The contributors to the glossy computer hobbyist publications that dot the magazine racks of bookstores and supermarkets these days in ever increasing numbers would have us believe that the day is rapidly approaching when personal computers will be as common in American households as toasters. While such extravagant claims are admittedly somewhat suspect, the booming sales of TRS-80's, Apple II's, Commodore Pets, and other home computers should make educators in all fields begin to devote some serious thought to how they might modify their teaching in order to reap the greatest pedagogical rewards from the unprecedented availability of sophisticated computer hardware and the resultant growth in their average student's computer literacy. Linguists especially should, I think, be quick to realize that the machines they buy to balance their checkbooks, teach their children arithmetic, and serve as their chess partners can prove quite useful in their teaching as well.

One area of linguistics in which personal computers can be put to good use is phonology. This is hardly surprising in view of the remarkable resemblance which phonological descriptions (at least those cast in the generative mold) bear to computer programs. Until recently, however, only those phonologists who were simultaneously specialists in computer science had access to machines that could process their 'software' while the vast majority of their colleagues were forced to 'run' derivations by hand.

To illustrate how a phonologist might use a personal computer in his or her work, I shall present, in Section 1 of this article, a set of seventeen phonological rules formulated in distinctive feature notation and containing most of the abbreviative devices commonly found in the generative phonological literature. In Section 2 these rules will be translated into Radio Shack's TRS-80 Level II BASIC and incorporated into a program that will run derivations in which the rules are applied in a specified order. Section 3 will provide some sample derivations which can be generated with this program. Section 4 will supply some technical notes about the program. Finally, in Section 5, I shall outline some of the pedagogical applications for which such a program is suited.
1. The Rules

It will be assumed that the phonological rules presented in this section apply to underlying representations composed of the segments and boundaries defined in classificatory matrix (19). The distinctive features abbreviated in that matrix are to be interpreted as follows: SE (segment), WB (word boundary), CS (consonantal), OB (obstruent), SY (syllabic), CT (continuant), AN (anterior), CO (coronal), VO (voiced), HI (high), LO (low), BA (back), RN (round), TN (tense), NA (nasal), ST (stress). The symbols chosen to represent the segments in (19) are all ones that can be typed on the keyboard of a TRS-80 home computer equipped with Radio Shack's lower case modification kit. Those symbols which do not belong to the International Phonetic Alphabet are explained in (18). It will be noted that I am using the signs + and # to represent a word-internal morpheme boundary and a word boundary, respectively. Phrase boundaries will be marked by a cluster of two or more #'s. Unless otherwise indicated by the label 'morpheme-level' or 'phrase-level', each rule will be regarded as applying within the domain of the word.
(1) **Obstruent Dissimilation**

\[ [+\text{obstruent}] \rightarrow [-\text{continuant}] / [-\text{continuant}] \]

I.e., if the second of two contiguous obstruents is a stop, the first must be a fricative, but if the second is a fricative, the first must be a stop.

(2) **High Back Vowel Lowering**

\[
\begin{align*}
  [+\text{syllabic}] & \rightarrow [-\text{high}] / [-\text{nasal}] \\
  [+\text{low}] & \rightarrow [+\text{low}] \\
  [+\text{back}] & \rightarrow [-\text{back}] 
\end{align*}
\]

I.e., [u] is lowered to [o] and [ı] is lowered to [ı] before an [a] that is either contiguous to the high vowel or separated from it by a single nonnasal consonant.

(3) **Front Vowel Rounding**

\[
\begin{align*}
  [+\text{syllabic}] & \rightarrow [+\text{round}] / [-\text{coronal}] \\
  [+\text{low}] & \rightarrow [-\text{low}] \\
  [-\text{back}] & \rightarrow [+\text{back}] 
\end{align*}
\]

I.e., a high or mid front vowel is rounded if it stands immediately after a labial consonant.

(4) **Consonant Voicing**

\[ [+\text{consonantal}] \rightarrow [+\text{voice}] / [+\text{voice}] / [+\text{voice}] \]

I.e., a voiceless consonant other than [h] or [?] is voiced in voiced surroundings.

(5) **Stress Placement**

\[ [+\text{syllabic}] \rightarrow [+\text{stress}] / C O (C V) O \# \]

I.e., the last closed syllable of a word is stressed, or, if there are no closed syllables, the first syllable is stressed.

(6) **h-Deletion**

\[
\begin{align*}
  [+\text{consonantal}] & \rightarrow \emptyset / [-\text{syllabic}] \\
  [+\text{continuant}] & \rightarrow \emptyset \\
  [-\text{voice}] & \rightarrow \emptyset 
\end{align*}
\]

I.e., [h] is deleted after a contiguous consonant.
(7) **Word-final Obstruent Devoicing**

\[ [+\text{obstruent}] \rightarrow [-\text{voice}] /\_\_ \]

i.e., an obstruent is devoiced in word-final position.

(8) **Geminate Vowel Simplification**

\[
\begin{array}{c|c}
+\text{syllabic} & +\text{syllabic} \\
\text{\(\alpha\text{high}\)} & \text{\(\alpha\text{high}\)} \\
\text{\(\beta\text{low}\)} & \emptyset \\
\text{\(\gamma\text{back}\)} & \text{\(\gamma\text{back}\)} \\
\text{\(\delta\text{round}\)} & \text{\(\delta\text{round}\)} \\
\end{array}
\]

i.e., a vowel is deleted if it stands immediately after a vowel that shares its specifications for the features high, low, back, and round.

(9) **Metathesis**

\[
[+\text{consonantal}] [+\text{obstruent}] \rightarrow 2 1 \\
\begin{array}{c}
\text{\(\alpha\text{high}\)} \\
\text{\(\beta\text{low}\)} \\
\text{\(\gamma\text{back}\)} \\
\text{\(\delta\text{round}\)} \\
\end{array}
\]

i.e., a liquid and a following obstruent are metathesized.

