Interpretive Implications of Using Bible-Search Software for New Testament Grammatical Analysis

By Harry Hahne, Ontario Theological Seminary

Presented at the Annual Meeting of the
Evangelical Theological Society, Nov. 24, 1994

In recent years Bible-search software has become readily accessible to the average scholar. Unfortunately, using a computer to search the Bible may lend unwarranted credibility to research. Computer-assisted biblical research is subject to the same errors as traditional research methods and opens up new potential sources of error. In my comparative tests of several popular Bible-search programs (*Gramcord*, *Bible Windows*, *Bible Works* and *TheWord*)[1] the same grammatical searches often produced radically different results.

These variations are due to several factors: (1) differences in the underlying grammatically-tagged biblical texts; (2) differences in the capabilities and assumptions of the search software; and (3) user errors. Researchers who use these tools should be aware of how these potential pitfalls can affect the accuracy of their analysis.

Differences in the Underlying Biblical Texts

Programs which search different biblical texts will obviously produce different search results.

1. Grammatically Tagged New Testament Texts

Most programs suitable for scholarly study of the New Testament use the United Bible Societies 3rd or 4th edition (e.g. *BWin*, *BWorks*, *TheWord*)[2] or Nestle-Aland 26th edition Greek texts (e.g. *Logos*, *Gramcord*, *Accordance*[3]).

However, the grammatical tagging systems used for Greek and Hebrew texts vary significantly. A tagged text attaches parsing, lemmas (dictionary forms) and sometimes word definitions to each Greek and Hebrew word in the biblical text. This allows sophisticated stylistic and grammatical searches that would be impossible with the biblical text alone.

The selection of grammatical tags is based on subtle and often unstated assumptions. Although the function of a Greek word is indicated largely by its spelling (its "morphology"), at times the function of a word must be determined by its relation to the context. There is always a tension between purely morphological analysis based on word forms and a more functional analysis based on the interaction of a word with other words in the sentence. Grammatical tagging schemes range along a spectrum from formal (morphological) to functional classifications. No scheme for classifying Greek words is purely formal or purely functional, since the function of a word is determined both by its morphology and its relation to the context. However, the more a tagging system tends toward the functional end of the spectrum, the more subjective the classifications become.

For example, Bauer's lexicon classifies οὗ as an adverb of place However, in 5 instances (Mt 18:20; Rom 4:15; 5:20; 1 Cor 16:6; 2 Cor 3:17), the Friberg text, which is a largely

[1]For a review of these programs see Harry Hahne, "High-Tech Bible Study: PC Bible Programs with a Graphical User Interface" *Computer-Assisted Research Forum*, 1, no. 3 (Spring/Summer 1993): 7-17 and idem, "PC Bible-Search Software for Scholarly Research" *ARC: The Journal of the Faculty of Religious Studies, McGill University* 22 (1994): 109-125. The following programs were used in researching this paper: *Bible Windows* 2.5 and 3.0, *Bible Works* 2.2.2, *Gramcord* 4.04, *TheWord* 3.05, *Logos* 1.6B. The results for *Logos* are usually not reported, since its grammatical search capabilities are too limited for scholarly research.

[2]*BWin* = *Bible Windows*. If no version number is specified, the statement applies to both version 2 and version 3. *BWorks* = *Bible Works*.

[3]*Accordance* is a sophisticated Macintosh grammatical-search program that uses the same database as the DOS-based *Gramcord* program. It is available from the Gramcord Institute.

functional system, classifies it as a conjunction, based on the nuances of the word in the context. A user would need to be aware of such functional classifications in order to find every occurrence of a particular part of speech.

