To appear in John Boyle, Jung-Hyuck Lee, and Arika Okrent *CLS 36 [Papers from the 36th Meeting of the Chicago Linguistics Society]Volume1: The Main Session.* 2000.

Linguistica: An Automatic Morphological Analyzer

John Goldsmith
The University of Chicago

This paper derives from an interest in developing algorithms which accept raw linguistic data as input, and produce as their output an analysis of the data, or a grammar.[1] In this day and age, that concern may strike an audience of linguists as a bit unusual, but I think it is a reasonable challenge to undertake. In this paper, I will discuss the techniques and the results of developing just such an algorithm for the purposes of learning morphology on the basis of essentially no prior knowledge save for the data. The primary goal described here is the determination of the location of the breaks between morphemes inside a word. All of the ideas that I will present have been implemented computationally, and the reader is encouraged to download the program in question, *Linguistica*, which runs under Windows and which can perform automatic morphological analysis of a corpus of the user's choice.

Linguists with long memories will remember that Zellig Harris published a renowned paper in 1955 entitled *From phoneme to morpheme* with just such a goal in mind. Some ten years later (1967), at the dawning of the age of computers for everyman, he tried out his algorithm on a small corpus, and got some interesting results, but little overall measurable success. Some years after that, Hafer and Weiss (1974) took up Harris's idea again, refined and sharpened it, consider a dozen variants of it, all sympathetic to Harris' less than fully elaborated notion. Again, the results were intriguing, but they were not standing on the threshold of unadulterated success.[2]

In the first part of this report, I will review Zellig Harris's idea, illustrate where it works well and where it fails. On the face of things, it seems plausible, but the results suggest we should look for alternative approaches.

In the second part, I will attempt to explain why no such local account can succeed at providing a morphology, and give a brief account of an alternative view which selects a morphology on the basis of a global measure of simplicity and optimal compression.

In the final part, I will give a brief overview of how the reader may download a working version of the automatic morphological analyzer, part of a program called Linguistica, available on the Web.

1. Zellig Harris: morpheme boundaries occur at positions of maximum phoneme choice

Harris' proposal in his 1955 and 1967 publications were not quite the same, but the later paper had been explicitly developed to make it computationally feasible; the procedure of the earlier paper was designed in terms of what questions a linguist would ask an informant. Harris' idea was that after any series of letters[3], there will typically be 1 or more letters in possible continuations of a word.[4] The

more choices there are, Harris reasoned, the more likely the spot is to be a morpheme boundary, in the sense that (for example) after the letters *jum*, one might find only two continuation letters in a typical English corpus (*p* as in *jump, jumps, jumping, jumped*, and *b* as in *jumble*). After *jump*, however, there are five possibilities (in the same corpus): *s* (jumps), *e* (*jumped, jumper), i* (*jumping*), *y* (*jumpy*), and word-boundary (*jump*). Each position between letters can be associated with such a count, which is called the *successor variety*. Harris reasoned that positions whose right-branching count was relatively large were likely to be morpheme boundaries. (One can also define the reverse, or mirror-image, tally, called the *predecessor variety*). What does "relatively large" mean? There are two natural interpretations: the successor variety at a particular position is large (1) relative to the successor variety of the immediately preceding and immediately following position, or (2) relative to a particular threshold value. Both interpretations have been explored; I will focus in what follows on the first interpretation.

It would seem that the same consideration ought to select prefixes as well. In a typical 50,000 word corpus of English (the first 50,000 words of the Brown corpus), there are 9 distinct letters that may follow a word-initial *d*, and 18 that can follow word-initial *de*, while after the first three letters of *decided*, only 6 letters can follow. So the Harrisian considers the possibility that positions whose successor variety is a local maximum—larger than the successor variety of the preceding and following position—marks a morpheme break, we declare a morpheme boundary after *de* (see Table 1 for sample data).

