REDUNDANCY RULES IN PHONOLOGY

RICHARD STANLEY

University of California, Berkeley

This work contains a discussion of redundancy rules and the role they play in the phonological component of a generative grammar. Phonological redundancy rules were first given a clear theoretical foundation in the MORPHEME STRUCTURE RULES used by Halle (1959) to predict redundant phonological information in morphemes. We will be concerned in what follows with reformulating and extending this original theory of morpheme structure rules. We will give particular attention to the formal nature of morpheme structure rules and to the use of blanks in representing redundancies, and will give a solution to a long-standing problem in this area. In spite of this improvement in morpheme structure rule theory, we will find other motivations for introducing a new formal device, MORPHEME STRUCTURE CONDITIONS, to replace morpheme structure rules. We will attempt to show that, both in the types of statements which can be made about the structure of morphemes and in the formal nature of these statements, morpheme structure conditions are to be preferred over morpheme structure rules.

1.1. INTRODUCTION. The fact that natural languages possess a high degree of phonological redundancy is well known. Also, it has long been observed that the existence of phonological redundancy is due to the fact that each language exhibits systematic constraints on its phoneme sequences so that not all sequences of phonemes form possible morphemes of the language. Morpheme structure rules capture the intimate connection between the phonological redundancy of a language and the constraints it exhibits on phoneme sequences in its morphemes. In fact, a morpheme structure rule can be interpreted both as a statement of a constraint on phoneme sequences and as an algorithm for predicting redundant feature values in phoneme sequences. The morpheme structure rule itself is neutral as regards its interpretation. To illustrate, we observe in English that a morpheme-initial nasal cannot be immediately followed by a consonant. This observation will be reflected in a morpheme structure rule which says, in effect, that any segment following a morpheme-initial nasal must be [−Consonantal]. Interpreted as the statement of a constraint on possible phoneme sequences in English, this rule says that no [+Consonantal] segment can follow a morpheme-initial nasal. Interpreted as a algorithm for predicting redundant feature values, the rule shows that the value of the feature Consonantal may be left blank in segments immediately following a morpheme-initial nasal, since this value is predictably '−' (i.e. 'minus') in this position. Thus, a full set of morpheme structure rules for a language will do two things: it will state, in terms of features, all constraints on what sequences of phonemes are possible in morphemes, and it will allow each morpheme to have a representation in which redundant feature values are omitted.

The above-mentioned rule can be called a SEQUENCE STRUCTURE RULE: it makes a statement about possible sequences of phonemes. We can also make statements about single phonemes, regardless of their context. A rule giving such a statement can be called a SEGMENT STRUCTURE RULE: it makes a statement about the feature composition of individual phonemes. To illustrate this latter type of rule we can say that, in English, each [+Nasal] segment is also [+Voiced]. Like sequence structure rules, segment structure rules are, in themselves, neutral as regards their interpretation. Thus, the above segment structure rule can be interpreted not only as the statement of a constraint on the feature composition of phonemes, saying that no [+Nasal] segment may be [−Voiced], but also as an algorithm for predicting redundant feature values, showing that the value of the feature Voiced may be left blank in [+Nasal] segments, since this value is predictably '+'. A full set of segment structure rules for a language will suffice to predict all the nondistinctive feature values from the feature values which are distinctive. Segment structure rules have generally been put in the phonological rules and not in the morpheme structure rules. However, we will find reason to regard all redundancy rules, whether they involve predictions within a single segment (segment structure rules) or in a sequence of segments (sequence structure rules), as being included in the morpheme structure rules. We will return to this question in detail below.

It has been common for linguists to view phonological redundancy as a statistical entity to be treated in the way redundancy is treated in information theory (e.g. Carroll 1961:201–5; Cherry 1961:94, 115–20, 180–7, 272–3; Gleason 1961: 373–90; Hockett 1958:84–91; Miller and Chomsky 1963:439–43). The redundancy of a language is then given as a percentage, which is high if the language is highly redundant and low if the language has little redundancy. However, such a percentage does not give any information about the phonological structure of the language. It merely represents an average, statistical value of the redundancy for the whole language, and does not tell us specifically what local constraints in phoneme sequences contribute to this redundancy. Since these local constraints are of primary interest in the study of the phonological redundancy of natural language, it is clear that we need a more detailed statement of redundancy than the statistical one. We need a statement which concentrates on the specific nature of these local constraints, and not merely on the over-all effect that these constraints have on the efficiency of a language as a communication system (as this efficiency is given in a statistical measure of redundancy). Such a statement is provided by the morpheme structure rules.

1.2. THE TWO LEVELS OF REPRESENTATION. In a generative phonology of a language there are two levels of representation: the SYSTEMATIC PHONEMIC level and the SYSTEMATIC PHONETIC level (Chomsky 1964:68). The phonological component of a grammar consists of a set of PHONOLOGICAL RULES (called P rules) which map representations of sentences at the systematic phonemic level onto the representations at the systematic phonetic level. The morpheme structure (MS) rules, on the other hand, do not map one level of representation onto another, but rather state the redundancies that exist at a single level, the systematic phonemic level.

