes face a trade-off between expressiveness and ease of use. The more options they present, the more complicated they become, and some options (such as Boolean combinations among different words) are difficult to specify in a dialog interface. In such cases, the most powerful interface may be a command line in which users can type a detailed textual description of their search. For example, both BWk and BWn allow users to include accents in searches of the Hebrew Bible. BWn’s dialog box allows users to specify only one accent per word, but BWk’s command line includes regular expressions that permit a user to search for (say) every place a word occurs with any of the disjunctive accents.

Formulating such a request can be formidable for novice users. Both programs that support command lines provide a dialog box (“D”) as a memory aid in constructing morphological specifications for command line queries, but users will need to devote a fair amount of effort to learning the details of the command line language in order to get the most out of one of these packages.

BWk’s command line language has two classes of tokens: those that specify individual words in the biblical text, and those that specify grouping or cooccurrence relations among these words. Here are several examples.

Consider the impact of the definite article on המקום in Gen 28:11. The common English translation “a certain place” is indefinite, influenced no doubt by the uncertainty implicit in the verb פגע. Does the usage of the definite state elsewhere warrant this, or is the author reminding us that this is a very special place, the one where Abraham previously offered sacrifice to God in 12:8? A search for המקום alone turns up many cases where the article is part of the idiom הזה המקום. To avoid this, we search for the noun *not* followed by the demonstrative, using the ‘!’ operator to exclude the unwanted demonstrative:

המקום הזה!

The initial single quote indicates that the word tokens must follow one another in the order given (which in this case is specified right-to-left, conforming to the usual Hebrew order).

Windows Software for Bible Study

Sometimes we want to match more than just word forms. In studying Isaac's final declaration to Esau in Gen 27:33, I wanted to find other cases where the Qal passive participle בָּרוּךְ is followed within a reasonable distance (say, three words) by a Qal imperfect of the verb הִיָּה. BWk returns four cases with the following command:

```
'הייה@vqi* *3 ברוך@vqs*
```

The '@' sign separates the root form (in italics here, in Hebrew characters in the program) from the morphological codes, and the '*3' means that up to three words may intervene.

Most English translations skew the distribution of lexemes with respect to the original text. That is, they do not uniformly translate each original lexeme with a consistent English one. A well-known example is the translation of υἱός, properly "son" but sometimes rendered "child" or "children" in the AV. To find all verses in the AV that include both the word "child" or "children" and the Strong's code 5207 (for υἱός), one enters

```
.child* 5207
```

where the initial '.' indicates that all of the listed words must occur in the verse (in any order), and "child*" matches any word that begins with the letters "child" (including "child", "children", and "childless").

It is instructive to observe that this query, like many, is approximate. It will yield some verses (like Matt 11:19) in which both "son" (translating υἱός) and a word beginning with "child" appear. One could further refine this search to exclude verses including the words "son" or "sons" by using the negation operator "!", thus:

```
.child* 5207 !son*
```

Even this could in principle fail; υἱός is translated "foal" in Matt 21:5, and if this verse happened to contain "child" or "children" as well, it would be returned even though "child" would not necessarily be the reflex of υἱός. Automated searches in any Bible software package are a tremendous time-saver for the student, but their results must be reviewed manually if they are to support accurate linguistic inductions.

Lib's query language is much more complicated, partly because it is not restricted to searching for linguistic constructs within a single text, but must support searches across multiple resources. Lib's queries are formulated from terms (a single word or a quoted string), combined with operators specifying Boolean combinations and proximity among terms. These elements can be qualified by "Fields" that name different sections of a document (e.g., in a Bible, the text itself, the words of Christ as a subset of the text, and footnotes). "Data types" are distinct classes of information (such as biblical references or morphological codes), and "modifiers" change the matching rules for a term (e.g., enabling wildcards or regular expressions, turning case sensitivity on, turning stemming off). The result can be greater complexity in command line queries.

Lib does not support negation of a term within a phrase construction, so the search for המקום without following הזה is somewhat more complex than in BWk:

```
הזה 2 BEFORE המקום) NOTEQUALS המקום
```

That is, “find all verses that contain **הַמְקוֹם** where the **הַמְקוֹם** in question is not followed within two characters by **הַזֶּה**.”

Simple morphological queries are very similar to those in BWk. The search for **בָּרַךְ** and **הָיָה** in Lib is,

```
[בָּרַךְ=vqs??????] BEFORE 50 [הָיָה=vqi??????]
```

The “50” indicates the maximum number of characters that may come between the two terms. (Lib specifies proximity in characters, rather than words.) However, the Lib interface does not give access to some of the morphological features encoded in the BHS database, and sets them to presumed default values in order to do the search. For example, in the query under discussion, it sets the “Jussive?” field to “False,” and as a result does not find instances (such as Deut 33:24) where the verb **הָיָה** is jussive, instances that BWk does recover.

Lib’s equivalent for the query for translations of **ὑἱός** by derivatives of “child” becomes

```
child* AND greekstrongs=5207
```

“greekstrongs” is a *data type*, distinct from the actual biblical text, and must be specified explicitly in order to access this information in a Lib search. By distinguishing different classes of information in this way, Lib’s query language is more discriminating. The KJV does not use digits in the text, but the NASB does, and is commonly available with Strong’s numbers. A search of the NASB for ‘666’ in BWk hits both the Strong’s number for **ἄπουσία** in Phil 2:2 and the digits in the text in 1 Kings 10:14, 2 Chr 9:13, and Ezr 2:13, a conflict that Lib will not make. The tradeoff is that the user must learn the data types involved, a task that is made difficult because some of them (including “greekstrongs”) are not documented in the Lib help system.

Lib’s AND operator, like the ‘.’ in BWk, is satisfied as long as its arguments occur in the same verse, so this query is subject to the same ambiguity as the BWk query discussed above. Lib also provides Boolean operators that restrict their attention to a single location within a verse. The construction

```
child* ANDEQUALS greekstrongs=5207
```

returns only those verses where the same word that begins with “child” is associated with the Strong’s number 5207.

When the text being searched is Hebrew or Greek, a user needs to enter both Roman and non-Roman characters in a command line. A very convenient feature of the BWk interface is that it is able to “guess” the appropriate font with reasonable success. In Lib, the user must explicitly switch keyboards while formulating the query.

Lib uses the standard Modern Hebrew keyboard for typing Hebrew at the command line. While this is a defensible choice, a more natural keyboard mapping for many scholars would be to follow the conventional transliteration of biblical Hebrew wherever possible, as we did in encoding BHS and as BWn and BWk have implemented in their interfaces. Thus (for example), the Hebrew root **גָּדַל** is typed “gdl” in BWn and BWk but “dsk” in Lib.