(10) **Nasal Assimilation**

\[
[-\text{syllabic}] [+\text{nasal}] \rightarrow [\text{\(\alpha\text{anter}\)}] [\text{\(\beta\text{coronal}\)}] [\text{\(\gamma\text{high}\)}] [\text{\(\delta\text{back}\)}] [+\text{consonantal}] [+\text{obstruent}] [\text{\(\alpha\text{anter}\)}] [\text{\(\beta\text{coronal}\)}] [\text{\(\gamma\text{high}\)}] [\text{\(\delta\text{back}\)}] [-\text{continuant}]
\]

i.e., a nasal consonant assumes the point of articulation of an immediately following oral stop other than [?].

(11) **i-Umlaut**

\[
[+\text{syllabic}] \rightarrow [-\text{back}] [\text{\(\alpha\text{high}\)}] [\text{\(\beta\text{coronal}\)}] [\text{\(\gamma\text{high}\)}] [\text{\(\delta\text{back}\)}]
\]

i.e., a back vowel is fronted before a syllable containing a high front vowel if no more than two consonants intervene.
(12) **Alpha-Switching**

\[
\begin{array}{c}
\text{[+syllabic]}
\text{[+back]}
\text{[+round]}
\text{[+high]}
\end{array}
\rightarrow
\begin{array}{c}
\text{[+high]}
\end{array} \backslash \begin{array}{c}
\text{[+back]}
\end{array}
\]

I.e., [u] is fronted to [y] and [o] is raised to [u].

(13) **Nasalization**

(Morpheme-Level) \[ [+\text{syllabic}][+\text{nasal}]c \rightarrow [+\text{nasal}]\emptyset \]

I.e., a nasal consonant which stands between a preceding vowel and a following consonant is deleted with compensatory nasalization of the vowel, provided that all three segments belong to the same morpheme.

(14) **Vowel Insertion**

(Phrase-Level) \( \emptyset \rightarrow a/ [+\text{obstruent}]_1 [+\text{obstruent}]_2 \)

I.e., within a phrase, an [a] is inserted between two obstruents even if the latter belong to different words.

(15) **Nasalized Vowel Laxing**

\[
\begin{array}{c}
\text{[-syllabic]}
\text{[+nasal]}
\text{[-back]}
\text{[-low]}
\end{array}
\rightarrow
\begin{array}{c}
\text{[-tense]}
\end{array} \backslash \begin{array}{c}
\text{[-high]}
\end{array} \backslash \begin{array}{c}
\text{[-stress]}
\end{array}
\]

I.e., a nasalized high or mid front vowel is laxed and, if unstressed and high, lowered to mid.

(16) **Palatalization**

\[
\begin{array}{c}
\text{[+consonantal]}
\text{[+obstruent]}
\text{[+continuant]}
\text{[-anterior]}
\text{[+coronal]}
\end{array}
\rightarrow
\begin{array}{c}
\text{[-anterior]}
\text{[+coronal]}
\end{array} \backslash \begin{array}{c}
\text{[+syllabic]}
\text{[+back]}
\end{array} \backslash \begin{array}{c}
\text{[-back]}
\end{array}
\]

I.e., [s] and [x] become [SH] and their voiced counterparts [z] and [GH] become [ZH] before a high front vowel.
(17) Glottal Stop Insertion

(Phrase-Level) \( \emptyset \rightarrow \text{?/[syllabic]/}[+syllabic] \)

i.e., a glottal stop is inserted before a word-initial vowel if an immediately preceding word within the same phrase ends in a vowel.

(18) Definitions of unusual phonetic symbols

<table>
<thead>
<tr>
<th>Symbols</th>
<th>Definitions</th>
</tr>
</thead>
<tbody>
<tr>
<td>NG</td>
<td>a voiced velar nasal</td>
</tr>
<tr>
<td>@</td>
<td>a low front unrounded vowel</td>
</tr>
<tr>
<td>†</td>
<td>a mid central unrounded vowel</td>
</tr>
<tr>
<td>Ø</td>
<td>a tense mid front rounded vowel</td>
</tr>
<tr>
<td>GH</td>
<td>a voiced velar fricative</td>
</tr>
<tr>
<td>SH</td>
<td>a voiceless palatoalveolar fricative</td>
</tr>
<tr>
<td>ZH</td>
<td>a voiced palatoalveolar fricative</td>
</tr>
<tr>
<td>?</td>
<td>a voiceless glottal stop</td>
</tr>
<tr>
<td>!</td>
<td>a high central unrounded vowel</td>
</tr>
<tr>
<td>I</td>
<td>a lax high front unrounded vowel</td>
</tr>
<tr>
<td>Y</td>
<td>a lax high front rounded vowel</td>
</tr>
<tr>
<td>E</td>
<td>a lax mid front unrounded vowel</td>
</tr>
<tr>
<td>OE</td>
<td>a lax mid front rounded vowel</td>
</tr>
<tr>
<td>M</td>
<td>a diacritic indicating that the preceding vowel is nasalized</td>
</tr>
<tr>
<td>'</td>
<td>a diacritic indicating that the preceding vowel is stressed</td>
</tr>
</tbody>
</table>
(19) Classificatory matrix

|     | #  | + | a | i | u | p | t | k | b | d | g | f | s | x | e | o | m | n | NG | @ | † | l | r | j | w | y | ø | v | z | GH | SH | ZH |
| SE  | -  | + | + | + | + | + | + | + | + | + | + | + | + | + | + | + | + | + | + | + | + | + | + | + | + | + | + | + |
| WB  | +  | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
| CS  | -  | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
| OB  | +  | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
| SY  | -  | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
| CT  | -  | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
| AN  | -  | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
| CO  | -  | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
| VO  | +  | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
| HI  | -  | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
| LO  | -  | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
| BA  | -  | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
| RN  | -  | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
| TN  | -  | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
| NA  | -  | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
| ST  | -  | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - |

? h aM IM uM eM oM @M yM øM tM !M a' aM' i' iM' u' uM' e' eM' o'
2. Program Listing