So far, so good: but the procedure *also* declares a morpheme boundary after (almost) *all* initial *de*-s, most of which are *not* prefixes (as in *dead, demon, deep, Delhi*, and so on). And it also declares equally loudly that there is a morpheme boundary after *da*, after *di*, and after *do*, too! *Fa, fe, fi, fo* and *fu* are likewise declared morphemes (as are a wide range of CV sequences, and some CCV sequences like *bla, bri, bro, cha, gra, gri, gro, pla, pre, pri, pro, sha, sho, sta, ste, sto, tra, tri* as well as *imp, qui* and *wor,* for the same reasons – the method cannot distinguish between phonological freedom brought on by morphology or simply left open by phonology). *oppo-* is also identified as a morpheme (because of the existence of *oppo-se, oppo-nent,* and *oppo-rtunity*), and *conservatives* is parsed *co-nserv-ati-ves.*

What's wrong with the Harrisian approach? The main problem with Harris's algorithm is that it cannot distinguish between freedom due to phonological combination and freedom due to a boundary between two morphemes. And this is a very difficult problem to fix up with small fixes, for the very problem that causes the distress is the problem the algorithm is supposed to deal with in the first place.[5]

2. Naive description length and an evaluation metric

Harris asked the right question, but the kind of answer that he offered was in an important sense heading off in the wrong direction. (Current received linguistic

opinion, if such could be said to exist, is that his question was the wrong one to ask, and that it is a matter of pure indifference whether his answer worked to any degree.) There *is* no *local* criterion that can determine where and how a word should be divided into morphemes; there is only a *global* criterion, or so I shall suggest in this section. A global criterion is one that is based on all decisions about all of the words in the entire lexicon: the correct analysis of each word is potentially influenced by decisions about any and all other words, directly or indirectly. Does this spell the end to Zellig Harris's dream of a procedural or algorithmic account of morphological analysis? Not at all, and that is what I wish to show in what is necessarily a sketchy fashion in the rest of this paper; I have provided a detailed account elsewhere (Goldsmith 2000).

Let us begin with some observations which can serve as starting points on the road to a precise theoretical formulation:

(1) a. A word which is morphologically complex reveals that composite character by virtue of being composed of (one or more) strings of letters (or phonemes) which have a relatively high frequency throughout the corpus.

b. An explicit recognition of such high frequency substrings allows for a compact description of the words of a language. Lexicographers know what they are doing when they indicate the entry for the verb *laugh* as *laugh, ~s, ~ed, ~ing*; they recognize that the tilde " ~ " allows them to utilize the regularities of the language in order to save space and specification, and implicitly to underscore the regularity of the pattern that the stem possesses. A measurement of the degree of compression allowed in this way will be a very good measure of the success of a morphological analysis.

c. *However*: morphological analysis is not *merely* a matter of frequency of particular strings of letters. Not every word that ends in *–ing* is morphologically complex (*string, sing*, etc.). Even more: every word that ends in *–ity* also ends in *–ty*, and so final *–ty* has a higher frequency than *–ity*; but still, *-ty* is a suffix only in a few words (like *sixty*), while *-ity* is a suffix in far more words, despite its lower frequency (*insanity, precocity*, etc.). And *y* has a higher frequency than either of them, and it *is* a suffix in some words (like *dirty, runny*, etc.), but it is not in *insanity, precocity*, and so forth. The point is this: the frequencies matter, but only in the overarching context of a total morphological analysis of all of the words of the language.

Let us consider the following proposal:

(2) Naive Minimum Description Length: In analyzing a corpus, devise an analysis of the words into stem + suffix with the requirement that every stem and every suffix must be used in at least 2 distinct words. Tally up the total

number of letters in (a) each of the proposed stems, (b) each of the proposed suffixes, and (c) each of the unanalyzed words, and call that total the "naive description length".

This is a perfectly reasonable statement; it faithfully represents the view that has suggested for many decades and which in some respects has become received opinion: long-term memory is costly, and a grammar should be valued in proportion to how little information it presupposes among the underlying entries in the lexicon of the language in question.