While the free packages are intended primarily for use with English texts, it is possible to cut and past Hebrew or Greek text into the search box and search these versions. OLB can search for single Hebrew words (either by root or by textual form), but not for Hebrew phrases. In Greek, it

can search for words or phrases. eSw can search for words or phrases in Hebrew or Greek, as can Swo. However, the counts given by these programs do not always agree with those from packages that advertise Hebrew and Greek capabilities, and they are not recommended for such use.

Some packages offer functions that, while not strict searches, provide information that one might otherwise seek through a search. BWn can generate a concordance for a specified book, sorting references either by their words or by their morphological tags. eSw offers an “analyze” function that compiles a list of all the words (and optionally, Strong’s numbers) that occur in a specified stretch of text, ordered by their frequency. This function can help a student quantify leading themes in various sections of a book, or in various books. A similar list of words ordered by frequency is one of the options provided by BWk’s Report Generator, as well as a list of lemmas with all inflections ordered by frequency (similar to BWn’s morphological concordance). Lib offers several useful reports by clicking on a word in a text. The Lemma Report generates a list of all instances of a root (from a Hebrew text) or a lemma (from a Greek text), sorted by morphology, much in the spirit of Mandelbrot’s concordance. For English versions with Strong’s numbers (KJV and NASB), Lib has a special cursor-based search capability inspired by the venerable Englishman’s Concordances, generating a key-word-in-context list of all passages that contain the Strong’s number associated with the selected word.

4.3 Results

Packages differ in how they let the user manipulate the results of the search, summarized in Table 5.

Table 5: Presenting Search Results

| | BWk | BWn | Lib | eSw | OLB | Swo |
|-----------------|-----|-----|-----|-----|-----|-----|
| List of Hits | X | X | X | X | X | X |
| Verse List | DE | | S | E | E | E |
| Graphic Display | X | | | | | |

List: The most fundamental way to present the results, supported by all packages, is a list of the references that match the query, with the text of the reference. (Lib lets the user turn off the text display, which can affect search speed.)

Verse List: Some packages let the user store a list of hits for later reference. In some cases, this list is Static (“S”); one can save, load, or sort the list, or copy the items listed to the clipboard. Other packages let users Edit (“E”) the list by adding or deleting verses. BWk offers a much more Dynamic (“D”) verse list capability. Users can combine multiple verse lists using operations such as

- union (“Give me all the verses that occur in either of these two lists.”)
- intersection (“Give me all the verses that occur in both of these lists.”)
- difference (Give me all the verses in list A that do not occur in list B.”)

This facility lets a user combine the results of individual queries to reveal information that would not otherwise be obvious, and for which a single query might be overly complex.

Graphic Display: Sometimes the results of a search can be best understood if presented graphically. BWk’s detailed statistical report offers a number of options for such a graphical display. The fundamental values that it can display include the number of hits (that is, the specific words matched in a search) in each book or chapter, or the number of verses containing a hit in each book or chapter. Both counts can be normalized by the number of words or verses in the unit that is plotted, or by the number of words or verses in the version. The plot can be ordered either in normal book order or in increasing or decreasing order of the statistic.

For example, Figure 3 plots the occurrence of imperative verbs in Ephesians, and clearly shows the bipartite structure commonly associated with Paul’s epistles, with exposition in the first half and exhortation in the second. The plot calls our attention to the anomalous imperative at 2:11, but shows that it is much less prominent than the imperatives in chapters 4-6. Examination of the text shows that this is a rhetorical command to “remember” a fact, which is consistent with the expository trend of the first half.

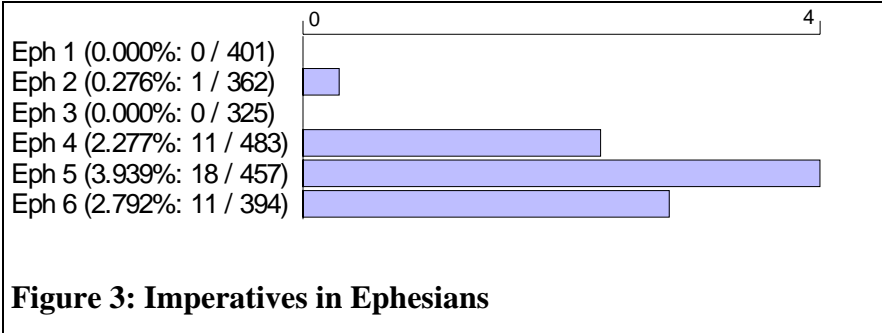
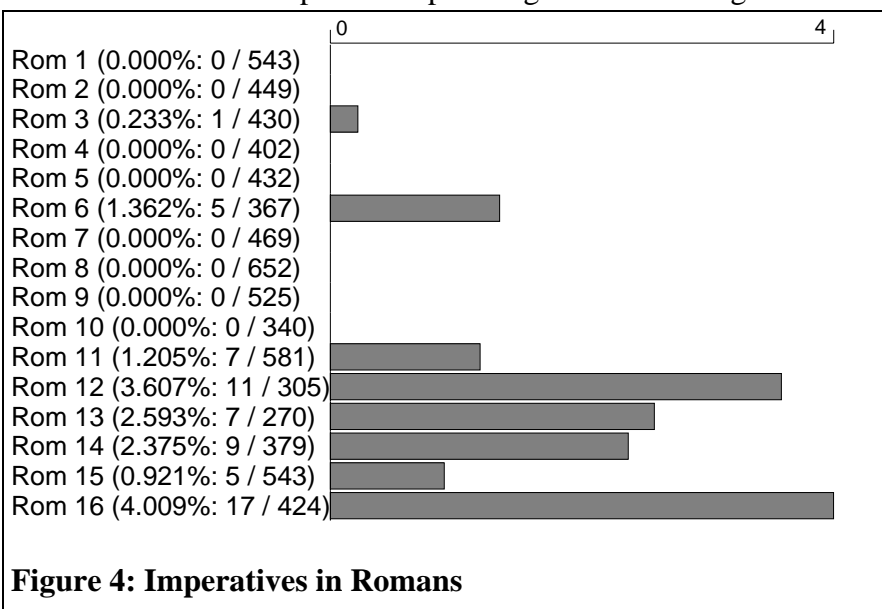


Figure 4 plots imperatives in Romans. Again, the classic bipartite division is clear. The single imperative in chapter 3 (“let God be true”) can easily be considered as primarily expository in its effect. Two other features, though, encourage us to

develop a more nuanced understanding. First, chapters 9-11 are clearly an integrated thematic section dealing with Israel, but the distribution of imperatives does not align with the thematic break between 11 and 12. This overlap may be explained as a linked keyword transition [5], in which the writer anticipates an upcoming section and begins to exhibit its characteristics at the end of an earlier section.