10 REM *Generative Phonological Derivations*
20 CLEAR 20:CLS
30 DEFINT A-Z
40 DIM SG$(3), SE(3), WB(3), CS(3), OB(3), SY(3)
50 DIM CT(3), AN(3), CO(3), VO(3), HI(3), LO(3)
60 DIM BA(3), RN(3), TN(3), NA(3), ST(3)
70 DATA #, -1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
80 DATA +, -1, 1, 1, 1, -1, 1, 1, -1, 1, -1, 1, -1, 1, -1, 1
90 DATA a, -1, -1, -1, 1, -1, -1, 1, 1, -1, 1, -1, 1, -1, 1, -1, 1
100 DATA i, -1, -1, -1, 1, -1, -1, 1, 1, -1, 1, -1, 1, -1, 1, -1, 1
110 DATA u, -1, -1, -1, 1, -1, -1, 1, 1, -1, 1, -1, 1, -1, 1, -1, 1
120 DATA e, -1, -1, -1, 1, -1, -1, 1, 1, -1, 1, -1, 1, -1, 1, -1, 1
130 DATA o, -1, -1, -1, 1, -1, -1, 1, 1, -1, 1, -1, 1, -1, 1, -1, 1
140 DATA aM, -1, -1, -1, 1, -1, -1, 1, 1, -1, 1, -1, 1, -1, 1, -1, 1
150 DATA iM, -1, -1, -1, 1, -1, -1, 1, 1, -1, 1, -1, 1, -1, 1, -1, 1
160 DATA uM, -1, -1, -1, 1, -1, -1, 1, 1, -1, 1, -1, 1, -1, 1, -1, 1
170 DATA eM, -1, -1, -1, 1, -1, -1, 1, 1, -1, 1, -1, 1, -1, 1, -1, 1
180 DATA oM, -1, -1, -1, 1, -1, -1, 1, 1, -1, 1, -1, 1, -1, 1, -1, 1
190 DATA @M, -1, -1, -1, 1, -1, -1, 1, 1, -1, 1, -1, 1, -1, 1, -1, 1
200 DATA yM, -1, -1, -1, 1, -1, -1, 1, 1, -1, 1, -1, 1, -1, 1, -1, 1
210 DATA ¢M, -1, -1, -1, 1, -1, -1, 1, 1, -1, 1, -1, 1, -1, 1, -1, 1
220 DATA IM, -1, -1, -1, 1, -1, -1, 1, 1, -1, 1, -1, 1, -1, 1, -1, 1
Becker - 9

