Enabling the Production of High-Quality English Glosses of Every Word in the Hebrew Bible

Drayton Benner
President | Miklal Software Solutions
PhD Candidate | Northwest Semitic Philology | University of Chicago
DraytonBenner@MiklalSoftware.com
BibleTech Presentation | March 16, 2013

Introduction

Good afternoon. I'm delighted to be with you this afternoon to talk about a software tool designed to enable the production of high-quality glosses of every word in the Hebrew Bible. The glosses that are produced using this software tool will be used in a print Hebrew-English interlinear Bible.

Let me show an image from another Hebrew-English interlinear so that if any of you have never seen an interlinear, you can get an idea as to what their constituent parts are and how it can be used. This is an image from *The Interlinear NIV Hebrew-English Old Testament*.

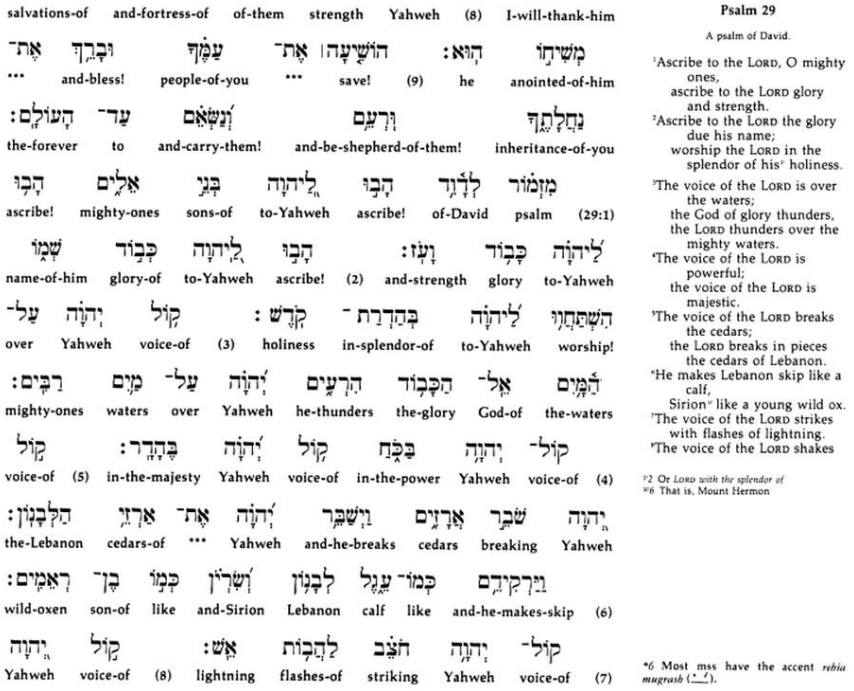

Figure 1: Sample print interlinear page, The Interlinear NIV Hebrew-English Old Testament

You can see that the Hebrew is on top with English glosses underneath it. In the margin is the text of the English translation. And there are also footnotes for the translation as well as an occasional note on the Hebrew. Easily the most time-consuming task in a work like this is producing the glosses for each Hebrew or Aramaic word of the Bible. The glosses need to be literal yet contextual, and they need to be friendly to the marginal translation.

Interlinears like this are wonderful. Those who read Hebrew at a beginning or intermediate level can grow in confidence as they read the Hebrew text by using an interlinear to read significant swaths of text. In my case, I sometimes still like to use an interlinear if I want to read ten or twenty chapters of Hebrew as rapidly as possible to get a sense of the whole.

The print book for which I wrote the software tool is entitled *Hebrew-English Interlinear Old Testament with BHS Text*. This print product is not yet on sale and has not been announced publicly. [1] The publisher has graciously allowed me to present on it here today, but if anyone is live-blogging or tweeting, you're welcome to write generally about the software tool, but please respect the kindness of the publisher and do not write anything that would indicate what translation is being used or which publisher is involved.

[1] Now available, as of November 2013. Cf. http://www.amazon.com/gp/product/1433501139/.

With that caveat out of the way, the software tool is entitled *The Hebrew ESV Print Interlinear Enabler*, a mouthful I'll shorten to The Enabler from here on, so you can surmise that the ESV will be the marginal translation, and Crossway is the gracious publisher in question.