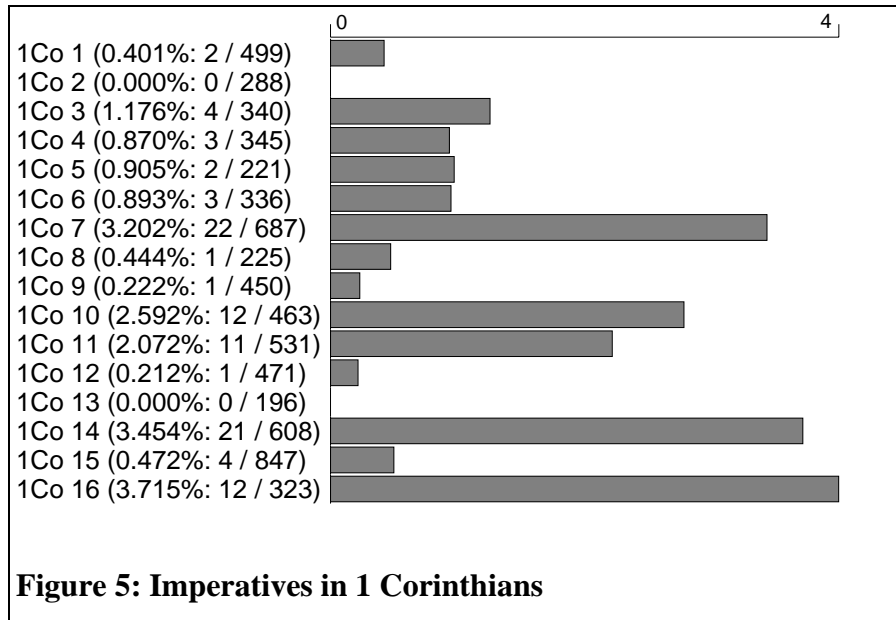


though, encourage us to develop a more nuanced understanding. Second, the five imperatives in chapter 6 cannot be dismissed as rhetorical devices in an otherwise expository section. The chapter is unambiguously hortatory, and provides the theological foundation in the first part of the book on which the more extended exhortations of chapters 11-16 rest.

The distribution of imperatives in 1

Corinthians (Figure 5) suggests that the “exposition plus exhortation” model does not apply here, at least not at the book level. In fact (though not implied by this plot), the book is structured around a series of topics, some suggested by the Corinthians, others initiated by Paul.

Plots such as these are potentially very valuable in exploring the structure of biblical texts. At first glance, it seems natural to



plot frequency statistics per chapter, but this approach has several weaknesses. Chapters do not necessarily correspond to the natural discourse units of the text, either in extent (a natural unit may be wider or narrower than a chapter) or in their limits (which may not correspond with those of natural units). The same can be said of fixed width windows that are sometimes used in plots of this sort (for example, plotting occurrences in windows ten verses wide). This mismatch results in a profile that distorts the actual structure of the text. A much better approach is to let the window width change dynamically with the distribution [4], an algorithm that could be easily implemented by any of these packages. With this refinement, plots such as these become powerful tools for visualizing the structure of texts. For instance, the peaks in plots of first person singular and second person plural grammatical forms in Galatians clearly marks the major divisions in that book’s argument [6], but these features are not visible with plots at the chapter level.

It would be even more useful if software packages provided an option to generate a file containing, not verse references, but the index number of each hit in a search, together with the number of words per verse and per chapter, so that users could directly manipulate distributional information in a package such as Excel or Mathematica. A further refinement would be to let the user define and annotate a number of fields with each hit to capture contextural features (e.g., direct vs. indirect or human vs. divine speech, putative literary source), and provide a simple flat-file database function (sorting and searching) to help the user perform supplementary studies.

4.4 Timing

One of the most important characteristics of a Bible software package for me personally is searching speed. I often build searches incrementally, starting with a simple search, then refining it to narrow in on what I want. If each search takes more than a second or two, my thought process is interrupted. This experience is in line with research in computer interface design. A recognized authority on web site design describes 0.1 second response as ideal, 1 second as maximum acceptable, and 10 seconds as likely to cause users to leave a site and look elsewhere

[3]. In one experiment, if a process lasted longer than 8.5 seconds, users assumed the computer had frozen and rebooted [1].

For the purposes of this review, I timed a variety of search tasks on the packages being reviewed. My platform is a 700 mhz Toshiba Portege 3490 CT with 256M RAM, running Windows 2000. I timed searches using the XNote Stopwatch program (<http://www.xnotestopwatch.com>). This program lets the user define a hot key (I use F11). Each time the hot key is pressed the program records a time stamp in a log. To time a search, I set up the search, then do whatever action is needed to execute the search (such as pressing <Enter> or clicking with the mouse) concurrently with pressing F11, and then press F11 again when the search is complete. The greatest inaccuracy in this approach is my own response time in pressing F11 at the end of the search, but comparisons with BWk's internal search timer show that my response adds less than one second to the actual processing time. More precisely, the difference between my measurement and the internal timer has a median value of 0.41 seconds with an inter-quartile spread of 0.15 seconds. In practice, this means that my measured times are about a half-second longer than the actual time, and are accurate to within two-tenths of a second. In other words, differences of less than 0.2 seconds are not meaningful.

A different method must be used in each package to define when the search is over. Here are the events at which I pressed F11 to stop timing.

- BWk posts the number of verses, hits, and the total time.
- BWn closes the search window and opens a window listing the results.
- Lib produces the word "Finished" and the number of hits in the tool bar of the search window.
- OLB opens a window listing the results.
- ESw turns the cursor from an hourglass back to a pointer.
- Swo has an activity bar that grows across the bottom of the search window until it reaches the right-hand end.

A number of factors might affect search speed. These include

- the length of search region (it might take longer to search the entire Bible than just to search a single book);
- the number of verses that actually satisfy the query (there might be a processing overhead associated with each successful match);
- the number of terms in the query (it might take longer to look for more words than for fewer);
- the presence of wild cards (which might take more time to match with the text).