There are four major morphologically-tagged Greek NT texts commonly used in Bible-search software:[4]

1. **Gramcord text** (used by *Gramcord*, *Accordance*): This text uses primarily formal classifications, though particles and conjunctions include functional subcategories. When a word could be understood in more than one way, both classifications are often included and the search results are flagged to indicate this ambiguity.[5]
2. **Original Friberg text** (used by *BWorks*, *TheWord*): This text is more functionally classified, based on discourse analysis. The text seeks to provide not only morphological information, but also functional classifications of words based on the context of the sentence and the overall discourse.[6] If a researcher does not recognize that the Friberg text includes some functional classifications, the search results could be misleading or incomplete.
3. **Revised Friberg text** (used by *BWin 3*). In late 1994 the Fribergs revised their text to include a larger number of formal classifications. In many instances both formal and functional classifications are included and many words were reclassified. *BWin 3* uses the revised Friberg text in place of the CCAT text which was used in earlier versions.
4. **CCAT Text** (used by *BWin 1, 2* and *LBase*): This text is based on the original Friberg text, but the tags were modified to be more consistent with the Septuagint text produced by this group.[7] Some of Friberg's discourse level analysis was removed, resulting in primarily morphological classifications.

Unfortunately, except for *Gramcord*, the manuals for popular Bible-search programs rarely discuss the assumptions used in the classification of words. Yet it is essential that researchers understand the nature of the underlying machine-readable biblical text if their analysis of the text is to be meaningful.

The print edition of the Friberg text has an appendix outlining the criteria used for the tags.[8] This is helpful when using *BWorks*, which stays very close to the original Friberg text. However, in some instances *TheWord* deviates from the Friberg tags without documenting the changes. *BWorks* and *TheWord* only use one of Friberg's multiple classifications of ambiguous words, without documenting which classification was chosen.

The following chart shows the "family tree" of various texts and the programs which use these texts:

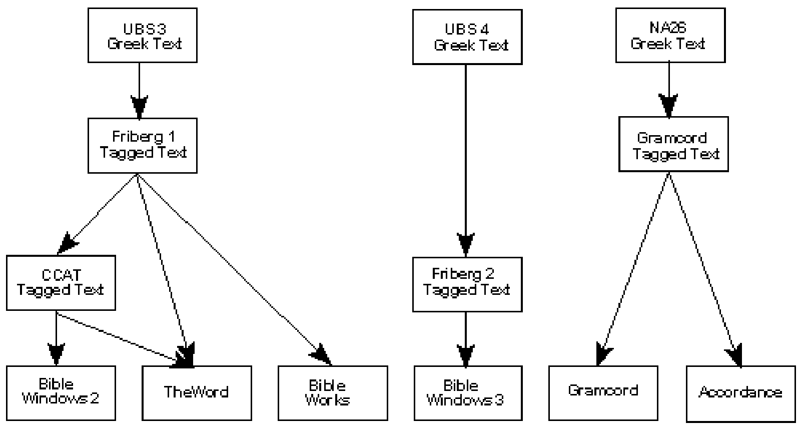
[4]Texts based on a Strong's numbering system, such as *Logos* are not included. Although *Logos* uses largely formal morphological tags, the tags are based on the Textus Receptus, even when the NA26 Greek text is displayed. Thus a search for future participles with *Logos* finds 17 matches, while programs which base their grammatical tags on the UBS3 or NA26 text find 12 matches.

[5]Project Gramcord began in the late 1970's under the direction of Paul A. Miller. The grammatical tags were primarily selected by James L. Boyer, then Chairman of the Department of NT and Greek at Grace Theological Seminary.

[6]The tagged text was originally created by Barbara and Timothy Friberg to provide raw data for Timothy's linguistics dissertation on NT Greek word order.

[7]This text is one of several biblical and classical texts created by the Center for Computer Analysis of Texts (CCAT) at the University of Pennsylvania under the direction of Robert Kraft.

[8]Barbara and Timothy Friberg, eds., *Analytical Greek New Testament* (Grand Rapids: Baker, 1981). The second edition will be printed in 1994.



2. Database Text and Classification Errors

While users assume the accuracy of Bible-search tools, the underlying texts are rarely completely free from error. When the databases are created, the classifications of lemmas (dictionary forms) and grammatical forms are often performed initially by an automatic parsing program. Sometimes the human proofreaders may fail to catch errors.

Errors can fall into three classes: (1) errors in the basic biblical text; (2) incorrect lemmas attached to words in the text; and (3) incorrect grammatical classifications.