Structure

With introductory matters out of the way, I will first discuss the requirements for this software tool, then show you each of its parts, then gloss a sample pericope, then discuss a bit of how I produced algorithmic glosses before giving some results and concluding remarks.

Requirements

The Enabler needs to:

- Enable the user to produce literal yet contextual glosses that are friendly to the ESV translation for each Hebrew or Aramaic word in the Hebrew Bible.
- Enable the user to do his glossing at a high level of quality and in a consistent manner. Crossway is very concerned that the quality of the glosses be quite high.
- Enable the user to gloss quickly.

How to satisfy these requirements

The software tool needs to provide the user with lots of data relevant to producing a gloss in a visually compact manner so that most of the time, the user can gloss a Hebrew word without jumping from screen to screen or from resource to resource.

The software tool needs to provide ways for the human glosser to quickly dig deeper to get more data when necessary.

The software tool needs to provide ways in which the user can check his work for consistency, both while he glosses and also after the fact. The Hebrew Bible is long—over 400,000 words. Well, over 400,000 once we separate out the conjunction waw, the article, and inseparable prepositions. This is too much data for anyone to keep in their head.

The software tool needs to make the human glosser's job easier by algorithmically glossing each word in such a manner that it gets it correct most of the time, allowing the human glosser to focus his efforts on the more challenging and less tedious cases.

My place in this process

Here I should add a note that I was not involved in this project from the start. Crossway had their glosser, whose name is Thom Blair, start with another software tool but, halfway through the glossing efforts, commissioned the software tool I'm about to show you to help better achieve their goals of quality, consistency, and speed.

Introduce the Enabler

Without further ado, let me show you the Enabler; let's start with the main pane, the Interlinear Text Editor, and look at the sources of data that it presents.

Figure 2: The Interlinear Text Editor

Components of the Interlinear Text Editor

Hebrew text, lexeme, and morphology

The top three rows present the Westminster Hebrew text together with its lexical and morphological information. A project that is well-known to many of you, the *Westminster Hebrew Morphology* is now maintained by the *J. Alan Groves Center for Advanced Biblical Research*, headed by Kirk Lowery. It is a faithful electronic representation of *Codex Leningradensis*.

The version of the text and morphology used here is 4.2. The project was locked into using version 4.2 before I came to be involved in it. You will occasionally see the Hebrew text with an orange background. That signals that there was a change to the text or, much more commonly, the lexical or morphological analysis, between version 4.2 and version 4.14, the most recent version when I became involved in the project. Double-clicking on those brings up the version 4.14 data.

Lexham contextual glosses

The contextual glosses in the next row come from the *Lexham Hebrew-English Interlinear Bible*, a project headed by Christo Van der Merwe and owned, I believe, by Logos.

CBHAG non-contextual glosses

The non-contextual glosses in the next row come from *The Comprehensive Biblical Hebrew and Aramaic Glossary*, produced by Chip Hardy, a professor of Old Testament at Louisiana College and also a fellow PhD candidate with me in Northwest Semitic Philology at the University of Chicago. Among the sets of non-contextual glosses I have examined for Biblical Hebrew, it is the highest quality set. These glosses are keyed to the Westminster morphology, both on the lexical level and at the level of verbal stems.

ESV English-Hebrew Reverse Interlinear

The ESV text comes from the *ESV English-Hebrew Reverse Interlinear*, which Logos produced. When there are multiple English words corresponding to the Hebrew lexeme, generally only the heart of those words is being presented, as determined by those who produced this reverse interlinear.

User's previous glosses for that lexeme

The next to last row addresses the desire for consistency. It shows up to six ways in which the user has glossed this particular Hebrew lexeme in the past, moving from the most frequent way to the least frequent way. However, both because we want the display to be compact and also because we want to

group like glosses together, we are not always showing the precise gloss. Rather, we are performing natural language processing on the user's glosses to shorten them and group them together on the basis of those shortened versions. Thus, if the user glossed a lexeme "he is going" one time and "we went" another time, we are going to produce the single word "go" for each of them and group them together.

User's gloss/algorithmic gloss

The final row is for the user's gloss for this Hebrew lexeme. This is what will appear in the final print product underneath the Hebrew. However, we want to make the user's job of glossing as easy as possible and as fast as possible, and we can do that by making a gloss algorithmically. When the gloss being reported is algorithmic, the background is yellow so that the user knows it.