To explore these factors, I timed each package's performance on a set of queries. I included some queries on the KJV so as to include the packages without detailed support for Hebrew and Greek. After each query, I list and discuss the results, and then draw some general conclusions. All times are reported in seconds.

A single package may produce different results on the same search at different times. For example, if one searches the KJV for the word “worship” twice in a row on BWk, the first search may take half a second, but the second less than a tenth of a second, measured by the program’s internal timer. This speed-up appears to be due to disk buffering in Windows. When a program asks for data from the hard drive, Windows first stores it in high-speed RAM memory. If a subsequent operation calls for the same data before it has been replaced with other data, access is much faster the second time around. For another example, I have clocked the same search in Lib at 19.7 seconds one time but 146.0 seconds (!) another time. This slow-down appears to be due to the overload of data structures internal to Lib, an overload that sometimes causes the program to crash when many intensive searches are run in quick succession. I take several steps to help neutralize such effects. I run only one program at a time. I start it fresh, then run through the tests once, recording the times. Then I exit it and start the next program. All tests are done with the same (minimal) set of other programs running in Windows. With these precautions, these times are replicable to within about 20%.

The search functions in all programs produce a list of verses that satisfy the query, together with a preview of the text in the verse. Lib offers an option to turn this preview off. I report both sets of times for Lib; those with preview off are much more stable than those with preview on.

A Rare Word.—The word “chapmen” (meaning “merchants,” strictly camp followers catering to soldiers) appears only once in the KJV, at 2 Chr 9:14. Table 6 shows the time each program needed to find this word, first within the entire Bible, then restricting the search to 2 Chronicles.

Table 6: “chapmen” in KJV. Notes: (a) BWn cannot restrict KJV search to a subset of books.

| | Verses | BWk | BWn | Lib (Preview) | Lib (No Preview) | eSw | OLB | Swo |
|---------|--------|-----|-----|---------------|------------------|-----|-----|-----|
| Gen-Rev | 1 | .5 | .8 | 4.3 | 5.3 | 5.5 | .8 | 5.8 |
| 2 Chron | 1 | .3 | (a) | 2.3 | 3.5 | 0.6 | .4 | 1.1 |

All programs take longer to search a longer portion of text.

A Common Word.—“Jesus” appears in 935 verses in the Bible, all in the NT. Table 7 shows the times for the whole Bible and for the NT.

Table 7: “Jesus” in KJV. Notes: (a) BWn cannot restrict KJV search to a subset of books. (b) After observing the much longer search time in the NT, I repeated this search, and recorded a time of 146 seconds! (c) eSw finds 940 verses; I have not collated the two sets of results to understand the difference.

| | Verses | BWk | BWn | Lib (Preview) | Lib (No Preview) | eSw | OLB | Swo |
|----------|---------|-----|-----|---------------|------------------|-----|-----|-----|
| Gen-Rev | 935 (c) | .7 | 1.4 | 19.7 (b) | 24.4 | 3.1 | 1.7 | 5.3 |
| Matt-Rev | 935 | .4 | (a) | 125.6 | 26.8 | 1.6 | 1.0 | 4.7 |

Again, most programs (this time except Lib) take longer to search a longer portion of text. Comparing Table 6 with Table 7 shows that in most cases search time increases dramatically with the number of verses retrieved. This increase is important because many searches consist of

successive refinement. I generally start with a simple query that retrieves too many cases. As I review the retrieved cases, I recognize features that are common to instances that are not of interest to me, and then I revise the query to exclude them. This process is useful only if queries are so quick that I don't feel distracted by the time it takes them to complete.

Number of words in Query.—Six verses in the NT mention “Troas,” and six contain all four of the words “Lord”, “Saviour”, “Jesus”, and “Christ” (in any order). Thus searches for these two enable us to compare how much of a difference multiple search terms make, without confounding any effect due to number of hits or size of the search region. Table 8 shows the resulting times.

Table 8: Number of Search Terms. Notes: (a) BWn cannot restrict KJV search to a subset of books, so these times are for the entire Bible. (b) eSw uses the American spelling “savior” in its KJV, and this spelling was used in this test.

| | Verses | BWk | BWn | Lib (Preview) | Lib (No Preview) | eSw | OLB | Swo |
|--|--------|-----|-----|---------------|------------------|---------|-----|-----|
| Troas | 6 | 1.0 | .5 | 6.6 | 5.7 | 1.0 | .5 | 3.6 |
| Lord + Saviour + Jesus + Christ | 6 | 0.8 | .8 | 6.0 | 8.5 | 1.2 (b) | .8 | 3.8 |

With the exception of BWk and Lib in preview mode, all programs showed an increase for the longer query. The differences for BWk, eSw, and Swo are within the timing error.

Wild Cards and Strong's Searches.—The tests in Table 9 show performance on searches involving wild cards and Strong's numbers. The asterisk “*” is commonly used in queries to match any string of non-blank characters. Thus “right*” matches “right”, “rights”, “righteous”, “righteousness”, “righteously”, and so forth. In the absence of wild cards, most packages assume that search terms must be bounded by white space or punctuation, but BWn and Swo do not make this assumption, so their searches are effectively wild card searches unless the users inserts special delimiters.

Table 9: Wild Cards and Strong's. Notes: (a) BWn does not support Strong's numbers.

| | Verses | BWk | BWn | Lib (Preview) | Lib (No Preview) | eSw | OLB | Swo |
|------------------|--------|-----|-----|---------------|------------------|-----|-----|-----|
| right* | 834 | 1.1 | .9 | 135.3 | 29.0 | 3.2 | 1.8 | 5.8 |
| child* + 5207 | 51 | .6 | (a) | 15.0 | 8.1 | 2.7 | 0.6 | 5.2 |

These values are most appropriately compared with searches that do not include these complications. The search for “right*” (which occurs in 834 verses) can reasonably be compared with the search for “Jesus” in the whole Bible (with 935 verses retrieved) in Table 7. These results suggest that little penalty is paid for using wild cards or Strong's numbers in searches.

Hebrew Searches.—I timed three Hebrew searches with varying levels of complexity. The first is for the phrase **המקום הזה**. The second is for **המקום** without a following **הזה**. The third is for the Qal passive participle **ברוך** followed within three words (20 characters for Lib) by a Qal imperfect of the verb **היה**.

