appendix:

The algorithm for *trop* requires the identification of the complete set of second, third and fourth level separators, labeled S2, S3 and S4; the algorithm refers to these three sets repeatedly. The algorithm operates separately on each initial segment.[1]

Associated with the algorithm is a push-down stack, operating on a Last in – First out (LiFo) basis. The algorithm operates on one segment X at a time, beginning either the first or second part of **any** *pasuk*. X is placed (pushed onto) on the stack with Sc set to S2.

If stack = NULL, STOP.

Pop the stack - i.e., operate on the top stack member, a sentence fragment and its associated set Sc; once popped, that stack member is deleted from the stack. If (c+1) is greater than or equal to 5, discard the segment, (placing it back to its position within the original segment.)

Search X as Hebrew is read right to left, until the first symbol in set Sc is found.

If none is found discard X, (placing it back to its position within the original segment) and return to Step 1.

If such a symbol is found, divide X into two segments, X1 and X2, the first segment, X1, from the beginning of X ending with the word on which the symbol is found with Sc = S(c+1), and the second segment, X2, set to the remainder of X with Sc unchanged.

Push X1 and Sc onto the stack.

Push X2 and Sc onto the stack.

Return to Step 1. (RECURSION).

Note that we omit further complicating the algorithm by not including the hierarchy of divisions that the algorithm implies depending on how many times a segment was divided, before terminating. This is clarified twice below after the algorithm is illustrated on each of two examples.

[1] Just after my father died, when I first read in Rav Breuer's book about how *trop* operates, I wrote a recursive LISP program to implement the operation. The full algorithm (in English and not LISP) for the operation implied by the *trop* and a protracted illustration follows. However, it should be noted that when a segment is divided into a right and left segment, only the **left segment** maintains the same *mafsik* as the original segment. This repeats itself until the operation completes.

Let us now apply the algorithm to the *pasuk* from *Ki Tavo* used as an example above. Let us first consider the simpler second half of the *pasuk*, which is placed on the stack, with Sc = S2.

וְהָלַכְתָּ֙ אֶל־הַמָּק֔וֹם אֲשֶׁ֤ר יִבְחַר֙ יְהוָ֣ה אֱלֹהֶ֔יךָ לְשַׁכֵּ֥ן שְׁמ֖וֹ שָׁם

The algorithm moves to step 5), finding the *zakeif katon* on הַמָּק֔וֹם. This is the major division of the second half of the *pasuk*. The algorithm then creates two segments from after the *esnaḥtah* up until the word הַמָּק֔וֹם and the remainder of the *pasuk* after the word הַמָּק֔וֹם. The former segment has Sc = S3; for the latter segment Sc is still = S2. The former and then the latter segment are pushed onto the stack. The latter segment אֲשֶׁ֤ר יִבְחַר֙ יְהוָ֣ה אֱלֹהֶ֔יךָ לְשַׁכֵּ֥ן שְׁמ֖וֹ שָׁם is popped off the stack and again the algorithm moves to step 5) again stopping at a *zakeif katon*. Again, two segments are pushed onto the stack; first אֲשֶׁ֤ר יִבְחַר֙ יְהוָ֣ה אֱלֹהֶ֔יךָ with Sc = S3 and then לְשַׁכֵּ֥ן שְׁמ֖וֹ שָׁם with Sc = S2. We now have three segments on the stack. לְשַׁכֵּ֥ן שְׁמ֖וֹ שָׁם is popped off the stack and the *pashta* divides the segment. Again, two segments are pushed onto the stack; first לְשַׁכֵּ֥ן שְׁמ֖וֹ and then שָׁם. Now finally, each segment is popped off the stack when no requisite separator is found.

At this point the stack has 2 segments:

אֲשֶׁ֤ר יִבְחַר֙ יְהוָ֣ה אֱלֹהֶ֔יךָ with Sc = S3 and below

וְהָלַכְתָּ֙ אֶל־הַמָּק֔וֹם again with Sc = S3.

Note the two segments are identical. Each is divided by the *pashta* with two segments then pushed onto the stack. In each a separator is not found in step 4. After the third segment, the stack is empty and the algorithm halts at step 1).

Note the major division around the word הַמָּק֔וֹם. Each segment before and after הַמָּק֔וֹם have a further division on the words וְהָלַכְתָּ֙ and אֱלֹהֶ֔יךָ. The next (slighter) division occurs on the words שְׁמ֖וֹ and יִבְחַר֙. Note how logical the three levels of division are.

We now turn to the more complex first half of the *pasuk*.