200 T$="Underlying Representation"
210 PRINT "Please enter an UNDERLYING REPRESENTATION, using slashes (/')s"
220 PRINT "to separate each segment & boundary from its neighbors"
230 INPUT UR$
240 W$=#/#/+UR$+
250 T=0;A=1;G=26
260 FOR B=1 TO 4
270 IF A+B-1>LEN(W$) GOTO 380
280 B$(B)=MID$(W$,A+B-1,1)
290 IF B$(8)=/1 GOTO 320
300 C$=C$+B$(B)
310 NEXT B
320 T=T+1
330 SG$(T)=C$
340 A=A+B
350 C$=""
360 GOTO 260
370 GOTO 260
380 GOSUB 12000
390 PRINT STRING$(G-LEN(T$),",")+T$;": ",Z$
400 W$=Z$
700 FOR H=1 TO 17
710 ON H GOTO 2500,2100,1000,1100,1200,1300,1400,1500,1600,1700,1800,1900,
    2000,2200,2300,2400,2600
720 PRINT STRING$(G-LEN(T$)," ")+""+T$; "
730 IF Z$ <> W$ PRINT Z$ ELSE PRINT STRING$(LEN(Z$),"-"
740 W$=Z$
750 NEXT H
760 GOTO 9990
990 REM *Phonological Rules*
1000 T$="Obstruent Dissimilation"
1010 FOR K=1 TO T
1015 IF SG$(K+1)="+" THEN L=2 ELSE L=1
1020 IF OB(K)=1 AND OB(K+L)=1 GOSUB 1050
1030 NEXT K
1040 GOTO 720
1050 CT(K)=CT(K+L)*-1
1050 GOSUB 15000
1070 RETURN
1100 T$="High Back vowel Lowering"
1110 FOR K=1 TO T
1112 M=0
1115 FOR L=1 TO 2
1117 IF SG$(K+L+M)="+" THEN M=M+1
1118 IF SY(K)=1 AND SY(K+L+M)=1 GOTO 1140
1120 IF SY(K)=1 AND SY(K+L+M)=1 GOTO 1150
1130 NEXT L
1140 NEXT K
1145 GOTO 720
1150 IF LO(K)=-1 AND BA(K)=1 AND LO(K+L+M)=1 AND BA(K+L+M)=1 GOTO 1170
1160 GOTO 1140
1170 HI(K)=-1
1180 GOSUB 15000
1190 GOTO 1140
1200 T$="Front Vowel Rounding"
1210 FOR K=1 TO T
1215 IF SG$(K-1)="+" THEN L=2 ELSE L=1
1220 IF SY(K)=1 AND LO(K)=-1 AND BA(K)=-1 AND LO(K+L)=1 AND CO(K+L)=1 AND CO(K+L+M)=-1 GOSUB 1250
1230 NEXT K
1240 GOTO 720
1250 RN(K)=1
1250 GOSUB 15000
1270 RETURN
1300 T$="Consonant Voicing"
1310 FOR K=1 TO T
1315 IF SG$(K-1)="+" THEN L=2 ELSE L=1
1316 IF SG$(K+1)="+" THEN M=2 ELSE M=1
1320 IF VO(K-L)=1 AND VO(K+M)=1 AND CS(K)=1 GOSUB 1350
1330 NEXT K
1340 GOTO 720
1350 VO(K)=1
1360 GOSUB 15000
1370 RETURN
1400 T$="Stress Placement"
1405 C=0:FOR V=1 TO T
1406 IF SG$(V)="#" GOSUB 1405
1410 NEXT V
1420 IF SG$(K)="#" THEN K=0
1407 NEXT V
1408 GOTO 1412
1409 C=C+1
1410 C(C)=V
1411 RETURN
1412 FOR PC=1 TO C-1
1413 FOR K=C(PC+1) TO C(PC) STEP -1
1414 IF SG$(K+l)="4" THEN L=2 ELSE L=1
1415 IF SG$(K+L+1)="4" THEN M=2 ELSE M=1
1420 IF SY(K)=1 AND SY(K+L)=-1 AND SY(K+L+M) <> 1 GOTO 1470
1430 NEXT K
1440 FOR K=C(PC) TO C(PC+1)
1450 IF SY(K)=1 GOTO 1470
1460 NEXT K
1465 GOTO 1490
1470 ST(K)=1
1480 GOSUB 15000
1490 NEXT PC
1495 GOTO 720
1500 T$="h-Deletion"
1510 FOR K=1 TO T
1515 IF SG$(K+1)="4" THEN L=2 ELSE L=1
1520 IF SY(K)=-1 AND CS(K+L)=1 AND VO(K+L)=-1 AND CT(K+L)=1 GOTO 1550
1530 NEXT K
1540 GOTO 720
1550 T=T-1
1551 FOR X=K+L TO T
1552 SG$(X)=SG$(X+l)
1553 NEXT X
1560 GOSUB 11000
1570 GOTO 1510
1600 T$="Word-final Obst. Devoicing"
1610 FOR K=1 TO T
1620 IF OB(K)=1 AND WB(K+1)=1 GOSUB 1650
1630 NEXT K
1640 GOTO 720
1650 VO(K)=-1
1660 GOSUB 15000
1670 RETURN
1700 T$="Geminate Simplification"
1710 FOR K=1 TO T
1715 IF SG$(K+1)="4" THEN L=2 ELSE L=1
1720 IF SY(K)=1 AND SY(K+L)=1 AND HI(K)=HI(K+L) AND LO(K)=LO(K+L) AND BA(K)=BA(K+L) AND RN(K)=RN(K+L) GOTO 1750
1730 NEXT K
1740 GOTO 720
1750 IF ST(K+L)=1 THEN SG$(K)=SG$(K+L)
1751 T=T-1
1752 FOR X=K+L TO T
1753 SG$(X)=SG$(X+1)
1754 NEXT X
1760 GOSUB 11000
1770 GOTO 1710
1800 T$="Metathesis"
1810 FOR K=1 TO T
1815 IF SG$(K+1)="+' THEN L=2 ELSE L=1
1820 IF CS(K)=1 AND OB(K)=-1 AND NA(K)=-1 AND OB(K+L)=1 GOTO 1850
1830 NEXT K
1840 GOTO 720
1850 M$=SG$(K)
1851 SG$(K)=SG$(K+L)
1852 SG$(K+L)=M$
1860 GOSUB 12000
1870 GOTO 1810
1900 T$="Nasal Assimilation"
1910 FOR K=1 TO T
1915 IF SG$(K+1)="+' THEN L=2 ELSE L=1
1920 IF SY(K)=-1 AND NA(K)=1 AND CS(K+L)=1 AND OB(K+L)=1 AND CT(K+L)
2100 =-1 GOSUB 1950
1930 NEXT K
1940 GOTO 720
1950 AN(K)=AN(K+L)
1960 GO$UB 15000
1970 RETURN
2000 T$="i-Umlaut"
2010 FOR K=1 TO T
2012 M=0
2015 FOR L=1 TO 3
2017 IF SG$(K+L+M)="+' THEN M=M+1
2018 IF SG$(K+L+M)="#" GOTO 2040
2020 IF SY(K)=1 AND SY(K+L+M)=1 GOTO 2050
2030 NEXT L
2040 NEXT K
2045 GOTO 720
2050 IF HI(K+L+M)=1 AND BA(K+L+M)=-1 GOTO 2070
2060 GOTO 2040
2070 BA(K)=-1
2080 GOSUB 15000
2090 GOTO 2040
2100 T$="Alpha-Switching"
2110 FOR K=1 TO T
2120 IF SY(K)=1 AND BA(K)=1 AND RN(K)=1 GOSUB 2150
2130 NEXT K
2140 GOTO 720
2150 BA(K)=HI(K)*-1
2160 GOSUB 15000
2170 RETURN
2200 T$="Nasalization"
2210 FOR K=1 TO T
2220 IF SY(K)=1 AND NA(K+1)=1 AND SY(K+2)=-1 GOTO 2250
2230 NEXT K
224¢ GOTO 72¢
225¢ NA(K)=1
2251 GOSUB 15¢¢¢
2252 T=T-1
2253 FOR X=K+1 TO T
2254 SG$(X)=SG$(X+1)
2255 NEXT X
225¢ GOSUB 11¢¢¢
226¢ GOTO 221¢
23¢¢
231¢ T$="Vowel Insertion"
231¢ FOR K=1 TO T
2315 IF SE(K+1)=—1 THEN L=2 ELSE L=1
232¢ IF OB(K)=1 AND OB(K+L)=1 GOTO 235¢
233¢ NEXT K
234¢ GOTO 72¢
235¢ FOR X=T TO K+l STEP —1
2351 SG$(X+1)=SG$(X)
2352 NEXT X
2353 SG$(K+1)="a"
2354 T=T+1
236¢ GOSUB 12¢¢¢
237¢ GOTO 231¢
24¢¢
240¢ T$="Nasalized Vowel Laxing"
241¢ FOR K=1 TO T
242¢ IF SY(K)=1 AND NA(K)=1 AND BA(K)=—1 AND LO(K)=—1 GOSUB 245¢
243¢ NEXT K
244¢ GOTO 72¢
245¢ TN(K)=—1
2451 IF ST(K)=—1 THEN HI(K)=—1
246¢ GOSUB 15¢¢¢
247¢ RETURN
25¢¢
250¢ T$="Palatalization"
251¢ FOR K=1 TO T
2515 IF SG$(K+1)="*" THEN L=2 ELSE L=1
252¢ IF CS(K)=1 AND OB(K)=1 AND CT(K)=1 AND (AN(K)=—1 OR CO(K)=1)
253¢ AND SY(K+L)=1 AND HI(K+L)=1 AND BA(K+L)=—1 GOSUB 255¢
253¢ NEXT K
254¢ GOTO 72¢
255¢ AN(K)=—1
2551 CO(K)=1
2552 HI(K)=1
2553 BA(K)=—1
256¢ GOSUB 15¢¢¢
257¢ RETURN
26¢¢
260¢ T$="Glottal Stop Insertion"
261¢ FOR K=1 TO T
262¢ IF SY(K)=1 AND WB(K+1)=1 AND SY(K+2)=1 GOTO 265¢
263¢ NEXT K
264¢ GOTO 72¢
265¢ FOR X=T TO K+2 STEP —1
2651 SG$(X+1)=SG$(X)
2652 NEXT X
2653 SG$(K+2)="?"
2654 T=T+1
2660 GOSUB 12000
2670 GOTO 2610
9990 T$="Phonetic Output”: PRINT STRING$(G-LEN(T$)," ")+T$;" : ;
9991 FOR X=1 TO T
9992 IF SG$(X)="+" GOTO 9995
9993 IF SG$(X)="#" THEN SG$(X)=" ";
9994 PRINT SG$(X);
9995 NEXT X
9996 END
11000 FOR K=1 TO T
11010 IF SG$(K)="+" AND (SG$(K+1)="+" OR SG$(K+1)="#") GOTO 1140
11020 NEXT K
11030 GOTO 12000
11040 T=T-1
11050 FOR X=K TO T
11060 SG$(X)=SG$(X+1)
11070 NEXT X
11080 GOTO 11000
12000 FOR K=1 TO T
12010 READ A$, SE, WB, CS, OB, SY, CT, AN, CO, VO, HI, LO, BA, RN, TN, NA, ST
12020 IF A$<> SG$(K) GOTO 12000
12030 SE(K)=SE
12040 WB(K)=WB
12050 CS(K)=CS
12060 OB(K)=OB
12070 SY(K)=SY
12080 CT(K)=CT
12090 AN(K)=AN
12100 CO(K)=CO
12110 VO(K)=VO
12120 HI(K)=HI
12130 LO(K)=LO
12140 BA(K)=BA
12150 RN(K)=RN
12160 TN(K)=TN
12170 NA(K)=NA
12180 ST(K)=ST
12190 RESTORE
12200 NEXT K
13000 Z$=""
13010 FOR X=1 TO T
13020 Z$=Z$+SG$(X)'
13030 NEXT X
13040 RETURN
15000 READ A$, SE, WB, CS, OB, SY, CT, AN, CO, VO, HI, LO, BA, RN, TN, NA, ST
15010 IF SE<> SE(K) GOTO 15000
15020 IF WB<> WB(K) GOTO 15000
15030 IF ST<> ST(K) GOTO 15000
15040 IF NA<> NA(K) GOTO 15000
15050 IF CS<> CS(K) GOTO 15000
15060 IF OB<>CT(K) GOTO 15000
15070 IF SY<>SY(K) GOTO 15000
15080 IF CT<>CT(K) GOTO 15000
15090 IF AN<>AN(K) GOTO 15000
15100 IF CO<>CO(K) GOTO 15000
15110 IF VO<>VO(K) GOTO 15000
15120 IF HI<>HI(K) GOTO 15000
15130 IF LO<>LO(K) GOTO 15000
15140 IF BA<>BA(K) GOTO 15000
15150 IF RN<>RN(K) GOTO 15000
15160 IF TN<>TN(K) GOTO 15000
15170 SG$(K)=A$
15180 RESTORE
15190 GOTO 13000
3. **Some Sample Runs**