Table 10: Hebrew Searches. Notes: (a) Lib finds only three verses, because it does not recognize jussives as imperfects.

| | Verses | BWk | BWn | Lib (Preview) | Lib (No Preview) |
|---------------------------------------|--------|-----|-----|---------------|------------------|
| המקום הזה | 40 | .4 | 2.4 | 3.3 | 3.8 |
| הזה without המקום | 75 | .6 | 1.3 | 14.3 | 4.7 |
| Construction with ברוך and היה | 4 (a) | 0.9 | 1.3 | 5.0 | 10.6 |

These times are comparable for those in English versions. Excluding **הזה** causes a modest increase in search time for BWk and a greater increase for Lib, but actually results in a decrease for BWn. The search with morphological specifications is the longest search of the entire test suite for BWk, but even so BWk's time is shorter than any Hebrew search for the other programs.

Greek Searches.—I timed two Greek searches. First I searched for all optative verbs in the NT. In scanning the list of results, I noticed that many were instances of **εἰμί**, so I did a second search for optatives that were not **εἰμί**. Table 11 shows the results.

Table 11: Greek Searches. Notes: (a) BWn cannot formulate a morphological search that *excludes* a specified lemma. (b) Lib's syntax allows two forms for this search: [*/vo??????] (the asterisk matching any lemma) and [/vo??????] (simply leaving the lemma unspecified). These times are for the second form. The first form leads to extremely long searches. I have never seen a search in this form terminate, and in one instance I waited over 15 minutes for a response before killing the program.

| | Verses | BWk | BWn | Lib (Preview) | Lib (No Preview) |
|---------------------------------------|--------|-----|-----|---------------|------------------|
| Optative verbs | 63 | .6 | 1.9 | 17.7 | 7.5 (b) |
| Optative verbs other than εἰμί | 51 | .5 | (a) | 11.8 | 4.3 |

Excluding a specific lemma actually reduces search time in the two programs that support lemma exclusion.

Summary.—These experiments enable us to compare the various packages in terms of search speed, and also to make some observations about expressiveness. I describe speed in two ways. First, I report the minimum and maximum speeds of each program over the set of problems. Second, I rank the programs for each test, assigning a score ranging from 1 for the fastest to 6 (in English searches) or 3 (in Greek and Hebrew searches) for the slowest. (If two programs have the same speed on a test, I assign them the same rank, and skip the next rank number. Thus if the

times for a given test were (.5, .8, 1, 1, 1.3, 1.6), I would assign ranks (1, 2, 3, 3, 5, 6). If a program cannot run a test, it is assigned the lowest rank for that test. I rank Lib on the basis of its no-preview times, since they are more stable than times with preview turned on.

BWk is clearly the fastest search program. Only one test case exceeded the critical “one-second” barrier that [3] requires for “maximum acceptable response.” Lib is by far the slowest search program of those reviewed, and on four out of thirteen tests it exceeds the “ten second” barrier that is likely to cause users to turn elsewhere. BWn offers reasonable search times; only one English search exceeds one second, and though all of the Hebrew and Greek searches do, its maximum time of 2.4 seconds is shorter than Lib’s shortest search time on any test. Among the free programs, OLB is clearly the fastest.

BWn is not as expressive as BWk and Lib. It cannot exclude a specified lemma from a morphological search, or restrict a KJV search to a subset of books. (But given its speed on English searches, this latter constraint is not serious.) BWk’s Advanced Search Engine is more expressive than Lib’s command line, but because it is the only package to offer this capability, I

Table 12: Summary of Timing. Unless otherwise specified, numbers are ranks. Ranks in parentheses are tests that the program cannot run.

| | Scope | Verses | BWk | BWn | Lib (No Preview) | eSw | OLB | Swo |
|---------------------------------|-------|--------|-----|-----|------------------|-----|-----|-----|
| chapmen | All | 1 | 1 | 2 | 3 | 4 | 2 | 5 |
| chapmen | 2 Chr | 1 | 1 | (6) | 5 | 3 | 2 | 4 |
| Jesus | All | 935 | 1 | 2 | 6 | 4 | 3 | 5 |
| Jesus | NT | 935 | 1 | (6) | 5 | 3 | 2 | 4 |
| Troas | NT | 6 | 2 | 2 | 6 | 4 | 1 | 5 |
| Lord + Saviour + Jesus + Christ | NT | 6 | 1 | 1 | 6 | 4 | 1 | 5 |
| right* | All | 834 | 2 | 1 | 6 | 4 | 3 | 5 |
| child* + 5207 | All | 51 | 1 | | 5 | 3 | 1 | 4 |
| <i>Median Rank</i> | | | 1 | 2 | 5.5 | 4 | 2 | 5 |
| <i>Max time (sec)</i> | | | 1.1 | 1.4 | 29.0 | 5.5 | 1.8 | 5.8 |
| <i>Min time (sec)</i> | | | 0.3 | 0.5 | 3.5 | 0.6 | 0.4 | 1.1 |
| | | | | | | | | |
| המקום הזה | OT | 40 | 1 | 2 | 3 | | | |
| הזה without המקום | OT | 75 | 1 | 2 | 3 | | | |
| Construction with ברוך and היה | OT | 4 | 1 | 2 | 3 | | | |
| Optative verbs | NT | 63 | 1 | 2 | 3 | | | |
| Optative verbs other than εἰμι | NT | 51 | 1 | (3) | 2 | | | |
| <i>Median Rank</i> | | | 1 | 2 | 3 | | | |
| <i>Max time</i> | | | 0.9 | 2.4 | 10.6 | | | |
| <i>Min time</i> | | | 0.4 | 1.3 | 3.8 | | | |

did not include a test case to evaluate its timing.

Usually, a Bible student will want one or more programs running constantly, but sometimes one may wish to start up a program to answer a short question. The programs vary widely in how long they take to launch. Lib is the slowest, ranging from 82 seconds with only the home page open to over 200 seconds if the user had several resources open when the program was closed. BWn is the fastest, at 17 seconds. BWk takes 35 seconds, eSw takes 39, OLB takes 38, and Swo takes 28 seconds.

5 Collateral Resources

The student of the Bible works in two directions: inward (studying the biblical text itself) and outward (exploring what others have written about the Bible). To varying degrees, all of the tools under review provide some access to such resources, but they vary widely in the types of resources they provide and the amount of material that is available in the formats required for each platform. In this section, I first review the kinds of resources available and then survey what each package makes available.

5.1 *Kinds of Resources*

Resources other than Bible texts may be grouped into lexicons, grammars, Bible dictionaries, commentaries, and special-purpose tools.

Lexicons.—Tools that support Hebrew and Greek often integrate these lexicons so that clicking on a word in the text brings up the lexicon entry for that word. Examples of lexicons that are available electronically include (in Hebrew) the Strong's definitions, HALOT, TWOT, and BDB (both with and without verse references), and (in Greek) Strong's definitions, BDAG, TDNT, Louw-Nida, Thayer, Friberg, various editions of Liddell-Scott, and the UBS Greek dictionary.