Each morpheme is represented at the systematic phonemic level as a SYSTEMATIC PHONEMIC MATRIX.[1] Each row of this matrix corresponds to a distinctive

[1] The assumption is that each morpheme has a SINGLE underlying representation, its systematic phonemic matrix; the P rules derive the various allomorphic forms of each

feature of the language, and each column corresponds to a segment (systematic phoneme) of the morpheme. Each entry in this matrix is either '+' or '−'. Thus, in particular, there are no blank or '0' entries. The systematic phonemic representation of a sentence consists of a string of systematic phonemic matrices (one for each morpheme of the sentence) together with a labeled bracketing which is the surface structure of this string (Chomsky 1964:64). The P rules apply to the systematic phonemic representation in cyclical fashion: the entire set of P rules is applied, in the proper order, to each string of matrices enclosed by the innermost bracket sets of the surface structure, then the innermost brackets are erased and the process is repeated; this is kept up until there are no more brackets, and the string which results is the systematic phonetic representation (Chomsky 1964: 13; Chomsky and Miller 1963:313–19).[2]

We are not concerned here with the P rules, but rather with the systematic phonemic matrices which form their input. As noted above, these matrices are fully specified: they have no blank entries. This full specification is desirable, as we will see in §3.3 below, since the P rules are most easily and naturally defined as operating on fully specified matrices. However, we would not want to enter a morpheme in the DICTIONARY as its fully specified systematic matrix, for, clearly, this matrix contains much that is redundant. Thus we will enter each morpheme in the dictionary as a DICTIONARY MATRIX which is identical with the systematic phonemic matrix of the morpheme except that all redundant feature values are left blank (or, equivalently, are replaced by the symbol '0').[3] The MS rules, consisting of sequence structure rules and segment structure rules, will map each incompletely specified dictionary matrix onto the corresponding fully specified systematic phonemic matrix.

It is important to notice that the MS rules, as here conceived, are exclusively redundancy rules, and that they represent redundancies at a single level, the systematic phonemic level. That is, dictionary matrices and systematic phonemic matrices are representations of the same level in the sense that each dictionary matrix is simply a less fully specified, redundancy-free version of the corresponding systematic phonemic matrix. On the other hand, it would be totally wrong to view systematic phonemic representations as redundancy-free versions of the corresponding systematic phonetic representations. The P rules, which map the

morpheme in their respective environments from this systematic phonemic matrix. This assumption is, of course, the usual one of generative phonology. It must be modified in the obvious way in the case of suppletion.

[2] Actually, in the works cited, not all of the P rules are regarded as being cyclic; rather there is a set of precyclical rules, a set of cyclical rules, and a set of postcyclical rules. The precyclical rules are just the morpheme structure rules, the cyclical rules describe the main phonological processes, and the postcyclical rules are low level phonetic rules. For us, however, the morpheme structure rules are not regarded as forming part of the P rules but are statements which form part of the lexicon; thus there are just two kinds of P rules, cyclical and postcyclical. This, together with the fact that for us the input to the (cyclical) P rules consists of fully specified matrices, constitutes the main difference between our position and that of the works cited. Our position and the justification for it will be fully discussed below.

[3] Chomsky (1964:66) has used the term CLASSIFICATORY MATRIX in the same sense that we use the term dictionary matrix.

former onto the latter, are in no sense redundancy rules; their role is to change feature values, not to fill in blank feature values.

The history of the ideas which led up to the position stated in the previous paragraph is quite interesting, and an outline of its main points may be helpful. It seems fair to say that, in the descriptivist tradition, phonemic representations were conceived of as containing just what was not redundant in phonetic representations. There were many attempts to clarify this conception and to give it a precise theoretical framework. One such attempt was the original distinctive feature theory of Jakobson, Fant, and Halle (1951) and of Jakobson and Halle (1956). A careful reading of these works shows that the phonemic representations, as there conceived, were simply less fully specified, redundancy-free versions of the phonetic representations, for to pass from the former to the latter one needed only to fill in blank feature values, never to change feature values. In fact, a phoneme was exactly what was non-redundant (invariant or distinctive) in all its phonetic realizations. Thus phonetic and phonemic representations were, in a significant sense, representations of the same level, and differed only in whether or not the redundant information of that level was explicitly indicated (see Chomsky 1957 for an enlightening discussion of these and related issues).

The discovery made by Halle (1959) was that the above conception of phonology was not adequate. Specifically, he showed that in order to characterize many of the generalizations that can be made about the phonological structure of a language, a level of representation was needed which was far more abstract than the phonemic representations referred to above. The representations of the level proposed by Halle were in terms of units which he called incompletely specified morphonemes, and the level itself can be called morphonemic to distinguish it from the earlier phonemic representations. Morphonemic representations, like phonemic representations, left all redundant information blank. However, unlike phonemic representations, morphonemic representations were not merely redundancy-free versions of phonetic representations, for the rules which mapped the former onto the latter could involve changing of feature values (including insertions, deletions, permutations of segments, etc.) as well as filling in of redundant feature values. In fact, Halle found that NO representations related to phonetic representations by statements of redundancy, or by any 'biunique' statements, played a role in an adequate grammar.