I will explain below why I refer to this as a *naive* Minimum Description Length view; but it is not at all a bad initial hypothesis. As a hypothesis, it has one important characteristic: using it, we can evaluate two or more competing hypotheses *independently* of how we arrived at the hypotheses. Any particular morphological hypothesis regarding a corpus will consists of a list of stems, suffixes, and unanalyzed words, and assigning a naive description length is very easy, so choosing the *best* analysis among a set of analyses in hand is easy (except in the case where two analyses coincidentally have the same naive description length, an unlikely case with a large corpus). For those with long memories, this notion of a naive description length corresponds precisely to Chomsky's notion of a evaluation metric of grammar based on the grammar's total length, a notion which Chomsky abandoned in the shift to a principles and parameters view of universal grammar, first in Chomsky and Lasnik 1977, and later in Chomsky 1981.

Let us next observe that a theory which has an explicit description length formula (= evaluation metric) can be associated with an explicit automatic learning algorithm if two further conditions hold: (1) first, we must have an initial boot-strapping procedure which can devise *some* sort of morphological analysis of the data; (2) second, we must have toolbag of procedures which can take a good look simultaneously at the data and the analysis, and propose changes that *might* improve the analysis. It is extremely important to understand that the toolbag of procedures–heuristics, we may call them–need not be particularly clever or even good at what they do, for their effects will be accepted *if and only if they reduce the total naive description length.* (Later we will have a *description length* which is not "naive", but that will have no effect on this basic notion.) Selecting a grammar reduces now to a problem of optimization.

Consider an example such as the following. Suppose the corpus that we were analyzing consisted of the following words: *walk, walks, walking, walked, jump, jumps, jumping, boy, boys, sing, sings*. There are 54 letters in this set of words, and thus the strictest morphological analysis, which posits no analysis for each word, would have a naïve description length of 54 letters. An analysis which proposes a stem list {*walk, jump, boy, sing*}and a suffix list {*s, ing, ed*} has a total naïve description length of 15 + 6 = 21 letters, which is clearly an improvement. Observe that if we consider an analysis in which the word *sing* were analyzed as *s + ing*, the total description length would increase, since it would require positing

a new stem *s*; hence the desire to achieve minimum analytical length keeps us from embracing that hypothesis.

3. Bootstrapping heuristic

It is not difficult to construct algorithms to produce an initial morphological analysis of a corpus of languages whose morphology is as simple as those of Indo-European languages.[6] The first step is to produce a candidate set of suffixes (or prefixes). I will describe one very simple method which gives surprising accurate results.

As we have already noted, we would expect a morpheme to have the property that the letters (phonemes) that compose it occur far more frequently in the *right* order – the order that spells the morpheme! – than would be expected letters were combined in a random fashion. That seems so obvious that it may be hard to imagine that its implementation could return something of value – and yet it does. Let us count the raw frequencies of occurrence of each letter in a corpus, and call the probability $p(\lambda)$ of a letter λ simply the proportion of the count of λ to the total number of letters. If language were random – if there were no morphemes and no phonotactics – then the expected frequency of a sequence of letters would be the product of the probabilities of each of the letters: $p(\lambda_1)p(\lambda_2)...p(\lambda_n)$. So we will compare the actual frequencies of all such sequences to those expected probabilities, and to keep the numbers manageable we will compute the logarithm of this ratio of comparison: $\log p(\lambda_1 \lambda_2 ... \lambda_n) / p(\lambda_1)p(\lambda_2)...p(\lambda_n)$.[7] This measures, if you will, the stickiness of the sequence of letters (phonemes), but we also care about how often the sequence as such appears, and hence we use the measure in (3), where λ_n is a word boundary "#" in all cases.

$$(3)\quad p(\lambda_1\lambda_2...\lambda_n)\log\frac{p(\lambda_1\lambda_2...\lambda_n)}{p(\lambda_1)p(\lambda_2)...p(\lambda_n)}$$

It is straightforward to apply this measure to all sequences of 2 – 6 letters (phonemes) from a corpus and select the top 40 sequences according to the measure in (3); illustrative examples from corpora of 50,000 words are given in (4).

