A Note on Segment Inventories, Redundancy Conditions and A-Rules

Thilo C. Schadeberg
Department of African Linguistics, University of Leiden

1. Introduction

A curious feature of the SPE model of phonology (and also of other models) is that it provides no systematic place for an inventory of segments within the description of a particular language. Such tables of segments as are usually given have no formal status and are, from a theoretical point of view, merely appendices for convenient reference. It appears that the only principled way of finding out about the segment inventory is to inspect all phonological representations and to collect all the different types of segment matrices. This is counter-intuitive because it fails to account for the speaker's knowledge of what segments his language has or doesn't have.

Redundancy Conditions (RC) are supposed to account for redundant feature specifications. Stanley (1967) argued that RCs operate on fully specified segments and segment sequences; in particular he demanded that RCs should neither change nor fill in feature specifications. Stanley's proposal has not become generally accepted, at least not in African linguistics, and not always for good reasons.

It has long been noted that identical restrictions on segment and segment sequence structures may hold within the morpheme and across morpheme boundaries. In the SPE model such restrictions must be stated twice: as a RC in the lexicon, and as a P-Rule in the phonology proper. Stewart (1983) has solved this deficiency of RCs by extending their domain from the morpheme to the word; it would be interesting to find out whether his Word Structure Conditions supersede Morpheme Structure Conditions or whether both types are needed because they express different significant generalizations. In the same article, Stewart also introduces the device of "Associated Automatic Rules". These A-Rules state what happens when inadmissible (segments or) segment sequences arise in the course of morpheme concatenation.

Below I shall suggest that redundancy conditions, properly interpreted (section 2) and formulated (section 3), serve as a definition of the segment inventory (section 4). Prenasalized consonants (in Bantu) provide an
illustration of how sequential RCs and A-Rules interact to characterize "underspecified" segments (section 5). Finally, the hope is expressed (section 6) that phonological description will gain rather than lose when the goal of a redundancy-free representation is abandoned.

2. The First Function of Redundancy Conditions

The most obvious function of RCs is to make statements about redundancies in the feature specifications of segments and segment sequences. Such redundancies exist only with reference to a system. Two systems are involved: the feature system and the phonological system of a particular language. The feature system may be defined such that no segment can be both [+high] and [+low]; hence [+high, -low] and [-high, +low] segments have one redundant feature specification. If the feature system is held to be universal then such redundancies are also universal. A particular language may have no [-back, +round] vowels, then for that language [+back, +round] and [-back, -round] vowels have one redundant feature specification (leaving aside any major class features).

In the literature one frequently finds that RCs are given another, derived function: they are used to "save" particular feature specifications. This is usually presented in a table of segments where "redundant" specifications are either left blank or put in parantheses. There are several problems with this—in addition to the possible illicit use of blanks as pointed out by Stanley (1967).

First, there is no unique solution to the problem of defining the set of necessary RCs as long as the aim is a redundancy-free characterization of all segments in the system.

Second, given the same aim, it would be wrong to formulate all true redundancies because the remaining "non-redundant" feature specifications would no longer be sufficient to distinguish all segments. This has been experienced by Kuperus (1985:28): "The decision as to when sufficient SgSC's and logical implications have been formulated is a purely pragmatic one: one stops when to continue would be to analyse segments away."

Third, if RCs are "applied" to mark certain feature values as redundant then the order in which this is done becomes crucial.

These three points are, of course, interrelated. They will be illustrated by using a simple five-vowel system, and stating all redundancies in the form of negative conditions (for reasons I shall explain below).
Table 1

<table>
<thead>
<tr>
<th></th>
<th>i</th>
<th>e</th>
<th>o</th>
<th>o</th>
<th>u</th>
</tr>
</thead>
<tbody>
<tr>
<td>high</td>
<td>+</td>
<td>-</td>
<td>(-)</td>
<td></td>
<td>+</td>
</tr>
<tr>
<td>low</td>
<td>(-)</td>
<td>(-)</td>
<td>&lt;+&gt;</td>
<td>(-)</td>
<td>(-)</td>
</tr>
<tr>
<td>back</td>
<td>&lt;-&gt;</td>
<td>&lt;-&gt;</td>
<td>(+)</td>
<td>(+)</td>
<td>(+)</td>
</tr>
<tr>
<td>round</td>
<td>(-)</td>
<td>(-)</td>
<td>(-)</td>
<td>&lt;+</td>
<td>&lt;+</td>
</tr>
</tbody>
</table>

Feature specifications in parantheses are defined as redundant by the following four RCs:

(1) NOT \[+hi\] (2) NOT \[-lo\] (3) NOT \[+lo\] (4) NOT \[-ra\]

There is, however, a fifth RC which is equally true. RC5 defines the feature specifications given in Table 1 in angled brackets as redundant.