Errors in the biblical texts are rare, since the texts often derive from machine-readable texts used for typesetting the print editions. However, incorrect lemmas and grammatical tags are more common. For example, in Jn 15:26, the Friberg 1 text claims ὅν comes from εἰμί. This error appears in both *Bwin 2* and *BWorks*, but it is fixed in *TheWord*. *BWin 3*, which uses the Friberg 2 text, parses it correctly. James K. Tauber has collected a list of hundreds of errors in the CCAT tags, mostly in the lemmas. He suggests that many of the errors appear to be due to automatic parsing and lemmatization.[9]

3. Functional and Unusual Classifications

In many tagged texts, there are some functional or unusual classifications of words which can produce unexpected search results.

In *Gramcord*, many foreign words such as ὡσαννα are classified as interjections. However, foreign proper nouns are classified as nouns and parsed by function in context. By contrast, *BWin* and *BWorks* classify ὡσαννα as a particle.

Conjunctions and particles are particularly difficult words to classify. A beginning user might miss many occurrences of καί if he only searches for the word as a conjunction. Since καί also functions as an adverb in some cases, most programs will sometimes classify it as an adverb. However, as the following chart shows, the classification choices in individual instances vary considerably:

Program:	**Conjunction:**			**Adverb:**
	No subclass:	**Copulative:**	**Correlative:**	
Gramcord		8049 words	187 words	656 words
BWorks	8214 words 4727 verses			801 words 733 verses
Bwin	750 words (?)			750 words (?)
TheWord	5126 verses			753 verses

[9]James K. Tauber (jtauber@tartarus.uwa.edu.au) is a student of Linguistics at the University of Western Australia. He can supply a patch file which corrects many CCAT text errors. Unfortunately, it will not work with the versions of the database used by *BWin*, *BWorks* or *TheWord*.

Bwin was unable to report the total number of occurrences of καί, because it only allows 750 matches in a search. Since it is hard to predict how a program will classify the word in any given passage, the safest approach is to search for all possible classifications and manually eliminate invalid matches. The *Gramcord* manual documents how many times each word is classified as a conjunction, particle or adverb, which makes it easier to define searches that will find all occurrences of such words.

Since the Friberg text (*BWorks* and *TheWord*) attempts to classify many words by function based on discourse analysis, some classifications may be surprising to users. Friberg 1 uses the category of "substantive adjective" to refer to adjectives which are used as nouns in context. For example, ἀγαθός ("good") is classified as a substantive adjective in Mt 5:45 ("he makes the sun shine on the evil and the *good*). This type of classification affects 4131 occurrences of 1068 words in 3009 verses![10] While adjectives can certainly function as substantives, the term "substantive adjective" is not a part of speech used by most Greek grammars. It would be easy for a user to accidentally miss many important occurrences of adjectives unless he searches both for "adjectives" and "substantive adjectives". The Friberg 2 text eliminates the substantive adjective classification, but it introduces other surprising functional classifications. For example, in most cases Friberg 2 classifies relative pronouns as adjectives, with an adjective subtype of "relative."

Functional classifications such as those frequently used in Friberg's text are more subjective than formal classifications. Their value depends largely on the accuracy of the classifier's interpretation of the text. While they appear to be objective raw data, in fact they contain the prior conclusions of another researcher, which tends to skew the search results to fit the classifier's own viewpoint.

4. Treatment of Classification Ambiguities

Even the strictest formal classification method must classify certain words by function in context, since the morphology of these words is inconclusive.

While in most cases the meaning is clear in the context, in some instances the grammatical classification is subject to scholarly debate. For example, the gender of πονηροῦ could be either neuter or masculine. In Mt 6:13 the meaning is debated: Does the Lord's Prayer ask for deliverance from "evil" (neuter) or "the evil one" (masculine)? Since *BWin 2*, *Gramcord* and *Accordance* classify πονηροῦ in Mt 6:13 as neuter, a search for masculine adjectives will not find the verse. By contrast, *TheWord* and *BWorks* classify the word as masculine and do not allow the word to be found in a search for masculine adjectives! Only *BWin 3* acknowledges both possible parsings and allows the word to be found with either search.