Grammars.—Relatively few grammatical works are currently available, and only in Lib.

Bible Dictionaries.—Resources such as Eastons, ISBE, the Anchor Bible Dictionary are often linked to words in English versions of the Bible.

Commentaries.—A common practice in several packages is to synchronize a commentary with the text, so that as the user moves through one, the other scrolls to keep up with it. Commentaries are perhaps the most widely available digital resources, including classics such as Calvin, Barnes, Clarke, Keil and Delitzsch, and Leopold on Genesis, as well as newer commentaries such as the Word series.

Special-Purpose.—Some packages have electronic reference aids that do not correspond to conventional print resources, such as timelines or conversion tools for weights and measures.

In general, if a package supports one of these resources, Biblical references in the resource are enabled as hypertext anchors so that clicking on them takes one to the referenced passage. Much rarer is the ability to search through all of the available resources for a given Bible reference; only Lib supports this important function.

A frustrating aspect of collateral resources is that they are not portable from one tool to another. For example, Leopold on Genesis is available only on OLB. Keil and Delitzsch is available for free in eSw, or for \$120 from Lib. Calvin's commentaries are included on the OLB CD (though not available through the website, because of bandwidth limitations), but costs \$97 in Lib format.

One cannot use either of these resources with any package other than the one for which it is configured. This state of affairs is at the least frustrating for people who would like to focus on a single tool. Thus I can't read my free Keil and Delitzsch on Lib. Even worse, it means that having purchased an expensive resource (say, HALOT or BDAG) for one resource (such as BWk), one cannot use it in another (say, Lib), but must repeat the investment. The result is that many users may keep several tools open at the same time to have access to the desired resources. This situation is the electronic analogy to the piles of books that used to accumulate on my desk. At least it's easier to <Alt>-<Tab> among them than it was to dig Gesenius out of the bottom of the stack.

5.2 Tool-Specific Comments

Space does not permit a complete catalog of the resources available in each package. Instead, I discuss the kinds of resources that each tool supports, give some indication of the range available, and describe the kinds of support for using these resources.

Lib is without question the dominant program for accessing collateral resources. Its collection in all categories is by far the largest (over 3000 volumes), but by no means exhaustive. It lacks some older items that are available in other packages, such as the commentaries of Barnes and Gill, and of Leopold on Genesis. Many prominent publishers have adopted the Lib format for electronic versions of their works, so many recent works (e.g., the new ISBE, the Anchor Bible Dictionary, the Word Commentary series, the New American Commentary series) are available only for Lib. The preparation of individual works is excellent. Biblical references are all linked to the text. Lib's more elaborate search language enables such functions as searching the entire library for references to a particular biblical passage, a capability that is invaluable when dealing with a reference work such as a grammar or Bullinger's *Figures of Speech* that is not organized in biblical order. No other tool offers this capability. It is also the only package to offer any grammatical references. Lib includes a special tool that converts automatically among weights and measures in both modern and ancient units.

While Lib offers a wide range of collateral resources, **BWk** offers comparatively few. The program comes with a range of lexicons (TWOT and BDB for Hebrew, Friberg, UBS, Louw-Nida, Thayer, the intermediate Liddell-Scott for Greek), Bible dictionaries (ISBE, Easton, Nave's), and offers BDAG and HALOT for an additional fee, but otherwise does not appear to support new add-on modules. The only commentaries available (included with the program) are Robertson's Word Pictures in the New Testament and the Treasury of Scripture Knowledge. BWk's implementation of its collateral resources is mixed. Biblical references in BDAG and HALOT are linked to the text, but those in other resources are not. BWk includes two special tools, both editable by the user. A biblical timeline displays the relative chronological position of different events, and a synopsis tool permits the construction and display of synoptic passages (such as a harmony of the gospels).

One benefit of BWk's selective approach to reference tools is that it can optimize the presentation format for an individual tool, rather than following the same format for all tools. The Louw-Nida lexicon is an example. In Lib, it is presented like any other book, with a linear structure and a hierarchical table of contents. Figure 6 shows BWk's presentation of this tool. The windows (counterclockwise from upper left) provide (1) an alphabetical list of all available words, (2) pointers to the different semantic groups under which the word is referenced, (3) the definition for a word in one of these groups, (4) a list of other word families in the same semantic