(5) NOT \[-lo\] \[+ba\] \[-rd\]

Given RC1 through RC4, RC5 cannot be stated if RCs are to retain their capability of "filling in" blanks. Note that there is nothing wrong with RC5 as such. RC5 becomes quite acceptable if we order (!) it before RC2. In that case, however, it becomes necessary not to state RC2 through RC4. Thus, our simple example shows that if we want to use RCs for filling in blanks we must refrain from stating all redundancies of the system lest we end up with undistinguishable segments. It further demonstrates that several alternatives exist as to which RCs may be selected, and this in turn signals that we do in fact take account of the order of our RCs.

We have already observed that the discovery procedure and the theoretical status of such a table of segments are dubious. We must add now that the content of this table is ill-defined once segment sequence structure conditions are admitted as statements about redundancy. For example, restriction on consonant clusters in English bring about that [s] is more highly redundant ("needs less feature specifications" - ?) in the word [Split] than in the word [Sit]; should the table contain two different "non-redundant" segments [s]?

The obvious answer to these problems is that it is wrong to think that some specifications are redundant and others are not. Rather, redundancy is no more
and no less than a property of information that is represented in a system. It is neither useful nor possible to convert a redundant representation into a non-redundant one using the same system.

3. The Form or Redundancy Conditions

Three formal kinds of RCs have been proposed: positive, negative and implicational ("if-then") conditions. Positive conditions have been set up for the description of segment sequences, in particular to account for admissible syllable structures. Since I prefer to deal with syllables in an entirely different way (i.e., some version of CV-phonology, see Clements and Keyser 1983), I have no need for this type of positive RCs.

Positive conditions could also be used to describe equivalence. Suppose we have a three-vowel system [i - a - u]. In this system we could say: a vowel is high if it is not low, AND, a vowel is low if it is not high. This might be written as a positive RC: [ahi, -alo]. Furthermore, positive RCs could express single, unconnected statements. If a language has no glottalized segments we could write a positive RC: [-glottal restriction]. However, for every positive RC there is an equivalent, equally concise and readable negative RC. Our two examples can be rewritten as "NOT [ahi, alo]" and "NOT [+ glottal restriction]", respectively. Since the inverse is not true, i.e., not every negative RC can be rewritten as a positive RC, I suggest that positive conditions are a superfluous device.

Implicational conditions are the most widely used. The reason is, I suspect, that they most closely resemble the form of phonological rules and make it easy to think of RCs as filling in or providing so-called redundant specifications. It is interesting to note that Stanley (1967) discusses the need for positive and negative conditions but finds it unnecessary to give a parallel justification for implicational conditions.

Implicational conditions have the disadvantage that for any implication there exists a counter-implication, and there is no way to choose between the two equivalent formulations. Cook (1985:168f.) attempts to turn this property of implicational conditions into a useful descriptive device. In his framework, RCs fill in blanks, and this process is claimed to be made "conceptually simpler and more straightforward" by stating RCs as one-way If-Then conditions, some of which are each other's counter-implications. Thus, by not being bound to
state ALL counter-implications, he increases control over which redundancy statements are wanted and which are not. This is, I believe, in conflict with the logical nature and the function of RCs.

Implicational conditions are unnecessary since any implication with counter-implication can be reformulated unambiguously as a negative condition.

Negative conditions formally highlight the difference between RCs and A-Rules, the former expressing an unacceptable feature combination, and the latter the strategy for repairing impending violations. Negative conditions also closely mirror non-formal but precise paraphrases of specific cases, e.g. "no segment can be both [+high] and [+low]", or "there are no front rounded vowels in language X". I therefore suggest that negative conditions are the only useful and legitimate form of RCs.

4. The Second Function of Redundancy Conditions

We now return to the question of the language specific segment inventory. Suppose there is a universal set of features. We can now demand that (negative) RCs must exclude all segments that are logically possible constructs within this universal feature system but do not occur in a particular language. (There will be a subset of universally non-occurring segments.) In this way we will at the same time define the members of a particular sound system and describe all redundancies operating within it.

In Table 1 we have seen a five-vowel system described with the use of four distinctive features. Within this feature sub-system we could distinguish $2^4 = 16$ different vowels. They are listed and defined in the upper part of Table 2. In the lower part, the x's indicate that a particular segment is excluded by the respective RC as formulated above.

The last line of Table 2 shows the five vowels that pass the filter of all five RCs. Note that all five RCs are needed to define the five-vowel system; if for instance RC5 were not given then [u] and [y] would not be excluded. Demanding that the set of RCs should define the segment inventory is incompatible with using RCs for filling in blanks but makes it possible to state redundancies exhaustively, uniquely and truly unordered. (In fact, "uniquely" may be too strong a claim, but if competing solutions were possible they would at least be equivalent.)
Many Bantu languages have a set of prenasalized consonants such as [mb nd nj ng]. The following restrictions (redundancies) frequently apply to this set:

(a) The nasal must be homorganic with the following obstruent.
(b) The obstruent must be voiced.
(c) There are no other sequences of consonants.