In this fashion, we select the top 100 suffixes in our corpus, and consider analyzing into *stem + suffix* any words which end in a candidate suffix. Many words will not be analyzed at all; others will be multiply analyzed; a few will have unique factorizations. We wish to know how many occurrences there are of each stem and each suffix. In the case of words with a unique factorization, the task is simple, but in the case of words with multiple factorizations (say, an English word such as *laughing* may be analyzed as *laugh-ing*, *laughi-ng*, or *laughin-g*), we assign part of the word's count to each of the analyzes, in a

proportion that depends both on the length of the suffix and on the stem's and suffix's frequency elsewhere. Such a process requires an iterative computation,

(4) The top 40 suffixes in 50,000 word corpora based on (3)

English	French	Latin	Italian	Spanish (Quijote)
(e)s	(r)e	(((i)b)u)s	(((e)n)t)e	se
(t)e	((l)e)s	((t)u)m	((a)t)o	ar
((t)e)d	(((m)e)n)t	(i)t	((a)t)a	ó
(((t)i)o)n	(((t)i)o)n	(u)e	((n)t)i	ado
(l)y	(e)r	(t)a	((i)o)ne	le
(((t)i)n)g	ée	(t)i	(a)no	an
(s)t	a	(t)is	(a)re	ra
(e)r	le	o	ia	(n)te
(a)l	ue	(u)r	(a)le	to
ts	te	((r)u)nt	ni	(a)ndo
(((i)o)n)s	(o)ns	(t)es	li	ía
rs	ne	am	(i)co	en
((e)n)t	l	em	io	ro
m	(i)que	as	((i)c)a	(a)ba
an	ant	que	ra	la
ers	ts	at	ri	ón
h	it	et	ro	ta
ic	és	tur	ono	ada
ss	ie	rum	do	ia
ce	res	ae	si	dos
us	(t)é	ia	na	er
	tes	os	nto	is
	se	re	se	on
	ce	te	ati	lo
	x	tus	mos	mos
	ées	mus	so	so
	me			

but rather quickly a preliminary morphological analysis is achieved. To this analysis we apply the condition mentioned in (2), that is, we eliminate any hypothetical stems or suffixes that occur in only one word. In fact, we impose a much stronger restriction, one which goes back to Greenburg's criterion: every stem and every suffix must participate in at least one commuting structure as in (5). Every stem must share with at least one other stem the set of suffixes with which it appears in the corpus.

When we apply this algorithm to English, we derive sets of suffixes, called *signatures,* all of which appear with a set of stems. In English, the most frequent signature is composed of the suffixes *–ing, -ed, -s* and a null suffix (*NULL*). We, of course, call these "verbs", and a typical 50,000 word corpus will find about 15 such stems, such as *ask, add, attend, end, record, kick, talk,* etc. Other high-

frequency signatures discovered are nouns, most of which appear with two suffixes, *NULL* and *s*; some nouns appear with three suffixes: *NULL*, *-s*, and *'s*. Adjectives emerge with the signatures *NULL* and *ly*, some others with these suffixes plus *–ment*.

(5) Greenburg square

abound	
add	*NULL*
administer	*ed*
amount	*ing*
arrest	*s*
...	

While the result of the first pass bootstrapping heuristic is quite good, it contains a number of errors in the detail. To discover them and correct them, we need to allow the iterative heuristics to suggest modifications, and then to calculate whether those modifications lead to an overall improvement in the compactness of the analysis.