The derivations presented in (20)-(28) below demonstrate how the program presented in Section 2 elicits hypothetical underlying representations from the program user and then maps them step by step into their systematic phonetic counterparts through the application of rules (1)-(17) in the linear order (16)-(12)-(1)-(2)-(3)-(4)-(5)-(6)-(7)-(8)-(9)-(10)-(11)-(13)-(14)-(15)-(17).
(20) Please enter an UNDERLYING REPRESENTATION, using slashes (/´s) to separate each segment & boundary from its neighbors?

? s/t/k/o/+t/a/b/e/+SH/u/+u/n/d/a

Underlying Representation: ##siko+tabe+SHu+unda##

Palatalization: ##SHiko+tabe+SHu+unda##

Alpha-Switching: ##SHiku+tabe+SHy+unda##

Obstruent Dissimilation: -------------------------------------

High Back Vowel Lowering: ##SHiko+tabe+SHy+unda##

Front Vowel Rounding: ##SHiko+tabØ+SHy+unda##

Consonant Voicing: ##SHigo+dabØ+ZHy+ynda##

Stress Placement: ##SHigo+dabØ+ZHy+y'nda##

h-Deletion: -------------------------------

Word-final Obst. Devoicing: -------------------------------

Geminate Simplification: ##SHigo+dabØ+ZHy'nda##

Metathesis: -------------------------------------

Nasal Assimilation: -------------------------------

i-Umlaut: -------------------------------

Nasalization: -------------------------------

Vowel Insertion: -------------------------------

Nasalized Vowel Laxing: -------------------------------

Glottal Stop Insertion: -------------------------------

Phonetic Output: SHigodabØZHy'nda

(21) Please enter an UNDERLYING REPRESENTATION, using slashes (/´s) to separate each segment & boundary from its neighbors?

? s/o/t/t/+r/i/t/#s/y/b/o/t/+r/i/t/i

Underlying Representation: ##sott+ri#sybot+riti##

Palatalization: ##sott+ri#SHybot+riti##

Alpha-Switching: ##sutt+ri#SHybut+riti##

Obstruent Dissimilation: ##sust+ri#SHybut+riti##

High Back Vowel Lowering: -------------------------------

Front Vowel Rounding: -------------------------------

Consonant Voicing: ##sust+ri#SHybud+ridi##

Stress Placement: ##su'st+ri#SHybu'd+ridi##

h-Deletion: -------------------------------

Word-final Obst. Devoicing: -------------------------------

Geminate Simplification: -------------------------------------

Metathesis: -------------------------------------

Nasal Assimilation: -------------------------------------

i-Umlaut: ##su'st+ri#SHyby'd+ridi##

Nasalization: -------------------------------------

Vowel Insertion: ##su'sat+ri#SHyby'd+ridi##

Nasalized Vowel Laxing: -------------------------------------

Glottal Stop Insertion: -------------------------------------

Phonetic Output: su'satri SHyby'dridi
(22) Please enter an UNDERLYING REPRESENTATION, using slashes (/’s) to separate each segment & boundary from its neighbors
s/a/m/b/#/k/o/l/+i/NG/t