This can easily be expressed in three negative sequential RCs; the feature [place] in RC(a) may be thought of as a bundle of several place features.

\[
\begin{align*}
\text{RC}(a) & \quad \text{NOT} \quad \begin{bmatrix} +\text{cons} \\ -\text{place} \end{bmatrix} \begin{bmatrix} +\text{cons} \end{bmatrix} \\
\text{RC}(b) & \quad \text{NOT} \quad [+\text{cons}] [+\text{cons}] \\
\text{RC}(c) & \quad \text{NOT} \quad [+\text{cons}] [-\text{nas}] \\
\end{align*}
\]

Leaving aside details that are best described with reference to the syllabic structure, these three RCs are characteristic for the sound patterns of, for instance, Swahili, Nyakyusa and Umbundu. All three languages have the same
A-Rule associated with RC(a):

\[ A(a): [+\text{nas}] \rightarrow [\text{aplace}] \]

However, each language follows a different strategy when a nasal comes into contact with a voiceless consonant across a morpheme boundary. We can state these strategies as A-Rules.

Swahili: \[ A(b): [+\text{nas}] \rightarrow \emptyset \] e.g., \( n + p \rightarrow p \)

Nyakyusa: \[ A(b): [-\text{nas}] \rightarrow [+\text{voice}] \] e.g., \( n + p \rightarrow mb \)

UMbundu: \[ A(b): [-\text{nas}] \rightarrow \emptyset \] e.g., \( n + p \rightarrow m \)

These A-Rules change feature values and even delete segments in order to prevent or repair violations of RC(b). They show why RCs and A-Rules have to be formulated separately, and also why they must be linked together. Joint formalization as If-Then conditions would effectively turn them into P-Rules; giving our A-Rules the status of P-Rules would present two obviously related facts as a mysterious conspiracy. In addition, the example raises several interesting issues.

The case of UMbundu seems to suggest that A-Rules are ordered: place assimilation RC(a) must precede deletion RC(b). This is not how I would like to interpret the application of A-Rules. Rather, I suggest that no temporal sequence is involved and that A-Rules are all active simultaneously. In our case, the nasal must be (not: become) homorganic AND the voiceless obstruent must be suppressed.

Many Bantu languages including the three ones mentioned above have one or more bound morphemes consisting of just one nasal consonant, and this nasal typically occurs before some other consonant to which it is assimilated in place features. What is the proper specification and status of this homorganic nasal? In our framework where RCs do not fill in blanks we could not say that it is underspecified or in any way distinct from other nasals. What then is its place specification in the lexicon? (Cases where it precedes a vowel do not provide a satisfactory solution and have to be handled separately.) The answer is that the lexical representation of the homorganic nasal is fully specified but the place features are not defined. We could paraphrase this as "the lexical value of the place features is unpredictable and does not matter". (This is very much like a variable in a Pascal program, which occupies storage space at a
Some Bantu languages have a full set of homorganic nasals, but the velar nasal occurs exclusively in the sequence "homorganic nasal plus stop". Such languages are, for instance, Mituku, Kimbundu, and Dciriku. We cannot formulate a segmental RC that would exclude \( [\eta] \), hence our set of RCs will define a segment inventory that includes \( [\eta] \). We account for this distribution of \( [\eta] \) by formulating a (negative) sequential RC:

\[
\text{RC}(d) \quad \text{NOT} \quad [+\text{nas}] \[+\text{back}] \[-\text{cons}]
\]

The fact that there is no corresponding A-Rule limits RC\( (d) \) effectively to describing lexical items - though it is true of surface word structure, too. There just is no process by which the restriction stated in RC\( (d) \) would be violated.

6. Final Remarks

We might ask why linguists think that a non-redundant system or representation is desirable, when all the evidence we have is that language is redundant at all levels. Maybe there is the notion that the human capacity for the storage of the lexicon is limited, or that storage of redundant information would not be optimal. I don't know whether there is any evidence for this notion. On the contrary, storage of lexical elements in a redundant form would have the advantage of safer retrieval if we assume that access to the stored items may be imperfect. For example, it happens to me that I imperfectly remember a telephone number (information with very low redundancy); I may remember all but the least two digits, or all the digits but not their full order; in such a situation it would be very helpful to have RCs that disambiguate my imperfect knowledge.

And - if one more computer analogy is permitted - redundant storage might also provide faster access.

Language acquisition also appears to point towards redundant storage of the lexicon and the phonological system. Clearly, words are learned long before the full phonological system is acquired, and why would we want to assume a periodical housekeeping operation by the language learner in which he or she step by step removes redundant specifications? As for the phonological system,
it can be observed that small children have greater difficulty in understanding alien pronunciations than grown-ups, and also that they cannot help but acquire the exact dialectal norms of pronunciation of their environment. This is what would be expected if children were to store fully specified segments before they acquire the RCs that enable the grown-up language user to equate whole phonological systems.

My conclusion is that we neither want nor need a redundancy-free representation. RCs state the redundancies that exist in a particular phonological system, and Negative RCs are the proper tool for expressing what RCs are supposed to do, i.e. to exclude non-permissible segments.

References