4. Minimum Description Length

The quantitative evaluation which we compute needs to be more complex than the simple formula that we have termed in (2) "naive description length." An adequate description length, as proposed and explored by Rissanen 1989 in a quite general framework (which is to say, it does not explicitly address the field or problems of linguistics), is composed of two parts: a first term which describes how well the analysis fits the data, and a second term which describes the length or complexity of the analysis. Notions derived from information theory are used in both parts, and the two are added together to provide the total description length. By seeking the analysis that minimizes this figure, what we are led to is that analysis which best fits the data without paying too large a price by creating an analysis without too much detail and without overfitting the data.

The details are rather complex, and available elsewhere (see note 1). But the essence of the approach can be summarized here, though I will nonetheless issue the following warning: it is easy to get lost, at first approach, and to wonder whether behind such arithmetic complications could lie any insight that could not just as easily be formulated without numbers. A considerable amount of experimentation with this material has convinced me that this approach does an excellent job of making explicit what it is that we as linguists prefer when we compare two analyses of the same data.

We determine how well the analysis fits the data by computing the probability *p* assigned to the data by the model, and we interpret that as corresponding to a compression into $-log_2(p)$ bits (i.e., the base 2 logarithm of the

value of *p*). The probability that a morphology assigns to an entire corpus is the product of the probabilities that it assigns to each individual word, and the probability that a morphology such as ours assigns to an individual word *w*, composed of stem *t* and suffix *f*, in signature σ, is the product of three terms: the probability of signature σ, the probability of the stem *t* given the signature, and the probability of the suffix *f* given the signature. The probability of a signature σ is essentially the fraction of the words of the corpus that represent that signature σ, just as the probability of a stem *t*, given a signature σ, is the fraction of the occurrences of that signature σ in which the stem is *t*. The higher the probability that the analysis (that is, the morphology) assigns to the corpus, the better the analysis has done in modeling the data, all other things being equal. Of course, the higher the probability is, the smaller will be its base 2 logarithm multiplied by -1.

The complexity of the morphology is a bit more complex to explain, but most of it can be summarized as follows. Almost all structure—structure of a morphology, or even the structure of a grammar more generally—can be understood and presented as a set of lists, in which each item in the list is a "pointer": a connection either to another list, or to a primitive item (such as a letter or phoneme). The complexity of a list with N items in it can be calculated in a straightforward way: it consists of the sum of the length of each of the pointers on the list, plus the length of the statement which makes it explicit that there are exactly N items, no more and no less; this latter statement takes slightly more than $\log_2$ N bits to formulate. Finally, the length of a pointer to an a item *i* on a list is of length $-\log_2 prob(i)$, where again the units in which this length is measured is bits. When all of this is tallied up, a clear measurement of the complexity of an analysis is produced, and an automated process can determine which of two analyses is to be preferred.[8]

One of the great strengths of an approach such as this is that one knows exactly how much it costs to express a generalization in one's grammar (here, morphology), and one knows exactly how much improvement in the treatment of the data one has gained by investing that much additional effort in the analysis.

5. Improvement heuristics

There are four heuristics that are explored by the program.

First, the suffixes are examined to see if they are accidentally combinations of true suffixes. For example, the bootstrapping heuristic is likely to come up with the hypothesis in English that *ments* is a suffix, but it is necessary to come to the conclusion that a word ending in such *-ments* really has the structure [[X-ment] s]. So every suffix is checked to see if it is composed of the concatenation of two independently identified suffixes, and the overall analysis is examined to see if its overall compactness is improved by splitting such a suffix in two.