Achieving perfection with the algorithmic glosses is obviously impossible. What I was aiming for was to do the best I could without taking too long to develop it, balancing the cost of my development time over against the cost of the time saved for the glosser.

I'll talk later about the data sources I used and the algorithms I employed to produce these algorithmic glosses, but for now, let's continue to move forward in seeing how to use this software.

Using the Enabler

Introduction

Keeping with the theme of speeding up the user, we want to allow the user to change the algorithmic glosses as quickly as possible. As a result, we want the user to be able to use the keyboard and the mouse effectively.

Keyboard

The user can move from gloss to gloss using the tab key. The entire gloss, however, is not highlighted for replacement. There are some parts of many glosses that are determined by the morphology and the user's conventions. As a result, they will always or at least nearly always be correct. We do not want to highlight those parts of the algorithmic gloss so that the user does not need to bother with them.

Mouse

The user can also make use of the mouse to make quick changes. If the user wants the gloss to be a single word and sees it in the column, then he can just double-click on it.

Notes

There are times when the user also wants to include a note in the final product. These are generally when the Hebrew and the ESV don't match very well due to a text-critical issue. So, the software provides a way to mark notes. Thom is actually looking at every entry in the apparatus of BHS in order to make sure he doesn't miss a place where a note ought to be inserted so as to alert the user to a text-critical issue that affects the way in which the ESV has translated a verse.

Detailed Lexeme Information Table

In the main window, we have presented a great deal of information to the user in a visually compact manner so that he can pick a gloss for the vast majority of words without looking beyond that. However, there are occasions in which that is insufficient. He will often want to see more detailed information concerning how he has glossed a word in the past.

He can do that by looking at the Detailed Lexeme Information Table, which can be accessed quickly by double-clicking on a Hebrew lexeme, which, by the way, also copies the lexeme to the clipboard so that the user can quickly look up the entry in a Hebrew lexicon in his Bible software.

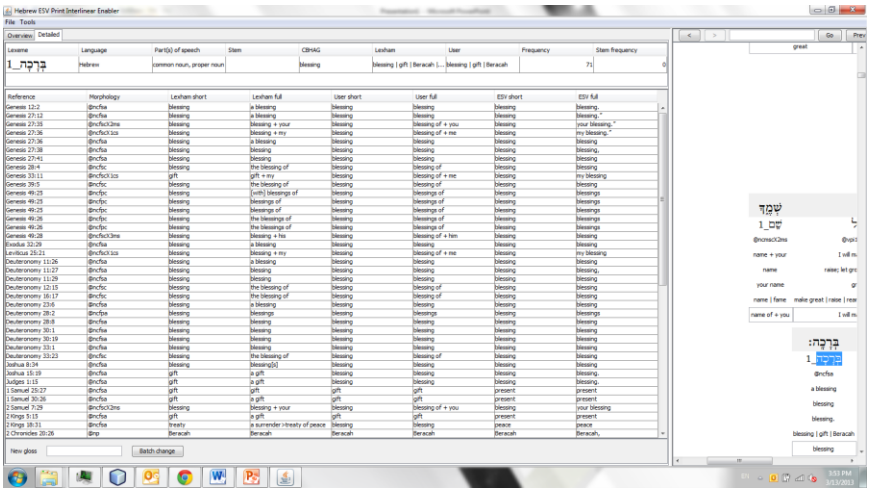

Figure 3: The Detailed Lexeme Information Table

At the top, there is information about the lexeme as a whole: part or parts of speech, non-contextual glosses, shortened contextual glosses from Lexham, shortened user glosses, and frequency information.

Then, most of the space is taken up giving information about each occurrence of the lexeme. The reference is given along with the morphology codes for that occurrence, and then there are six more columns from three sources: Lexham contextual glosses, user glosses, and ESV translations. For each of these sources, the full text is shown in one column, and a shortened version is shown in another column by using natural language processing.

This table can be sorted by any column or any combination of columns so that the user can examine how he has glossed a lexeme in the past as well as how that relates, say, to how the ESV has glossed it. If he wants to jump to a particular verse in the main display, he can just double-click on a row.

The Detailed Lexeme Information Table is invaluable in helping the user maintain his consistency. In fact, there may be times that he wants to change how he has been glossing a word. He can highlight the rows he wants to change and change them in one batch.