Bible-search programs would be more useful if they marked such words as ambiguous and allowed searching on the multiple classifications. The print version of the Friberg text includes multiple classifications in many instances. However, at this time only *BWin 3* allows searching on Friberg's multiple classifications. Although *BWorks* and *TheWord* both remove the multiple parsings in Friberg 1, the documentation does not explain the criteria used to make these choices.

Gramcord makes a good attempt at handling ambiguous classifications. In many cases, it tags words in multiple ways and flags the ambiguous classification in the resulting concordance. The documentation lists all ambiguous classifications which are used.[11] However, even Gramcord could be improved in this area. For example, it does not include the ambiguous classification of πονηροῦ in Mt 6:13.

[10]According to the statistics reported by *BWorks*.

[11]*Gramcord Users Guide*, pp. 77-80.

Differing Capabilities of Search Software

While on the surface most Bible-search programs appear to allow similar searches, there is considerable variation in the search capabilities of programs. Many programs lack the sophistication to perform finely tuned searches. Further, some of the hidden or poorly documented assumptions in the search engines can produce surprising results.

1. Use of Wildcards

Wildcards are symbols that indicate that any letter or letters will be accepted at a certain point in a word. Thus a search for "απο*" will find ἀπολύω, ἀποδίδωμι and other words which begin with "απο".

Some programs place a limit on the number of words that can match wildcards. *TheWord* allows a maximum of 300 words to match a wildcard and may not warn if this limit is exceeded. *Logos* only allows 32 words to match a wildcard, which is too restrictive for many useful searches.

Most programs (e.g. *TheWord*, *Gramcord*, *BWorks*, *Logos*) assume that the search expression includes the whole word, unless wildcards are explicitly included. However *BWin* uses full word searches for grammatical searches and double wildcard searches for word and phrase searches. (In a double wildcard search, the search letters can be found anywhere within a word.) This inconsistent behavior in *BWin* can easily confuse users and result in erroneous searches.

2. Limits on the Number of Matches

BWin has an undocumented limit of 750 matches per search. Since there is no error message that warns that the maximum number of matches has been exceeded, this can lead to misleading conclusions.

3. Different Ways of Reporting Statistics

TheWord reports matches in terms of the number of verses which contain the desired construction. *BWin* and *Gramcord* report the number of occurrences of the desired construction. *BWorks* reports both the number of occurrences and verses.

4. Word Order Sensitivity.

For some searches, word order is very important. For example, a search for substantival adjectives should find all occurrences of an article in agreement with an adjective only when the article appears just prior to the adjective, not after the adjective. In other cases, it is important to find all permutations of word order. For example, a search for genitive absolutes should allow either the genitive noun or the participle to appear first.

Programs differ in the importance they place on word order in search expressions. *Gramcord* requires an exact match of the order of the elements in the search definition. However, searches can be defined that find several combinations of word order in one pass and distinguish them in the resulting concordance. *BWorks* and *TheWord* do not distinguish the word order of search elements. This can result in many false matches. For example, a search for "μὲν . . . δὲ" finds 10 verses in which the order of words is "δὲ . . . μὲν". *BWin* is sensitive to word order in grammatical searches but not in word searches, which can produce inconsistent search results.

5. Duplication of Search Terms

Many grammatical constructions require that the same search term appear more than once (e.g. "δὲ . . . δὲ"). *Gramcord*, *BWin* and *TheWord* allow the same term to appear more than once. They properly find verses in which the word δὲ occurs twice. However, *BWorks* simply finds all verses in which the word δὲ occurs at least once.

6. Exclude Intervening Terms

False matches can frequently be eliminated by specifying terms that must *not* appear between search elements. For example, a future perfect periphrastic construction requires a future tense of εἰμι and the perfect participle of another verb in the same clause. Since it is highly unlikely that a finite verb will occur between these two search terms, the search can be improved if it specifies that no finite verb can intervene.