Second, signatures are examined to see if an error in analysis has occurred. For example, there is such a high proportion of stems that end in *t* in English (*defeat*, *subject*, *respect*, *draft*, *merit*, etc.) that the bootstrapping algorithm

routinely hypothesizes that these are the stems *defea-*, *subjec-*, etc., which take the suffixes *t, ted,* and *ts*. But noticing the letters common to all of the suffixes, the program can try out the modified analysis in which the *t* is shifted to the stems, and since the overall compactness is improved by this move, the shift is carried out. Putting this another way, a simple letter-counting approach may lead one to conclude that a more economical analysis is available for English if, in addition to the suffixes *ed,ing,s* we also include the suffixes *ted, ting, ts*, extracting the generalization that *t* is a very high frequency stem-final letter. The more articulated information-theoretic model described above rejects this alternative on the grounds that the pointers to the suffixes *ted, ting,* and *ts* are too much longer than the pointers to the suffixes *ed ,ing,* and *s* (and they are longer because of their lower frequencies).

Third, individual stems are examined, and if two stems differ only by one letter, they are examined to see if one stem should really be eliminated in favor of the other. For example, the analyses *celebrat-ed* and *celebra-tion* give rise to two stems, *celebrat* and *celebra-*. The overall compactness is improved if *celebration* is instead analyzed with the suffix *–ion,* and the stem *celebra* is eliminated in favor of the sole stem *celebrat-*.

Finally, the decision must be made as to whether some analyses involve too few examples exist which illustrate the pattern to justify maintaining it altogether.

After the heuristics have been tried out, the final analysis is quite an accurate overall morphology. The most difficult aspect is the last one mentioned – determining whether a pattern with few examples supporting it is worth including in the overall morphology.

6. An implementation available to the reader

The analysis described in this paper is implemented in a program called *Linguistica*, which is available for downloading and which runs under Windows. The user must have a text file containing the data which will be analyzed.

By clicking on **Help** on menu bar, the user can get some basic information about how to use the program and some of the options open to the user. In order to perform morphological analysis, the user first chooses the number of words which she wishes to analyze with the command **size** followed by the number of words. The command "**read;**" then brings up a dialog in which the user can indicate which file should be read as input. Finally, the command "**AM;**" calls the automatic morphological analysis. To view output, one can display various collections that have been computed, notably **signatures; stems;** and **stems;**

Table 2 and Table 3, we present a small sample of the output of this algorithm applied to a 500,000 word corpus of English.

In work in progress, we are applying similar techniques to parallel questions in phonology and syntax, using machine learning techniques to provide answers to questions regarding classification and cooccurrence.

Table 1 Successor variety

Initial sequence	Successor variety	examples
d	**9**	
da	11	
de	**18**	
dea	**5**	
deb	**2**	
dec	**6**	
ded	**2**	
dedi	**1**	dedication
dedu	**2**	deductible
dee	**1**	deep
def	**3**	
deg	**1**	degree
del	**5**	
dem	**2**	
den	**6**	
dep	**5**	
deq	**1**	dequindre
der	**2**	
des	**7**	
det	**3**	
dev	**3**	
dew	**1**	dewey
dey	**1**	dey
di	13	
dj	1	djakarta
do	16	
etc.		
ap	**3**	
apa	1	apartment
app	**5**	
appa	**1**	apparent
appe	**2**	
appl	**3**	
appo	**1**	appoint
appr	**3**	
apr	1	april

Table 2 Sample from top 10 signatures, English 500,000 word corpus

1. NULL.ed.ing.s	**4. NULL.s**	**7. NULL.ed.ing**
accident	aberration	applaud
ad	abolitionist	arrest
administer	abortion	astound
afford	absence	blast
alert	abstractionist	bless
amount	abutment	bloom
appeal	accolade	boast
assault	accommodation	bolster
attempt	accommodation	broaden
2. 's.NULL.s	**5. e.ed.es.ing**	cater
adolescent	achiev	**8. NULL.er.ing.s**
afternoon	assum	blow
airline	brac	bomb
ambassador	chang	broadcast
amendment	charg	deal
announcer	compris	draw
architect	conced	drink
assessor	conclud	dwell
association	decid	farm
3. NULL.ed.er.ing.s	describ	feed
attack	**6.e.ed.er.es.ing**	feel
back	advertis	**9. NULL.d.s**
bath	announc	abbreviate
boil	bak	accommodate
borrow	challeng	aggravate
charm	consum	apprentice
condition	enforc	arcade
demand	gaz	balance
down	glaz	barbecue
	invad	bruise
	liv	catalogue
	pac	costume
		10. NULL.ed.s
		acclaim
		beckon
		benefit
		blend
		blister
		bogey
		bother