Underlying Representation: ##samb#kol+iNGt##

Palatalization: ----------------------
Alpha-Switching: ##samb#kul+iNGt##
Obstruent Dissimilation: ----------------------
High Back Vowel Lowering: ----------------------
Front Vowel Rounding: ----------------------
Consonant Voicing: ----------------------
Stress Placement: ##sa’mb#kul+i’NGt##
h-Deletion: ----------------------
Word-final Obst., Devoicing: ##sa’mp#kul+i’NGt##

Geminate Simplification: ----------------------
Metathesis: ----------------------
Nasal Assimilation: ##sa’mp#kul+i’nt##
i-Umlaut: ##sa’mp#kyl+i’nt##
Nasalization: ##saM’p#kyl+iM’t##
Vowel Insertion: ##saM’pa#kyl+iM’t##
Nasalized Vowel Laxing: ##saM’pa#kyl+IM’t##
Glottal Stop Insertion: ----------------------
Phonetic Output: saM’pa kylIM’t

(23) Please enter an UNDERLYING REPRESENTATION, using slashes (/’s) to separate each segment & boundary from its neighbors
s/o/f/+h/e/GH/i/#/a/p/i/n

Underlying Representation: ##sof+heGHi#apin##

Palatalization: ##sof+heZHi#apin##
Alpha-Switching: ##suf+heZHi#apin##
Obstruent Dissimilation: ##sup+heZHi#apin##
High Back Vowel Lowering: ----------------------
Front Vowel Rounding: ##sup+heZHi#apyn##
Consonant Voicing: ##sup+heZHi#aby’n##
Stress Placement: ##su’p+heZHi#aby’n##
h-Deletion: ##su’p+eZHi#aby’n##
Word-final Obst., Devoicing: ----------------------

Geminate Simplification: ----------------------
Metathesis: ----------------------
Nasal Assimilation: ##su’p+eZHi#@by’n##
i-Umlaut: ##su’p+eZHi#@by’n##
Nasalization: ----------------------
Vowel Insertion: ----------------------
Nasalized Vowel Laxing: ----------------------
Glottal Stop Insertion: ##su’p+eZHi#@by’n##
Phonetic Output: su´peZHi #@by’n
(24) Please enter an UNDERLYING REPRESENTATION, using slashes (/’s) to separate each segment & boundary from its neighbors:

? p/e/r+/k/o/m+/k/a/p/+t/i/d

Underlying Representation: ##per+kom+kap+tid##
Palatalization: ------------------
Alpha-Switching: ##per+kum+kap+tid##
Obstruent Dissimilation: ##per+kum+kaf+tid##
High Back Vowel Lowering: ------------------
Front Vowel Rounding: ##p¢r+kum+kaf+tid##
Consonant Voicing: ##p¢r+gum+gaf+ti’d##
Stress Placement: ##p¢r+gum+gaf+ti’d##
h-Deletion: ---------------
Word-final Obst. Devoicing: ##p¢r+gum+gaf+ti’t##
Geminate Simplification: ------------------
Metathesis: ##p¢g+rum+gaf+ti’t##
Nasal Assimilation: ##p¢g+ruNG+gaf+ti’t##
i-Umlaut: ##p¢g+ruNG+g@f+ti’t##
Nasalization: __________ , __ _
Vowel Insertion: ##p¢g+ruNG+g@fa+ti’t##
Nasalized Vowel Laxing: -----------------------------------------
Glottal Stop Insertion: -----------------------------------------
Phonetic Output: p¢gruNGg@fati’t

(25) Please enter an UNDERLYING REPRESENTATION, using slashes (/’s) to separate each segment & boundary from its neighbors:

? s/i/+i/#/o/n/d/i/l/+p/a/d

Underlying Representation: ##si+i#ondil+pad##
Palatalization: ##SHi+i#ondil+pad##
Alpha-Switching: ##SHi+i#undil+pad##
Obstruent Dissimilation: ------------------
High Back Vowel Lowering: ------------------
Front Vowel Rounding: ------------------
Consonant Voicing: ##SHi’+i#undil+bad##
Stress Placement: ##SHi’+i#undil+ba’d##
h-Deletion: ------------------
Word-final Obst. Devoicing: ##SHi’+i#undil+ba’t##
Geminate Simplification: ##SHi’#undil+ba’t##
Metathesis: ##SHi’#undib+la’t##
Nasal Assimilation: ------------------
i-Umlaut: ##SHi’#yndib+la’t##
Nasalization: ##SHi’#yMdib+la’t##
Vowel Insertion: ------------------
Nasalized Vowel Laxing: ------------------
Glottal Stop Insertion: ------------------
Phonetic Output: SHi’?OEMdibla’t
(26) Please enter an UNDERLYING REPRESENTATION, using slashes (/’s) to separate each segment & boundary from its neighbors?

? b/e/#/o/p/t/#/#/a/z/+/a/z/+/OEM/b

Underlying Representation: ##be#opi##ax+xOEMb##

Palatalization: ##be#opi##ax+SHi#OEMb##

Alpha-Switching: ##be#upi##ax+SHi#OEMb##

Obstruent Dissimilation: ##be#upi##ak+SHi#OEMb##

High Back Vowel Lowering: -----------------------------------

Front Vowel Rounding: ##b/0#upy##ak+SHi#OEMb##

Consonant Voicing: ##b/0#uby##ak+SHi#OEMb##

Stress Placement: ##b/0"u/by##ak+SHi#OEM"b##

h-Deletion: -----------------------------------------

Word-final Obst. Devoicing: ##b/0"u'by##ak+SHi#OEM"p##

Geminate Simplification: -------------------------------------

Metathesis: ---------------------------------------------

Nasal Assimilation: -------------------------------------

i-Umlaut: ##b/0"y'by##ak+SHi#OEM"p##

Nasalization: ------------------------------------------

Vowel Insertion: ##b/0"y'by##ak+SHi#OEM"p##

Nasalized Vowel Laxing: ----------------------------------

Glottal Stop Insertion: ----------------------------------

Phonetic Output: b•?y"by @'kaSHi ?OEM"p

(27) Please enter an UNDERLYING REPRESENTATION, using slashes (/’s) to separate each segment & boundary from its neighbors?