וְלָקַחְתָּ֞ מֵרֵאשִׁ֣ית ׀ כָּל־פְּרִ֣י הָאֲדָמָ֗ה אֲשֶׁ֨ר תָּבִ֧יא מֵאַרְצְךָ֛ אֲשֶׁ֨ר יְהוָ֧ה אֱלֹהֶ֛יךָ נֹתֵ֥ן לָ֖ךְ וְשַׂמְתָּ֣ בַטֶּ֑נֶא

The segment is placed and then popped off the stack, encountering its first member of S2, the *tipḥa* on the word לָ֖ךְ. The segment is divided with וְלָקַחְתָּ֞ מֵרֵאשִׁ֣ית ׀ כָּל־פְּרִ֣י הָאֲדָמָ֗ה אֲשֶׁ֨ר תָּבִ֧יא מֵאַרְצְךָ֛ אֲשֶׁ֨ר יְהוָ֧ה אֱלֹהֶ֛יךָ נֹתֵ֥ן לָ֖ךְ with Sc = S3 and then וְשַׂמְתָּ֣ בַטֶּ֑נֶא with Sc= S2. The former followed by the latter are placed on the stack as described in steps 5 – 7.

וְשַׂמְתָּ֣ בַטֶּ֑נֶא is popped off the stack; since no separator in S2 is found, step 4) discards the segment back into its position in the *pasuk* and returns to step 1). The stack contains just the segment:

וְלָקַחְתָּ֞ מֵרֵאשִׁ֣ית ׀ כָּל־פְּרִ֣י הָאֲדָמָ֗ה אֲשֶׁ֨ר תָּבִ֧יא מֵאַרְצְךָ֛ אֲשֶׁ֨ר יְהוָ֧ה אֱלֹהֶ֛יךָ נֹתֵ֥ן לָ֖ךְ with Sc=S3.

The segment is divided at הָאֲדָמָ֗ה with the syntax again in complete consonance with the standard interpretation; the first segment tells us what is taken and the second tells us from where it originates. The initial segment וְלָקַחְתָּ֞ מֵרֵאשִׁ֣ית ׀ כָּל־פְּרִ֣י הָאֲדָמָ֗ה with Sc = S4 is placed on the stack followed by אֲשֶׁ֨ר תָּבִ֧יא מֵֽאַרְצְךָ֛ אֲשֶׁ֨ר יְהוָ֧ה אֱלֹהֶ֛יךָ נֹתֵ֥ן לָ֖ךְ with Sc = S3. Absent some of the repetitive details, the segment is divided at the *tevir* under the word מֵֽאַרְצְךָ֛, placing אֲשֶׁ֨ר תָּבִ֧יא מֵֽאַרְצְךָ֛ with Sc = S4 on the stack followed by אֲשֶׁ֨ר יְהוָ֧ה אֱלֹהֶ֛יךָ נֹתֵ֥ן לָ֖ךְ with Sc = S3, at the top of the stack.

The stack is popped and the phrase divides on the *tevir* under אֱלֹהֶ֛יךָ. אֲשֶׁ֨ר יְהוָ֧ה אֱלֹהֶ֛יךָ with Sc = S4 followed by נֹתֵ֥ן לָ֖ךְ with Sc = S3 are placed on the stack. נֹתֵ֥ן לָ֖ךְ is popped and discarded, after which אֲשֶׁ֨ר יְהוָ֧ה אֱלֹהֶ֛יךָ is divided by the *darga*, a trop symbol in S4. Both segments are discarded in the next step for different reasons; Sc > 4 for אֲשֶׁ֨ר יְהוָ֧ה and אֱלֹהֶ֛יךָ has no symbol in S4.

אֲשֶׁ֨ר תָּבִ֧יא מֵֽאַרְצְךָ֛ goes through an identical sequence when dividing on the *darga* under the word תָּבִ֧יא.

This leaves only the phrase וְלָקַחְתָּ֞ מֵרֵאשִׁ֣ית ׀ כָּל־פְּרִ֣י הָאֲדָמָ֗ה with Sc = S4. The *gairshaim,* a member of S4, divides the phrase on the word וְלָקַחְתָּ֞.

Subsequent steps discard both segments as before.

Again, let's examine the hierarchy of divisions. The major division is on the word לָ֖ךְ. The further divisions all occur to the left of לָ֖ךְ. The next level of division is on the word הָאֲדָמָ֗ה. The next divisions occur on both the words מֵֽאַרְצְךָ֛ and וְלָקַחְתָּ֞. The next divisions are on the words אֱלֹהֶ֛יךָ and תָּבִ֧יא. The final division is on the word יְהוָ֧ה. Note that the level of a division does not correlate with level of set the symbol is a part of; the *darga* on תָּבִ֧יא and the *tevir* on אֱלֹהֶ֛יךָ are parts of S3 and S4, respectively. However, note the accuracy of those syntactic divisions.

If using the *trop* to establish the *pasuk*'s syntax required the arduous use of an algorithm, I doubt *trop* would maintain the interest it has, minimal as it might be. A proof of recursion, not elementary education on the operation of *trop* required that.