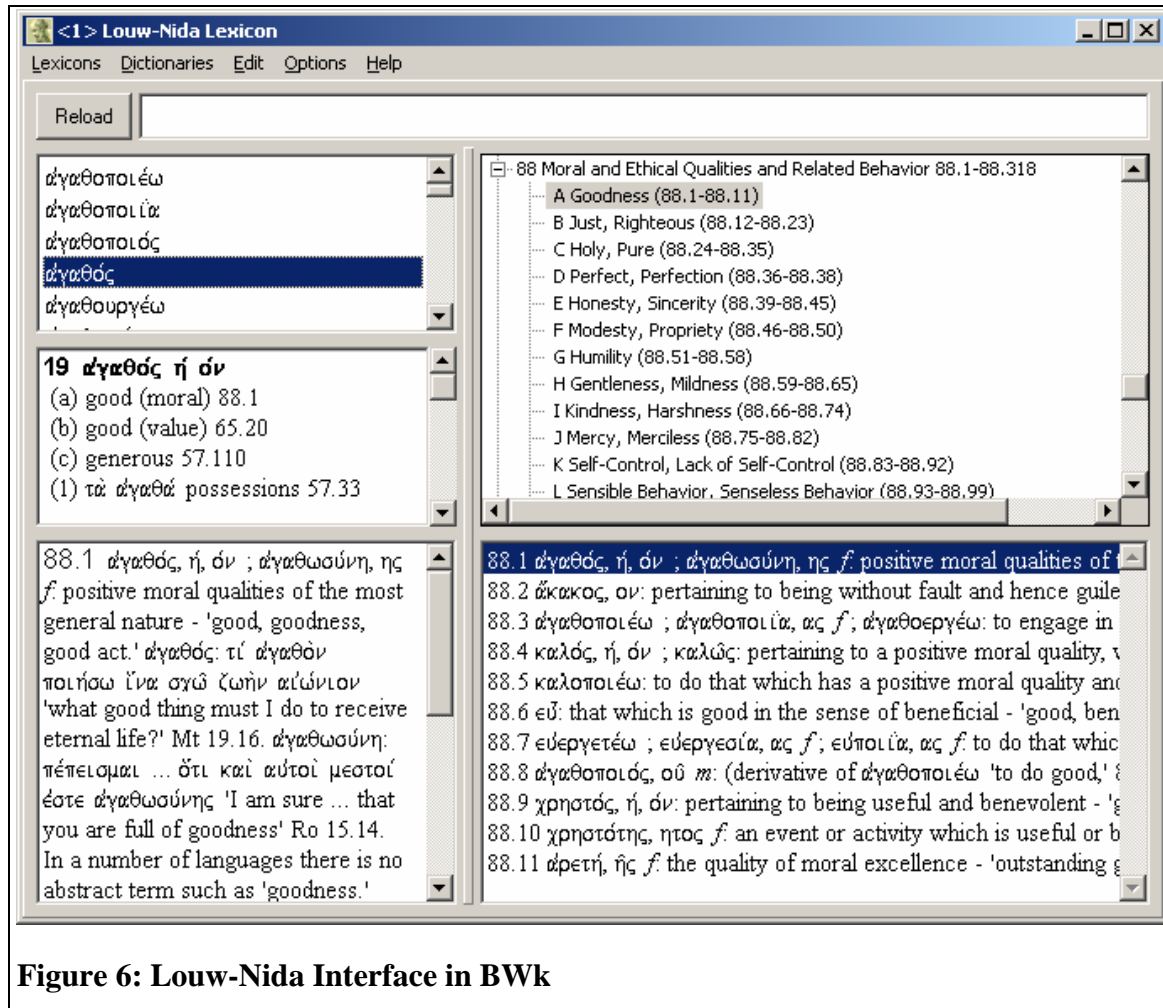


Figure 6: Louw-Nida Interface in BWk

group, and (5) a hierarchical display of the semantic categories. The definition appearing in (3) is defined by selecting a group from (2) and a word family from (4). This layout is much better suited than a linear presentation for comparing synonyms with one another. (It would be even more useful if the Bible references in window (3) were cross-linked to the text.)

In itself, **BWn** provides even fewer collateral resources than does BWk. Its Greek lexicons are Friberg, UBS, intermediate Liddell-Scott, and Louw-Nida, and its Hebrew lexicon is the abridged version of BDB with no internal references. It does not link biblical references to the text. However, in another sense BWn is the most open package. It offers two kinds of links to resources that it does not provide itself. First, it can link directly to Lib to access resources that are indexed either by lemma (such as a lexicon) or by reference (like a commentary). Thus it offers Lib users a relatively powerful and very fast search capability along with integrated access to their other resources that Lib supports so well. Second, it supports two net-based resources. One is the NET Bible (<http://www.bible.org/netbible/>), a new translation of the Bible with extensive notes by the translators that amount to a fairly technical commentary. The other is the Perseus project (<http://www.perseus.tufts.edu>), with direct links to the full Liddell-Scott lexicon and Greek morphological word analysis. In addition, the Perseus web site contains a rich selection of resources for the study of the classical world. BWn's philosophy is thus one of focusing on its core competence (fast grammatical searches) within an architecture that gives the user access to supplementary tools provided by others.

Unfortunately, some of the built-in links that BWn uses to access external resources have changed since the program was released. For example, an attempt to access the NET Bible from a specific passage leads to an HTTP 404 (page not found) error; the program tries to find (for example) <http://www.bible.org/netbible/mar.htm#1> for Mark 1, but the appropriate URL has been changed to <http://www.bible.org/netbible/mar1.htm>. This sort of address change is a perennial problem with web-based resources. It would be nice if BWn redirected web references through a configuration file that could be updated easily, and even nicer if it allowed users to add their own links for their favorite on-line Bible study sites.

The three free packages (OLB, eSw, Swo) all offer a range of supplementary material, including Bible dictionaries, commentaries, and devotional literature. All three support Strong's numbers, and clicking on a word brings up an expanded Strong's entry for the word in question. All three have special windows that show the entries in a selected Bible dictionary for the word over which the cursor is resting, and also provide for linked access to various commentaries on the passage under study. Users who purchase one of the more sophisticated packages may still want to install one or more of the free packages, for access to the inexpensive references that they provide, either for free or for the distribution fee of the CD, including Leopold on Genesis (OLB), Broadus on Matthew (OLB), Calvin (OLB, only on CD), Barnes (OLB, eSw, Swo), Gill (eSw, OLB), Clarke (eSw, Swo), Keil-Delitzsch (eSw), Luther on Galatians (Swo), Josephus (OLB, eSw, Swo), Ramsay on Paul (eSw), and many others. Bible references in the resources offered by these packages are uniformly cross-linked to the biblical text. eSw provides a reader for books formatted in the STEP format, an open format for digital texts that unfortunately is not widely used.

Even when a package provides a way to link from a word in the text to a dictionary, it may be convenient to collect dictionary references for of the words in a passage. Such a listing is one of the options in the BWk Report Generator. Lib's Exegetical Guide provides the same functionality. (I must protest the claim of Lib's Help file that this report "exegetes a passage from a Hebrew or Greek Bible." Exegesis is far more than just listing the parsings and dictionary definitions of each word in a passage.)

6 User-Originated Material

Bible study is an ongoing conversation involving the Scriptures and those who study them. Most Bible students do not simply listen to this conversation, but product material that could contribute to it, such as study notes, sermons, journal or magazine articles, commentaries, new Bible translations, etc. Ideally, the products of one's studies should be able to be integrated back into the Bible study package, so that it grows with the user (and with a broader community with whom the user may be involved). All packages except for BWn provide some support for such extension.

All packages except BWn permit users to take **notes on a passage** of the Bible and display these notes whenever that passage is displayed. Since Bibles in Lib are just one form of digital book, Lib permits users to attach notes to any book, analogous to writing notes in the margin of a paper volume.

Annotating digital resources with the results of one's study is a two-edged sword. On the one hand, these notes are easy to access in subsequent study. On the other hand, the format and location in which these results are stored may hinder their wider use and preservation.

Table 13: User-Originated Material

| | BWk | BWn | Lib | eSw | OLB | Swo |
|------------------------------|-----|-----|-----|-----|-----|-----|
| Notes on a passage | T | | TR | TR | TR | TR |
| Notes on a topic | | | TR | TR | TR | |
| Synopsis of Several Passages | X | | | | | |
| New Translation | X | | | | X | X |
| Arbitrary book | | | | | X | X |

The *format* of these materials is often proprietary to a specific software package, rather than being a generic format such as Word, HTML, or PDF. Thus they cannot be used apart from the package. BWk's note files are in the widely-used RTF format, but have the extension ".BWW" rather than ".RTF", hiding their reusability from the user. Lib permits the user to export any report or notes file as HTML, so that it can be accessed with a web browser.