Gramcord and *Accordance* allow multiple intervening exclusion and inclusion terms. They can specify words and parts of speech that *may* occur between search terms as well as words and parts of speech that *may not* occur between search terms. Other programs do not include a true "exclude intervening term" option. At first glance it would appear that the "and not" Boolean operator which is available in *BWin* and *TheWord* would accomplish the same thing. However, the "and not" operator defines what a *search term* may not be, not the types of words that cannot appear between search terms. Thus this feature can produce undesired interactions between the search terms.

The following chart illustrates the effect of excluding intervening terms in a search for future perfect periphrastics:

Source:	Matches:	Invalid:	Missing:
Nigel Turner, *Syntax*, p. 89[12]	6		
BWorks	13[13]	9	0
Gramcord			
Not exclude intervening verbs	12	6	0
Exclude intervening finite verbs	8	2	0
BWin			
Not exclude intervening verbs	9	3	0
Use "and not" indicative verbs	0	0	6

The search with *BWorks* has a large number of invalid matches since it has not allow excluding intervening terms. When *Gramcord* searches without excluding intervening terms, it also finds a large number of invalid matches, though not as many as *BWorks*, since it limits the context to a sentence. When *Gramcord* is set to exclude intervening finite verbs (indicative, subjunctive, optative, imperative), all but two of the false matches are eliminated. On a search with a larger number of results this could save considerable time. Since *BWin* has no true exclusion command, the second search term is set to "and not an indicative verb". As a result, there are no matches, because any verse with a future of εἰμί also has a finite verb (i.e. εἰμί).

7. Proximity of Multiple Search Terms

Many grammatical constructions require that two or more words be in close proximity, though not necessarily side by side. A search program should allow restriction of search expressions to a definable number of words.

Gramcord allows the user to specify that up to 200 words span from beginning to end of a construction and *BWin* allows specifying up to 20 words. By default both programs assume that all elements in a search expression are juxtaposed. If this number is not set appropriately many valid examples of a construction will be missed.

BWorks and *Logos* have less flexibility than either of these programs. By default, all words must appear somewhere in the same verse. The user can specify a maximum number

[12]James Hope Moulton, *A Grammar of New Testament Greek*, vol. 3, *Syntax*, by Nigel Turner (Edinburgh: T & T Clark, 1963), p. 89.

[13]Since *BWorks* reports the number of matching verses, not constructions, it does not indicate that more than one matching construction sometimes occurs in the same verse.

of *verses* in which to find the search terms. This is far less valuable for grammatical searches than a limit by number of words, though it can have value for discourse-level research.

8. Search Boundaries for Multiple Search Terms

For grammatical searches it is more valuable to set the search boundaries by sentences or clauses than by verses. A program based on verse boundaries would have difficulty with sentences that span several verses (e.g. Eph. 1:3-14). For discourse analysis, search boundaries should be set at the paragraph, chapter or book level. An ideal program would allow setting search boundaries by clause, sentence, a specific number of verses, paragraph, chapter, book. It would also allow the option of stopping at or ignoring various types of punctuation marks.

Accordance allows boundaries to be clause, sentence, verse, paragraph, chapter or book. *TheWord* allows boundaries to be verse, paragraph, chapter or book. Most other programs are more restricted. *BWin* does not allow specifying boundaries, though it will cross verse boundaries if the word proximity is set high enough. *BWorks* and *Logos* use verses as boundaries, but search expressions can cross verse boundaries if the proximity is set to 2 verses. *Gramcord* uses the sentence as a boundary, so it is more likely than a verse-oriented program to find all occurrences of a grammatical construction.

Programs differ in how they handle a conflict between the number of words in proximity and the search boundary. *BWin* will cross verse boundaries in an effort to compare the specified number of words. *Gramcord* will never cross a full stop (period, semi-colon (raised dot) or question mark), regardless of the maximum number of words allowed in the proximity. This subtle difference can produce significantly different search results. *Gramcord* misses 12 examples of "μὲν . . . δὲ" constructions that *BWin* finds because it ends the search at a full stop. In most of these cases, the punctuation is a semi-colon, which indicates that the two clauses are closely related. The choice of a semi-colon rather than a comma is a debatable editorial decision in each instance.