Lexeme Information Table

There is one more major display available to the user called the Lexeme Information Table. This pane has basic information about all of the lexemes, both Hebrew and Aramaic.

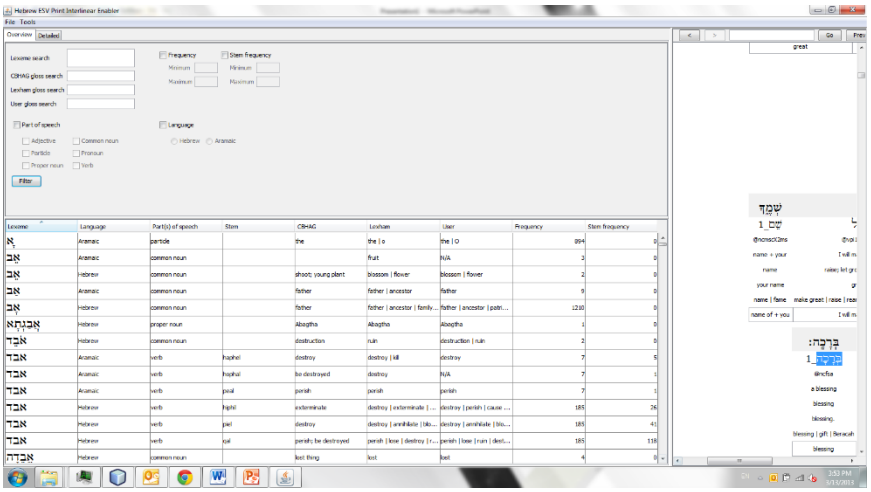

Figure 4: The Lexeme Information Table

Like the Detailed Lexeme Information Table, the Lexeme Information Table can be sorted by any column or combination of columns, and columns can be moved around. It can also be filtered by any or all of the columns up at the top. For example, I'll filter for any Hebrew verbs that the user has glossed as "destroy." There are lots of them, though probably not as many as if we were glossing Akkadian royal inscriptions.

Other tools

Finally, there are a few tools built into the Enabler for tracking progress and ensuring consistency and quality.

Spell checker exporter

The most important of these is the spell checker exporter. Even the most careful user is going to make spelling errors. I considered adding a spell-checker to the Enabler, but I decided that it really wasn't going to be cost-effective. Instead, what I did is set up a little tool for enabling spell-checking. Every once in a while, the user can check his spelling for a set of books or chapters.

For example, let's pick Genesis and Exodus and click "Copy to clipboard." It asks me if I want to update the Progress Tracker, which tracks not only which chapters have been completely glossed but also which chapters have been spell-checked. What it has done is copied all of the user's glosses for Genesis and Exodus to the clipboard, along with references and the like so that the user knows where the glosses are coming from. Now we can use Word's spell checker to look for any mistakes in spelling. On Thom's computer, we imported into Word's dictionary every word that appears in the ESV text so that he would not waste time having proper nouns and the like be flagged by Word's spell-checker.

Sample passage

Introduction

With this tour of the Enabler behind us, let me actually pick a passage to go through together so that we can see it in action. I won't be as careful as Thom is in his glossing, since going that carefully requires moving too slowly for us today, but I think you will get the idea.

In the data that I have loaded, almost all of the user's glosses for the Bible have been done. it is just the end of the Bible, following the English order, that doesn't already have glosses done for it, so let's have a look at a pericope I picked from near the end of the Bible: Zechariah 6:9-15.

As Thom works, he has not only the Enabler in front of him but also other resources at hand: Bible software on a second monitor and print resources like BHS, the ESV, commentaries, etc. close at hand. For this presentation, I only have one monitor available to me, so I'll make the Enabler a bit smaller than ideal and will at least have the ESV text up on the screen as well. In addition, I'll have other Bible software open in case we want to access something in one of them. I have Logos, BibleWorks, and Olive Tree all up and running.

Zechariah 6:9-15

Let's start out just by reading the text in the Hebrew and the ESV, one verse at a time, so that we get a feel for the passage. For the sake of variety, I'll bring up the ESV in Logos and the Hebrew in Olive Tree.

[Read Zechariah 6:9-15 in Hebrew and ESV, one verse at a time.]

[Gloss Zechariah 6:9-15.]