The *location* in which materials are stored is also important. The default action of most packages is to treat these materials as belonging to its program files, storing them internally to its own directory structure, and notes on different passages are sometimes stored in different files. This approach makes it easy to overlook these materials when backing up one's data files or when reinstalling the Bible program on a new computer. (Lib puts these materials by default in the user's "My Documents" folder, which is much better than storing them in the program directory. BWk defaults to the program directory for notes files, but permits users to redefine the directory.) It would be helpful if package vendors provided an additional level of indirection for user-generated material, treating them as data files whose default location should be specified by the user and then indexed via a directory stored locally by the program.

The other categories of material are usually saved as single files, and are much easier to share with other users.

It is sometimes useful to capture **notes on a topic**, which are not directly tied to a specific passage in a Bible or other book. In several cases, Lib users have created notes files that they make available to other Lib users. Online Bible permits these notes to be organized like a Bible dictionary, so that they can be accessed from a word in another book.

The sophistication of passage notes and topic notes varies from package to package. Some packages only support text in notes (T), while others also support live Scripture references (R) that lead via a mouse click directly to the referenced passage.

BWk's **Synopsis Tool** enables the user to construct synoptic displays that show correspondences across different passages, BWk includes a harmony of the gospels built with this tool; users could construct synopses of (say) Kings and Chronicles, or of Acts and the Pauline epistles.

Some packages allow users to format **new Bible translations** for use with the software, providing access to the various functions that are driven by biblical references, or to create **arbitrary books** in a format that the software can display.

The packages reviewed take a range of positions on the development and distribution of new resources by users. Some packages (notably BWk, OLB and Swo) encourage users to develop and distribute additional materials as they see fit. OLB provides free access to the software necessary to generate new books, and BWk includes a compiler for new texts (e.g., new Bible translations) in the package as distributed. In other cases (notably Lib), the technology for compiling resources is tightly controlled, and users cannot generate and sell new resources without a license from the company.

7 Getting Help

All of these programs are complex pieces of software with many features and options. Even experienced computer users need good resources for learning how to do things.

The first stop is the program's own help system. In most of the packages, this is accessible and well organized. The standard Windows help system provides users with three ways to access help information: a hierarchical table of contents, a precompiled index, and a "find" facility that searches the entire help file for individual words. When the Windows help screen is active, the user can shift between it and the program window, overlaying one with the other without having to close help first. Lib does not use this help facility, but instead presents help documents in a dialog window as a hierarchical table of contents. This display is troublesome in two ways. First, ordinary resource windows cannot overlay the dialog window, so it is difficult to switch back and forth between it and resources. Second, it offers neither an index nor a "find" function, functions that I personally find very useful in getting to know a piece of software. For instance, the table of contents to the help system doesn't mention Strong's numbers, and I would love to have been able to search the help system for "Strong's." In fact, this is possible. The help files are in standard Lib resource format, and can be opened and then searched in a standard resource window. But they do not open in a searchable window when accessed through the "Help" menu bar selection, and it is cumbersome to have to go through the extra steps to search them.

BWk and Lib offer video tutorials that show how to perform standard tasks. These tutorials are included in the purchase price of BWk, but require an extra purchase price from Lib. Only BWk and BWn include a printed manual, fairly succinct for BWn but 400 pages long for BWk.

In most cases, the on-line help was adequate, but users will sometimes need further information. For instance, Lib's help file says nothing about how to search for Strong's numbers in English versions. Lib makes up for the weakness of its on-line help with an excellent support section on its website, including a very active user group newsgroup. The discussions on this newsgroup are quite well organized, and it is relatively easy to find answers to many questions that are not treated in the on-line help. In addition, the web site offers pointers to a large number of sites maintained by users that have extensive discussions of many aspects of Lib usage. BWk's web page also supports a sortable, searchable newsgroup.

Electronic books, like printed ones, can have typographical errors. An important difference is that it is much easier to correct such an error in a disk file than on the printed page. Lib has a very well integrated mechanism for users to report errors that they find so that they can be corrected in future editions. The user selects the incorrect text, then opens the "Help" menu and

selects "Report a Typo," and the system formulates an email message with the required details about the location and nature of the error. This mechanism is a creative and progressive way to improve the quality of digital resources over time.

8 Conclusion

Bible study software has reached the stage of maturity where every student of the Bible should become acquainted with it. Even the free packages are a tremendous advance over manual concordances and cross-references, while the commercial packages will make the work of scholars in the original languages much more productive. The choice of a package will be guided by the student's budget, study habits, and working environment. Lib is the leader for users whose primary need is access to recently published electronic books. For serious students of the original languages, the robustness, speed, and flexibility of BWk's searching capabilities make it the clear choice, while some exegetes may prefer BWn's simpler but also speedy interface. Currently the packages do not exchange resources, so users with abundant hard disk space may want to install multiple packages. The free packages offer the most economical access to many classic resources, and users who wish the greatest freedom in producing and distributing their own resources will want to encourage the proliferation of these packages. But compared with the situation twenty-five years ago, there is no bad choice. The current range of offerings is a dream come true, and one that every Bible student should exploit.

References

- [1]P. Bickford. Worth the Wait? 1999. Web Page, http://developer.netscape.com/viewsource/bickford_wait.htm.
- [2]B. Friberg and T. Friberg. *Analytical Greek New Testament: Greek Text Analysis*. Grand Rapids, MI, Baker Book House, 1981.
- [3]J. Nielsen. *Designing Web Usability*. Indianapolis, IN, New Riders, 1999.
- [4]H. V. D. Parunak. Prolegomena to Pictorial Concordances. *Computers and the Humanities*, 15:15-36, 1981.
- [5]H. V. D. Parunak. Transitional Techniques in the Bible. *Journal of Biblical Literature*., 102:525-548, 1983.
- [6]H. V. D. Parunak. Dimensions of Discourse Structure: A Multidimensional Analysis of the Components and Transitions of Paul's Epistle to the Galatians. In D. A. Black, Editor, *Linguistics and New Testament Interpretation: Essays on Discourse Analysis*, 207-239. Broadman, Nashville, TN, 1992.
- [7]J. D. Price. *The Syntax of Masoretic Accents in the Hebrew Bible*. Lewiston, NY, Edwin Mellen Press, 1990.