Table 3 English suffixes from *Tom Sawyer*

Suffix	Occurrences	Remarks
s	3290	
ed	2447	
ing	1685	
er	1531	
e	1174	
ly	857	
's	738	
d	738	
y	625	
n	472	
on	346	spurious (*bent-on, rivers-on*)
es	329	
t	291	
st	270	signature NULL.ly.st, for stems such as *safe*
en	229	*behold, deaf, weak, sunk*, etc.
le	176	error: analyzed *le.ly* for *e.y* (stems such as *feeb-, audib-, simpl*)
al	167	
n't	164	
nce	151	signature *nce.nt*, for stems *fragr-dista-, indiffere-*
ent	148	spurious: (stems such as *pot-*)
tion	135	
r	135	
ter	132	spurious
k	129	spurious
ful	125	
ion	124	
'll	117	
an	117	spurious
ness	116	
nt	84	(see above)
ted	84	*chat-ted, submit-ted*, etc.
est	75	
ity	71	
ous	68	
ard	65	*drunk-ard*, etc.
able	64	
ious	57	
less	51	
ment	48	
id	48	signature *id.or* for *horr-, splend-, liqu-*
ure	47	

ive	44	
ty	39	as in *novel-, uncertain-, six-, proper-*
ence	38	
ily	31	
ward	21	
ation	21	
led	18	spurious
'd	18	
ry	17	spurious: stems such as *glo-* with signature *rious.ry*
rious	15	see immediately preceding
rs	12	spurious
ned	11	*awake-ned, white-ned, thin-ned*
ning	11	*begin-ning, run-ning*
age	9	
h	7	spurious
te	6	should be *ate*: e.g., *punctua-te*
ant	4	*triumph-ant, expect-ant*
r's	4	spurious
ance	4	

References

Chomsky, Noam. 1975 [1955]. *The Logical Structure of Linguistic Theory*. New York: Plenum Press.

Chomsky, Noam. 1981. *Lectures on Government and Binding*. Dordrecht: Foris Publications.

Chomsky, Noam and Howard Lasnik. 1977. Filters and control. *Linguistic Inquiry* 8/3:425-504

Goldsmith, John. 2000. Unsupervised learning of the morphology of a natural language. Ms. Available at http://humanities.uchicago.edu/faculty/goldsmith.

Hafer, M. A., & Weiss, S. F. (1974). Word segmentation by letter successor varieties. *Information Storage and Retrieval* 10:371-385.

Harris, Zellig. 1955. From phoneme to morpheme. *Language* 31:190-222, reprinted in Harris 1970.

Harris, Zellig. 1967. Morpheme boundaries within words: report on a computer test. *Transformational and Discourse Analysis Papers*, 73. Reprinted in Harris 1970.

Harris, Zellig. 1970. *Papers in Structural and Transformational Linguistics*. Dordrecht: D. Reidel.

Langer, Hagen. 1991. Ein automatisches Morphsegmentierungsverfahren für deutsche Wortformen. Unpublished doctoral dissertation, Georg-August-Universität (Göttingen).

Rissanen, Jorma. 1989. *Stochastic Complexity in Statistical Inquiry*. Singapore and Teaneck NJ:World Scientific Publishing Co.

Notes

1. I have profited from the assistance of a large number of linguists and non-linguists over the course of the work described in this paper, most recently that of Derrick Higgins and Svetlana Soglasnova. The work is described in greater detail in Goldsmith 2000, which is available at http://humanities.uchicago.edu/faculty/goldsmith, as is the software described herein. This work has been supported by a grant from Argonne National Laboratory/University of Chicago.