? b/a/d/+/b/i/d/#/b/o/d/#/#/b/e/t/+/o/NG/+b/4

Underlying Representation: ##bad+bid##bod##bet+OEM+bi##

Palatalization: ---------------------------------------

Alpha-Switching: ##bad+bid##bud##bet+uNG+bi##

Obstruent Dissimilation: ##baz+bid##bud##bet+uNG+bi##

High Back Vowel Lowering: ----------------------------------------

Front Vowel Rounding: ##baz+byd##bud##b0t+uNG+by##

Consonant Voicing: ##baz+byd##bud##b0d+uNG+by##

Stress Placement: ##baz+by'd##bud##b0t+uNG+by##

h-Deletion: -----------------------------------------

Word-final Obst. Devoicing: ##baz+by't##bud##b0d+uNG+by##

Geminate Simplification: -------------------------------------

Metathesis: ---------------------------------------------

Nasal Assimilation: -------------------------------------

i-Umlaut: ##baz+by't##bud##b0d+uNG+by##

Nasalization: ------------------------------------------

Vowel Insertion: ##baz+by't##bud##b0d+uNG+by##

Nasalized Vowel Laxing: ----------------------------------

Glottal Stop Insertion: ----------------------------------

Phonetic Output: b•?y"by @'kaSHi ?OEM"p

b•?y"by @'kaSHi ?OEM"p
(28) Please enter an UNDERLYING REPRESENTATION, using slashes (/\')s to separate each segment & boundary from its neighbors

? p/s/a/s+/h/t/#/i/p/+/n/a/m/+t/o/#/ZH/a/b

Underlying Representation: ##p!sas+hi!p!nam+to#ZHab##

Palatalization: -------------------------------

Alpha-Switching: ##p!sas+hi!p!nam+tu#ZHab##

Obstruent Dissimilation: ##p!sat+hi!p!nam+tu#ZHab##

High Back Vowel Lowering: ##p+sat+hi!p!nam+tu#ZHab##

Front Vowel Rounding: -------------------------------

Consonant Voicing: ##pt+zat+hi!b!nam+du#ZHab##

Stress Placement: ##pt+zat+hi!b!na`m+du#ZHab`b##

h-Deletion: ##pt+zat+i!b!na`m+du#ZHab`b##

Word-final Obst. Devoicing: ##pt+zat+i!b!na`m+du#ZHab`p##

Geminate Simplification: -------------------------------

Metathesis: -------------------------------

Nasal Assimilation: ##pt+zat+i!b!na`n+du#ZHa`p##

i-Umlaut: ##pt+zat+i!b!na`n+du#ZHa`p##

Vowel Insertion: -------------------------------

Nasalized Vowel Laxing: -------------------------------

Glottal Stop Insertion: -------------------------------

Phonetic Output: pt+zat`t+i?ib!na`ndu ZHa`p
4. Some technical notes

The program listed in Section 2 requires approximately 16K of memory. How it operates is outlined below:

LINE(S) FUNCTION

<table>
<thead>
<tr>
<th>10-172</th>
<th>Preliminaries</th>
</tr>
</thead>
<tbody>
<tr>
<td>20</td>
<td>Reserves 200 bytes of memory space for strings and clears the CRT (or monitor) screen.</td>
</tr>
<tr>
<td>30</td>
<td>Defines all non-string variables as integers.</td>
</tr>
<tr>
<td>40-60</td>
<td>Dimension 17 arrays so that the strings upon which the phonological rules operate may contain up to 30 segment and boundary elements. SG$(n)$ will store the symbol (or symbols) corresponding to the n-th segment or boundary in the string, while SE(n), WB(n), CS(n), OB(n), etc. will store the specifications for the features 'segment', 'word boundary', 'consonantal', 'obstruent', etc. associated with that segment or boundary. The '+'s, '-'s, and blanks of classificatory matrix (19) are replaced in this program by 1's, -1's, and 0's, respectively.</td>
</tr>
</tbody>
</table>

| 100-172 | List each segment and boundary found in (19) together with its specifications for the features SE, WB, CS, OB, SY, CT, AN, CO, VO, HI, LO, BA, RN, TN, NA, ST (in that order). |

<table>
<thead>
<tr>
<th>200-400</th>
<th>Obtaining the first line of the derivation</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Lets T$ equal the title of the first line of the derivation.</td>
</tr>
<tr>
<td>210-230</td>
<td>Elicit a hypothetical underlying representation (UR) from the program user.</td>
</tr>
<tr>
<td>240</td>
<td>Adds a pair of #'s to the beginning and end of the UR.</td>
</tr>
<tr>
<td>250</td>
<td>Initializes the variables T, A, and G. The most important of these variables is T, for it is used throughout the program to keep track of the number of segments and boundaries found in the utterance at any given stage in the derivation.</td>
</tr>
<tr>
<td>260-370</td>
<td>Scan the UR from left to right, assigning each segment or boundary encountered to the variable SG$(T)$, where T indicates the distance of the segment or boundary in question from the beginning of the UR.</td>
</tr>
</tbody>
</table>
Runs a subroutine consisting of lines 12000-13040. This subroutine compares each segment and boundary of the UR with the first element of data lines 100-172 until it finds a match. It then assigns the feature specifications found on that line to integer variables SE(K), WB(K), CS(K), OB(K), etc. whose subscript K corresponds to that of the string variable SG$(K)$ associated with the segment or boundary in question. Lines 13000-13040 concatenate all of the segments and boundaries that constitute the UR and then store the resulting string temporarily in the memory location labeled Z$.

Prints the first line of the derivation, which consists of the title 'Underlying Representation' followed by a colon, a space, and the underlying representation assigned to Z$.

Equates W$ with Z$.

Ordering and applying the rules

Begins a loop in which the rules are applied and their outputs printed.

Specifies the order in which the 17 rules beginning on every hundredth line from 1000 to 2600 are applied. It does this by causing the computer to branch, in the desired sequence, to those lines.