9. Forced Agreement of Grammatical Features

Many grammatical constructions require either that certain grammatical features agree or not agree. For example, a genitive absolute requires a clause with a genitive noun and a genitive participle that agree in gender and number. If agreement cannot be required between search elements, many false matches must be manually removed.

Since *BWorks* and *Logos* do not allow specifying agreement of grammatical features, these programs make it difficult to find genitive absolutes without substantial manual labor. *BWin* allows specifying agreement, but has no way of limiting the agreement to specific search terms. Thus a search for genitive absolutes is quite simple with *Bwin*, but it is difficult to find constructions in which individual pairs of search terms agree. For example, it would be difficult to find the common Greek construction "article1 article2 noun2 noun1", where *article1* and *noun1* agree with each other and *article2* and *noun2* are genitive and agree with each other (e.g. ὁ τοῦ τέκτονος υἱός in Mt 13:55). *Gramcord* and *Accordance* allow great flexibility in agreement. Any grammatical feature of selected pairs of words can be required or forbidden to agree. As a result, these programs can find very complex grammatical constructions with relatively few false matches.

10. Sensitivity to Diacritical Marks

TheWord requires that Greek accents and breathing marks be entered as they appear in the biblical text. This makes entry of search expressions tedious and error prone. It also results in missed matches, where the context changes the accents.

On the other hand, required entry of breathing marks is desirable, since otherwise it is difficult to distinguish similar word pairs such as εἰς and εἷς. *BWin* includes breathing marks in the word pick list for grammatical searches and gives the option of including them in word searches. If they are not included in word searches, breathing marks are ignored. With *Gramcord* ambiguities can be resolved by specifying the desired part of speech. *BWorks* has no way to easily distinguish οὗ (an adverb of place) from οὐ (the negative particle), since it classifies both as adverbs.

11. Comparative Results in a Typical Search

A search for "μὲν . . . δὲ" makes a useful case study that illustrates several of these differences in search capabilities of several programs. While the search is simple, it reveals an astonishing variation in search results:

Source:	Matches:	Invalid:	Duplicate:	Missing:
Nigel Turner, *Syntax*, p. 332	110			
Gramcord				
Not exclude intervening δὲ	112	0	14	12
Exclude intervening δὲ	98	0	0	12
BWorks				
Context of 1 verse	104	11	0	17
Context of 2 verses	157	47	0	0
BWin				
Word search mode	134	27	0	3
Grammatical search mode	97	0	0	13
TheWord				
Context of 1 verse	97	0	0	13

The different search results of these programs are due to the following factors:

1. *Gramcord*: This program misses 12 references that cross a full stop boundary (11 semi-colons and 1 period). *Gramcord* also includes 14 duplicate references in verses that have multiple occurrences of δὲ. If the search is performed with an intervening δὲ excluded, these duplicates are eliminated.
2. *BWorks*: This program includes 11 invalid citations that are due to its inability to distinguish word order. Some occurrences are missing because the default search boundary is one verse. If the search context is set to 2 verses, many more invalid citations are found, although more valid citations are also picked up.
3. *BWin*: In word search mode, *BWin* finds many invalid references due to insensitivity to word order. In grammatical search mode, *BWin* finds several references missed by *BWorks*, since it allows expressions to cross a verse boundary, as long as the specified number of words has not been exceeded.
4. *TheWord*: This program misses some occurrences that extend over more than one verse. It does not find as many invalid citations as *BWorks* because it is sensitive to word order.

In most cases these programs missed citations when the valid reference crossed a verse, sentence or semi-colon boundary. The programs picked up invalid citations due to the inability to distinguish word order or multiple occurrences of one of the search terms in a sentence.