Algorithmic glossing

Introduction

Having gone through a representative passage, you can see that the algorithmic glosser doesn't get it right every time, but it does get it right most of the time. I'll give some statistics later, but how does it do it? What sort of data did we use, and what sort of algorithms did we employ?

Natural language processing tools used

Let me first mention some data I used in addition to writing plenty of my own code.

WordNet

I used a database called WordNet. WordNet is a bit of a cross between a dictionary and a thesaurus with a splash of something else as well. It is useful for a variety of purposes. It performs stemming, that is, moving from the surface form of an English word to its dictionary form, giving all possible stems. It also gives information about related words, both in terms of semantics and in terms of etymology. Finally, it provides information about the frequency of different senses of words. These were all useful to me.

So, when Lexham has "it has been told," or the user inputs the gloss "he was telling" corresponding to some Hebrew verb, I make use of WordNet's resources to be able to shorten these simply to the lexical form, "tell." And I could see connections between words so that if the user glossed "do quickly," and a CBHAG entry had "hasten," I could move through etymological connections and synonym connections to see—at least in theory, I'm making up this example—that "quickly" was related to "hasten" and is a necessary part of the verb.

CMU Pronouncing Dictionary

I also used the CMU Pronouncing Dictionary. This dictionary, produced at Carnegie Mellon University, contains transcriptions of over 125,000 North American English words in IPA. It includes stress information as well.

This is useful in activities like declining verbs. Suppose you have a verb *visit* or *admit*, and you want to produce it in the past tense. The rules for how to change the base of the verb is not always only dependent on the orthography. Why do we spell visited with only one *t* but admitted with two *t*s? Both have two syllables and end in consonant-vowel-*t*. What's the difference? I'm not bold enough to take a poll here to see who knows the rule, but I suspect that the non-native English speakers are more likely to know it than those of us who are native English speakers. The difference is the stress. For verbs that end in consonant-

vowel-*t*, the *t* is doubled when adding *–ed* or *–ing* if and only if the stress is on the last syllable. Thus, *admit* doubles the *t*, while *visit* does not. This rule is not just applicable to *t*, by the way, but also *b*, *d*, *g*, *m*, *n*, *p*, *r*, and *z*. Thankfully the CMU Pronouncing Dictionary comes to our aid in marking stress for both of these verbs so that I can spell the inflected forms correctly.

A bunch of lists

Finally, I am also using a bunch of lists of things like irregular plurals.

Give a taste of the algorithms used

There isn't time to go into all the details of the algorithms I developed for algorithmic glossing, but let me give you a little overview. The process varies a bit based on the part of speech.

Proper nouns

Let's consider the simplest of all parts of speech, proper nouns. The Hebrew word for *David* is going to be glossed as *David* every time. So, we can look at how the user has glossed this Hebrew proper noun in the past, and if he has been consistent, we simply use that word. Pretty simple, right?

However, there are some proper nouns for which there is variation. Sometimes proper nouns can refer to multiple people or places, and in those cases they might be glossed differently. So, in the case in which the user is not consistent in how he has glossed this Hebrew proper noun in the past, we develop a set of possibilities as to how to gloss it.

The possibilities include every way in which the user has glossed it in the past as well as how the ESV translates it here and how Lexham glosses it here if these are different from how the user has glossed it in the past. All of these possibilities start out with a score of zero. We give some points to the ESV's translation and the Lexham gloss.

We then go through each previous occurrence of the lexeme that the user has already glossed, and we give some points to that gloss. How many points we give depends on how well the Lexham and ESV in that verse match the Lexham and ESV in this verse. If the ESV and Lexham were both the same there as they are here, then that is a good indication that the user will probably want to gloss this occurrence of the lexeme in the same manner as he did in that previous occurrence of the lexeme, so many points are awarded to how he glossed it that time. If neither the ESV nor the Lexham in that occurrence match their values here, then we award few points to how the user glossed it that time. If just one of them matched, then we award an intermediate number of points.

Once we have finished going through all the previous occurrences of this Hebrew proper noun, we have a score for each of our possibilities, and we pick that one as our algorithmic gloss. An important feature of this algorithm is that it continues to get better as the user continues to gloss. It learns as time goes on.

Common nouns

The process for common nouns is similar but more complex in two ways. First, sometimes the ESV translation or Lexham gloss might not be a single word in its lexical form. We need to shorten them to a single word, typically a noun, by doing some natural language processing.