Contain the 17 phonological rules, each of which operates as follows: It scans the segments and boundaries of string Z$ and performs its structural change upon those substrings of Z$ which meet its structural description. Each time it applies, the rule causes the computer to execute one of the following subroutines:

Subroutine 11000

Operating after the deletion of a segment, this subroutine eliminates any superfluous morpheme boundaries and then assigns values to the variables SE(K), WB(K), CS(K), OB(K), etc. that characterize the segment or boundary now occupying each 'slot' K in the shortened string.

Subroutine 12000

Operating after the insertion of a segment, this subroutine assigns values to the variables SE(K), WB(K), CS(K), OB(K), etc. that characterize the segment or boundary now occupying each 'slot' K of the expanded string.
Subroutine 15000

Operating after the structural change of a feature-switching rule, this subroutine compares the feature specifications of the altered segment with those listed on the data lines until it finds a segment with precisely the same feature values as the altered segment. It then assigns the symbol (or symbols) representing that segment to the string variable SG$(K)$ that corresponds to the altered segment.

Note:
All three subroutines end with lines 13000-13040, which link together all of the segments and boundaries of the modified utterance and assign them to the variable Z$, replacing the string formerly stored there.

720 Prints the title of the rule just processed, lining up the end of that title with the final letter of the title printed on the preceding line of the derivation. A colon and a blank space are printed after the title.

730 Prints the output of the rule just processed immediately after the space left by the print statement in line 720, unless the rule has brought about no change, in which case the rule's output is replaced by a series of hyphens equal in length to the form printed on the preceding line.

740 Equates W$ with Z$.

750 Completes the loop begun on line 700. Since the iteration variable H in line 700 equals '1 to 17', the computer must make seventeen passes through the loop, one for each phonological rule in the program.

760 Causes the computer to jump to line 9990.

9990-9996 Obtaining the last line of the derivation

9990 Prints the first half of the final line of the derivation, which consists of the title 'Phonetic Output', a colon, and a space.

9991 Begins a loop with as many iterations as there are boundaries and segments in the output of the last rule in the ordered set.

9992 Deletes all morpheme boundaries.

9993 Replaces all #'s with blank spaces.

9994 Supplies the second half of the last line of the derivation by printing the phonetic output one segment (or space) at a time.

9995 Completes the loop begun on line 9991.

9996 Terminates the run.
5. Pedagogical Applications

The most obvious pedagogical applications for the program presented in this article lie in the realm of computer-assisted instruction (CAI). Part of any introductory course in phonology must be devoted to teaching students how to read and apply ordered phonological rules. In the classroom, however, the amount of time that can be spent on the development of these basic skills is normally less than optimal. By contrast, a program like that in Section 2 allows students to practice both tasks to their hearts' content. Their computer-assisted work can take the following forms:

(29) a. Seated at a home computer (or the terminal of a larger data processing system) and provided with a hard copy of the rules of the program that has been loaded into the computer's memory, the students can be asked to input different underlying representations and to try to anticipate how they will be affected by each rule before the computer prints the answer on the next line of the ensuing derivation.

b. When the students run out of UR's, they can continue the exercise by making up new ones and inputting them.

c. As a further challenge, the students can be given a list of systematic phonetic representations for which they must attempt to supply all possible underlying representations. They can check their work by typing in their answers and seeing whether the resulting derivations yield the expected surface forms.

d. An exercise of a different sort consists in giving the students a set of UR's and a corresponding set of phonetic forms and asking them to determine the sequence in which the rules must be applied in order to convert the former into the latter. They can easily verify their answers by changing the order of the line numbers listed after the GOTO statement on line 710 and then running derivations with the modified program.

The program presented in this article is, of course, only intended as a sample. Teachers can change the number and complexity of the rules contained in it as they see fit in order to provide their pupils with the desired amount of practice.

The educational value of the program in Section 2 is not limited to its CAI applications. In addition to functioning as an extension of the instructor, the program can serve as the object of much valuable experimentation and study. In an age when computers seem destined to play ever expanding roles in virtually all spheres of human endeavor, including linguistic research, conscientious teachers of our subject should seize
every possible opportunity to cultivate their students' programming talents. Phonologists, for example, could teach their students not only how to write rules describing a particular body of phonetic data, but also how to incorporate those rules into simple computer programs for which a program like that in Section 2 might serve as a model. Before attempting to write rule-testing programs of their own, however, the students could 'get their feet wet' by analyzing and modifying the model program. A number of possible assignments come to mind:

(30) a. The students could be quizzed about the functions of various lines in the program.

b. They could be asked to suggest changes in the program that would increase the speed at which its rules are processed.

c. They could be required to add new rules or to modify existing ones. For example, they could be asked to ... 
   i. restrict the domain of Vowel Insertion (lines 2300-2370 of the program) to the word, 
   ii. transform the rule of Obstruent Dissimilation (lines 1000-1070) into one of Obstruent Assimilation that makes contiguous obstruents agree in their specifications for the feature 'continuant', or
   iii. revise the Metathesis rule (lines 1800-1870) so that it applies only to liquids and obstruents separated by a morpheme boundary.

d. They could be instructed to expand the inventory of segments and boundaries used in the program or to alter the set of distinctive features that define them. They might, for example, be told to add interdental fricatives to the set of segments classified in the data lines.

e. On a more advanced level, they might be assigned the task of restructuring the program to support simultaneous or random-sequential rule application, derivational or surface-phonetic constraints, global rules, diacritic features, or various other theoretical concepts and constructs.

f. Students with a background in computer science might even be given the assignment of translating the BASIC program into another computer language and assessing the relative merits of BASIC and that language as frameworks for phonological description.
Incorporating a component of computer programming into elementary linguistics courses would, in my opinion, have a number of beneficial results. It would better prepare those students who make linguistics their career for the linguistic science of the future. It would benefit those students who do not remain in the field by improving their ability to work with computers, an ability they could put to good use in many different professions. Finally, it would help to dispel the ivory tower image of linguistics and, in so doing, perhaps attract students to our discipline who might otherwise have shied away from it.