I do not discuss here other, very recent work on this subject, notably by Sylvain Neuvel and by Derrick Higgins (see his paper on this in the present volume) of the University of Chicago, and also by Marco Baroni of UCLA.

2. I became aware just days before this conference of a dissertation on the subject of morphological analysis by Hagen Langer (1991). Langer's remarks on the problem in general are insightful and very much *a propos*.

3. Throughout I shall refer to either phoneme or letter interchangeably. It is true that English (and French) have orthographies which are rather far from either phonetic or phonological reality, but what we are interested in are algorithms that work over a vast range of languages. One reasonable way to interpret what we are doing is to say that we are investigating what the morphological structure of a language would be if its phonological representations were just like our present English orthography.

4. In Harris 1955, he is concerned with utterances, while in 1967, Harris is concerned with words. The difference is not important for our discussion here, as it was not for Harris.

5. I have not done justice to Harris' account here, nor to Hafer and Weiss' (1974) careful empirical study of the various ways in which Harris's insights could be algorithmically implemented. But if I have not done justice to the details, I do believe that I have done justice in the final analysis; there are, I am convinced, no modifications within the program that Harris proposed, strictly construed, that solve the morpheme identification task.

Another place where the Harrisian algorithm fails turns up in cases where a set of stems appears with a set of suffixes that all begin with the same letter. This is more common than one might have expected if one had not looked at some corporal for several languages. French presents several cases of this sort. A number of nouns and adjectives end with the suffixes *al* and *aux* (nouns such as *journal-journaux* 'newspaper(s)' and adjectives such as *amical-amicaux* 'friendly (masc. sg. and masc. pl.)'). For these forms, the Harrisian analysis will point towards an incorrect stem such as *journa* and *amica*, and this decision will appear to be supported, spuriously, by the words *mieux* 'better' and *miel* 'honey' (*mie + l-ux*). The Romance languages present quite a few cases like this, where due to the historical origin of the verbal suffixes, many of the common verbal suffixes all begin with the same letter. For any verb stem that happens to appear in a corpus with only suffixes beginning with the same letter, the algorithm will give the wrong result, shifting the suffix-initial letter wrongly to the stem.

6. I do not really wish to underestimate the difficulty of coming up with an algorithm that produces an initial morphology of a corpus. Zellig Harris' is the only algorithm to be found in the prior literature, that I am aware of, that begins with no prior knowledge or analysis provided by a linguist and which produces a reasonably small set of hypothetical suffixes. Bear in mind that crude brute force is not the right way to go. For example, suppose we confronted a corpus of 5,000 words, each of length 7 letters. Each word can be divided into stem + suffix in 7 ways, and there are thus 7^{5000} possible morphologies. This is an unimaginably large number. But no crude brute force is indicated.

7. This is sometimes called the *pointwise mutual information* in the computational linguistics literature.

8. Providing an explicit means for algorithmically comparing solutions, instead of having a human being make the comparison "by eye" or by seat-of-the-pants intuition, is entirely parallel to

the requirement established by early generative grammar to make grammatical analyses explicit and not dependent on linguists' intuitions about a language. That is, one of the major points that generative grammar succeeded in making was that it was not sufficient for a linguist to offer a set of informal guidelines explaining how to use (for example) the dative case in German; what was necessary was a set of *procedures* explicit enough that they could be implemented by someone who knew nothing about German—a computer, for example. The reason was never that a computer would do a better job; it was, rather, that if we allow an intelligent person to be responsible for applying the informal specifications of our analysis, we will never know how much of the success is due to the human's tacit contribution to the implementation. The situation is quite parallel in the present case: what we need is a way to select among hypotheses, given a (large) set of observations from a language. If we allow that judgment to be made by unanalyzed human intuitions, we cannot know (in the case of either success *or* failure) whether success or failure is due to the linguistic theory or to the fuzzy application of human intuition.