The second way in which common nouns are more complex than proper nouns is that we sometimes need more than just the lexical form of the noun. Sometimes we need to make it plural. Sometimes we need to indicate that it is in construct with the following noun or has a pronominal suffix. Thankfully these tasks are not too hard given that this information is contained within the Hebrew morphology. There are rules for making plurals for most English nouns, and I supplemented these with a list of common irregular

plural nouns. Moreover, Thom has consistent standards for how to show that a noun is in construct and how to show a pronominal suffix on a noun.

Verbs

We'll skip adjectives and particles for the sake of time and jump to a brief description of verbs. Verbs are by far the hardest to gloss algorithmically. I spent more time working on glossing them algorithmically than any other part of speech, yet they are the ones that are most frequently in need of revision by the human glosser.

The process is similar to the process for nouns, but there are a couple additional challenges. First, we need to divide verbs up by stem. The bulk of our scoring needs to come from previous occurrences of the verb in the same stem.

Second, while it is completely mechanical to provide the subject of the verb and the pronominal suffix in the gloss based on the morphology codes, determining the tense of the English verb is much more challenging even after we have picked what verb to use. For infinitive constructs and infinitive absolutes, Thom consistently glosses them with English infinitives, so they are not too difficult. Thom always uses an *–ing* form of the verb for participles, though there can be more to the gloss than just putting the verb into its *–ing* form. But for perfects, imperfects, waw-consecutive perfects, and waw-consecutive imperfects, we pick a bunch of possibilities for the tense. These are drawn from how the user has glossed this verb before, the tense used by the ESV here, the tense used by the Lexham gloss here, and knowledge of the English tenses the user typically uses to gloss Hebrew verbs with the particular aspect of this Hebrew verb. We go through allotting points to each of our possibilities and then at the end pick the one that has the most points.

This is a challenge on multiple levels. It is a challenge to parse some English phrases and pick out the main verb and the tense. We need to be able to read "we will have gone" and pick out the subject "we," the auxiliary verbs "will" and "have," and recognize "gone" as signaling that the main verb is "go." It is also a challenge to be able to produce these tenses with any English verb. We might decide that we need to produce "you will have jumped" even though we have not seen this precise combination of words anywhere, so we can't just copy it from somewhere. It is also a challenge to recognize occasions when the Hebrew verb is only expressed in English by a verb with an object or adverb. Sometimes the user will gloss a Hebrew verb with something like "we will do quickly." We have to be able to figure out that we cannot shorten this in any way that excludes the verb "do" or the adverb "quickly;" they will only work as a gloss together.

I was able to produce algorithms to overcome these challenges—not with 100% accuracy by any means, but I was able to get it right more often than not. I suspect that with more effort, I could continue to raise the accuracy level, and it would probably be a lot of fun to do so, but I don't think it would have been cost-effective.

Results

Well, what have been the results? The Enabler does appear to have allowed Thom to produce higher quality glosses with more consistency, though I don't have a way of quantifying it. He has also gone back and changed over 500 glosses from the work he had previously done without the Enabler.

He has also gone faster as a result of the Enabler, glossing 58% more verses per hour than he was before using the Enabler. This is the case despite the fact that before the Enabler he was dealing almost

exclusively with prose, indeed most of the easiest Hebrew in the Bible, and with the Enabler he has primarily been doing poetry and prophecy, the more difficult Hebrew.

And how about the algorithmic glosses? How often have they matched what Thom wants? Well, first let me show a graph of how well we'd do simply picking the Lexham glosses or simply picking the ESV text for each word. I have the results broken down by part of speech, with the totals on the right.