12. Summary of Search Capabilities

The following table summarizes the search capabilities of several programs:

Feature:	Gramcord:	BWorks:	BWin:		TheWord
			Gram. Search	Word Search	
Wildcards:	optional	optional	none	implicit	optional
Match limit:	unlimited	unlimited	750	750	unlimited
Statistics reported:	occurrences	occurrences and verses	occurrences	occurrences	verses
Word order sensitivity:	Yes	Yes	Yes	No	No

Feature:	Gramcord:	BWorks:	BWin:		TheWord
			Gram. Search	Word Search	
Allow duplicate terms:	Yes	No	Yes	No	Yes
Exclude intervening terms:	Yes	No	No	No	No
Proximity: Type:	Words	Verses	Words	Words	Verse, paragraph, chapter, book
Limit:	200	Unlimited	20	20	1
Boundary: Type:	Full stop (period, question mark, semi-colon) or weak stop (comma, colon)	Specified number of verses	None	None	Verse, para-graph, chapter, book
Priority:	Boundary	Proximity	Proximity	Proximity	Proximity
Agreement of grammatical features:	Any combination of search terms	None	All search terms	None	None
Diacritical marks:	Ignored	Ignored	Required	Optional	Required

Common User Errors

Even the most powerful search software and carefully tagged database can produce misleading results if the user does not use the software correctly. There are several common sources of error when using Bible search software.

1. Lack of Understanding of the Program

A user may not understand the assumptions used in tagging the text or the limitations of the search engine. For example, if a user does not realize that word searches are insensitive to word order in *BWin* word searches, he may be surprised to discover the number of false matches in the "μὲν . . . δὲ" search. Some subtle search assumptions are documented, yet easily misunderstand. For example, since *BWorks* sets search boundaries by number of verses, one might assume that a context of "1" would mean the search terms must appear within a single verse. In fact, the number refers to the number of verses after the verse in which the first search term is found. Thus a search context of "0" means the words in the search expression must be found within one verse.

Unfortunately, many important search assumptions and database tagging guidelines are not documented for most programs. For this reason, it is wise to experiment with a search that has known results before attempting searches with unknown results.

2. Lack of Understanding of the Grammatical Construction

The user may not understand the grammatical construction well enough to formulate a search accurately. When searching for future perfect periphrastics, it is not enough to find future forms of εἰμί within a certain number of words of a perfect participle. The two words must function together as a single grammatical unit in the sentence. Restricting the search to a single clause and excluding intervening finite verbs will help reduce the number of false matches.

It is wise to read about a construction in conventional grammar books before attempting a computerized search. This will enable the search expression to be formulated more precisely and provide some guidelines for the reasonableness of the search results.

3. Failure to Search for all Permutations of the Construction

It is easy to assume erroneously that a single search has found all examples of a construction. A thorough search must consider all valid orders of the search terms. For example, a search for genitive absolutes must look for constructions with the genitive noun first as well as constructions with the genitive participle first. *Gramcord* allows a single search to include many different search expressions and flags the resulting concordance according to the construction category. This reduces the need for multiple searches.

A thorough search must also find constructions with functionally equivalent parts of speech. For example, many constructions which call for a noun would be valid with a substantival participle or substantival adjective.

4. Failure to Manually Eliminate False Matches

Even the best software will sometimes produce false matches which must be manually eliminated. Programs such as *Gramcord* which can exclude intervening terms produce fewer false matches than programs which cannot. For example, in a search for future perfect periphrastics, Gramcord reports only 8 matches instead of 12 matches if the search excludes intervening finite verbs. However, Lk 1:45 and 6:40 can only be eliminated by hand, since the fact that the participle functions as a substantive is only indicated by context.

Conclusions

These considerations lead to the several suggestions for users of Bible-search software:

1. Understand the tagging system used in your program. If functional tags are used, search for all possible classifications of the desired terms. Generally formal tags with widely used grammatical terms are easier-to-use and less error prone than functional tags. Functional tags have value in discourse analysis but tend to skew the search results to favor the assumptions and interpretations of the person who tagged the text.
2. Learn the assumptions and limitations of the search software, such as word order sensitivity and maximum number of matches. Consider how such limits could affect your search results.
3. Experiment with searches that have known results before you try searches with unknown results.
4. Be cautious about claiming to have found all examples of a construction until you have examined thoroughly all possible arrangements of the construction. Limitations of your search software may prevent you from finding some valid occurrences of the construction.

An awareness of the abilities and limitations of Bible-search software will help researchers to utilize Bible-search software more wisely and reduce the likelihood of erroneous conclusions.