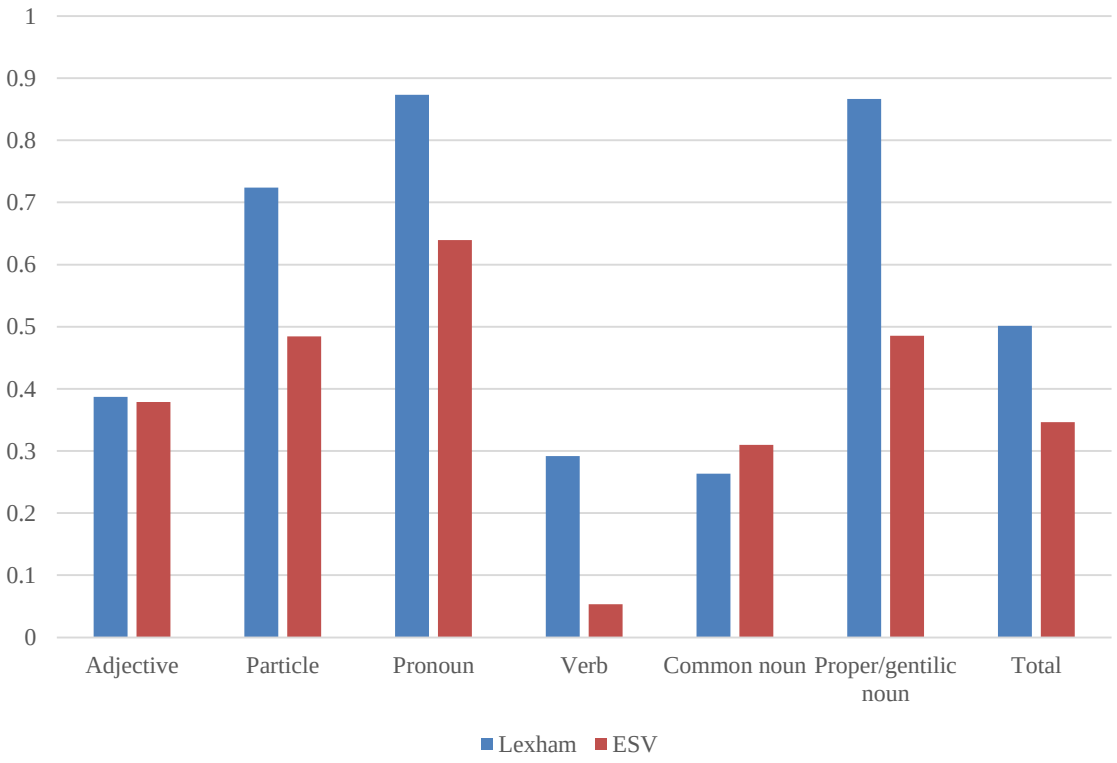

Figure 5: User glosses matching data sources in Job 25-Ezekiel 48 (using Enabler)

These start in middle of Job where Thom was when he started using the Enabler and go through Ezekiel, the last completed book as of the last time I grabbed the data from him.

You can see that picking Lexham would be better than picking the ESV, with the exception of common nouns.

This isn't too surprising: both the Lexham glosses and the user's glosses are supposed to be literal yet contextual, but this is a little unfair to the ESV, since I had to make a choice as to whether to take all of the ESV words matched from the Hebrew word or just what has been identified by those who did that work as the heart of the ESV translation of that Hebrew word. It would probably be wise to vary that decision by part of speech. The number for verbs would probably rise substantially if I had chosen differently.

Overall, just picking the Lexham gloss would do pretty well with pronouns, particles, and proper/gentilic nouns, but it wouldn't do very well with the more substantive words: verbs, common nouns, and adjectives.

Now let me add the algorithmic glosses to the graph.

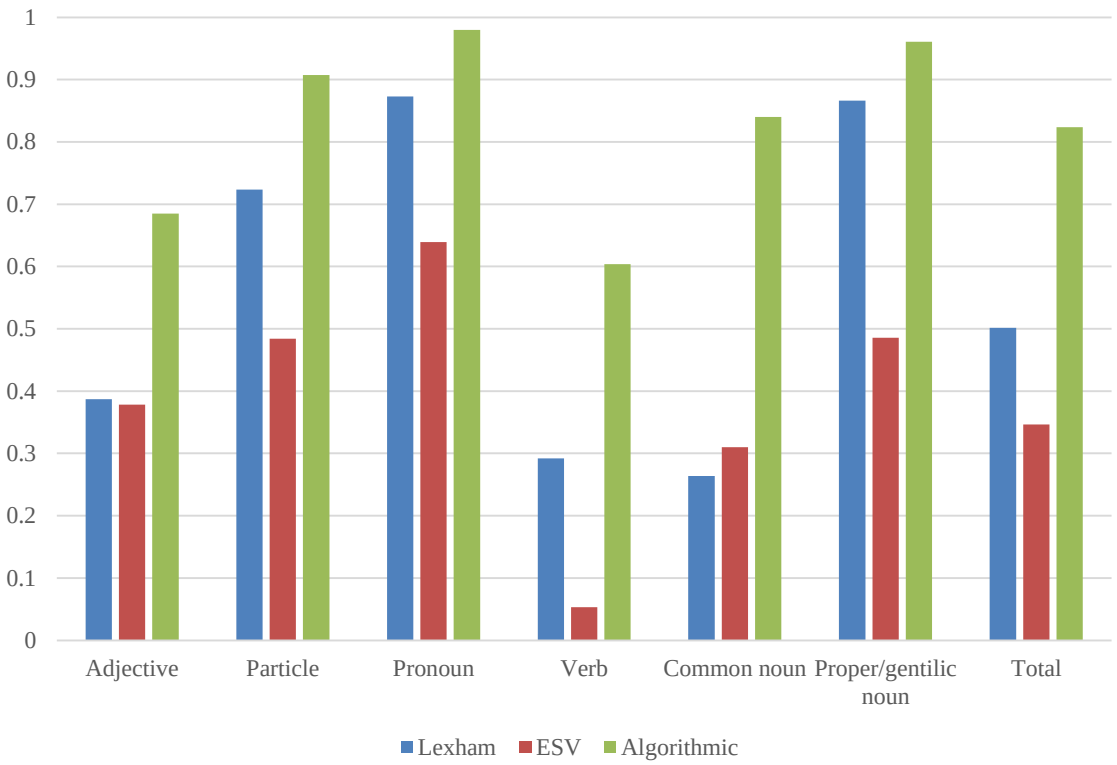

Figure 6: User glosses matching data sources in Job 25-Ezekiel 48 (using Enabler)

You can see that it does better with every part of speech, with the biggest improvement made in common nouns: going from 26% for Lexham to 84% for the algorithmic glosses.

In the total column, you can see that the user approves the algorithmic gloss with no changes 82% of the time. Going into the project, I was fairly confident that I could hit 80% but didn't think I'd hit 90%, so it is within the range of what I was expecting.

Unsurprisingly, verbs were the toughest; I only got 60% of them, but I'm satisfied with that. The only part of speech that disappointed me was adjectives at 68%, so I looked back at that one. The main issue was that I missed one of Thom's glossing conventions. Adjectives are marked for plurality in Hebrew, whereas they are not in English. When Thom glosses a Hebrew plural adjective with a singular English word, he marks it with an asterisk to signal that to the reader. Had I correctly done that, the percentage of matches for adjectives would have gone up to 79%.

What I would do differently next time

Despite being pleased with the results of the Enabler, I don't think it's perfect. So, let me list a few things that I think I would probably do differently next time if I were to do this project over.

Faster load time for a chapter

The first change I would make is GUI-related. Almost everything the Enabler does is very fast. The exception is loading a new chapter, which often takes several seconds, depending on how long the chapter is. Most of the time is actually not taken by the algorithmic glosser, which is fairly snappy, but rather simply laying out the screen. There are some parts of the layout wherein I used standard user interface

tools, but if I had coded them myself with the specific requirements I had, it could have loaded much faster.

Stanford Natural Language Processing Tools

Second, I would make use of an additional set of tools for performing natural language processing on English. Stanford's Natural Language Processing Group provides a set of tools for performing a variety of natural language processing, and I think it would have helped for me to use these on the ESV as a whole, and it might have helped if I had used them on the little phrases that make up Lexham's glosses and the user's glosses.

More use of Hebrew context in producing algorithmic glosses

Finally, in producing algorithmic glosses, I am mostly considering a Hebrew lexeme in isolation from its surrounding lexemes, simply taking advantage of the fact that the ESV translators and the Lexham glossers considered context. I use the Hebrew context in a handful of frequent cases, but I think that my algorithmic glosser would do better if I were to make more use of the Hebrew context, whether that be through the use of a Hebrew syntax or simply through considering n-grams.

Conclusions

In conclusion, there are several factors that together made me enjoy this project very much.

- It involved Hebrew and Aramaic.
- It involved developing complex algorithms to solve difficult tasks.
- It is being used to produce a product that will allow many people to access the Old Testament in its original languages to a greater extent than they would otherwise be able to do. I hope that it will be a blessing to many.