The rules, as there organized, consisted of an ordered set of MS rules followed by an ordered set of P rules. In some sense the MS rules were viewed as redundancy rules, but this fact was not emphasized, and the distinction between redundancy rules and feature-changing rules played no real role. An MS rule which happened to change feature values would not have been excluded if it seemed to give the right results. More important, the MS rules, to the extent that they were viewed as redundancy rules, were not viewed as stating redundancies at some particular level, but were viewed only as part of the whole set of phonological rules which mapped the incompletely specified morphonemic level onto the fully specified phonetic level. Moreover, since many redundancy rules appeared only quite late in the phonological rules, no FULLY SPECIFIED level ever actually appeared, in the derivations of the phonological rules, which would correspond to the level we have called systematic phonemic in this paper. Only morphonemic representations (consisting of strings of morphemes each in its incompletely specified dictionary form) and phonetic representations were of theoretical significance.

One of the proposals embodied in the present paper is that it is more natural to attribute theoretical significance to two FULLY SPECIFIED levels, the level of phonetic representations and the level obtained by filling in all redundant feature values in morphonemic representations. This latter level is the one we have called systematic phonemic (which, let it be recalled, we do regard as being fully specified). Essentially, this proposal amounts to demanding that redundancy rules be clearly distinguished from rules which change feature values; the former are rules which state redundancies at a single level, while the latter are rules which map one level onto another. Specifically, the MS rules state redundancies at the systematic phonemic level; the dictionary matrices are representations of this level before the redundant feature values have been specified by the MS rules. The P rules map fully specified representations of the systematic phonemic level onto fully specified representations of the systematic phonetic level.

It is possible that, just as MS rules state redundancies at the systematic phonemic level, there are rules which state redundancies at the systematic phonetic level. In fact, we have seen that it was just such a statement of redundancies at the phonetic level which was regarded by many linguists as, essentially, constituting a phonemic system. It is thus possible that stating phonetic redundancies in this way will enable us to capture something resembling a phonemic system, even though we are working in grammars which have only two levels of representation, systematic phonetic and systematic phonemic, neither of which resembles a phonemic level. This possibility, of course, does not alter the force of the arguments given in Halle (1959) and Chomsky (1964) that there is NO LEVEL OF REPRESENTATION in grammars which is phonemic. Phonetic redundancy rules would enable us to simplify the late P rules, for these P rules would have to state only the non-redundant, never the redundant, features of the phonetic level. Further, they would provide a framework for making generalizations about the phonetic pattern of a language, a framework which is, at present, conspicuously lacking. This view is certainly plausible. However, the argument given in §2.1 seems to show that no simple solution incorporating it is possible.

The fact that MS rules apply to individual morphemes, and not to strings of morphemes in a sentence, reflects the easily verified empirical fact that the constraints holding within single morphemes are more restrictive than the constraints which characterize larger units. (For some discussion, see Halle, 1959:39, § 1.57.) Since the MS rules apply only within individual morphemes, and never depend for their operation on anything outside the morpheme, we will view them as forming part of the dictionary rather than part of the phonological component (the P rules). In other words, the dictionary has two parts. One is a list of morphemes which includes (among other things) a dictionary matrix for each morpheme. The other part of the dictionary is a set of statements, the MS rules, which allow us to convert the abbreviated dictionary matrix of each morpheme into its fully specified systematic phonemic matrix whenever this morpheme is used in a sentence. We can assume that the LEXICAL RULE described by Chomsky (1965:84) states that the dictionary matrix of each morpheme selected is passed through the MS rules, yielding its systematic phonemic matrix, and that it is this systematic phonemic matrix which is actually inserted into the deep phrase marker. We can also assume that the dictionary matrices of grammatical formatives added by the transformational rules pass through the MS rules when they are added. In summary, then, all the phonological redundancy rules of the grammar are stated in a single separate set of rules, the MS rules, which predict all phonologically redundant information before either the transformational rules or the P rules apply. This way of viewing the MS rules has two desirable results. First, the P rules now operate only on fully specified matrices; the advantage of this will be discussed in §3.3. Second, the MS rules now occupy no fixed position in the set of P rules and thus can be used to state various kinds of redundancies which arise during the operation of the P rules; this will be discussed in §1.5 and §1.6.

At this point it is worth noting a major formal difference between MS rules and P rules. The P rules may change feature values (that is, change a minus to a plus or a plus to a minus), they may add or delete whole segments, and they may permute segments. The MS rules, on the other hand, must not do any of these things; their sole function is to fill in blank entries with the proper values ('+' or '−'), for, as noted, they are exclusively redundancy rules.

1.3. SEGMENT STRUCTURE RULES. In this and the following sections we will describe in more detail the formal nature of the two types of morpheme structure rules, segment structure rules and sequence structure rules. At the systematic phonemic level there is an inventory of SYSTEMATIC PHONEMES, each of which is represented by a one-column, n-row matrix, where n is the number of distinctive features in the language. Each row in a matrix corresponds to one of these features, and each entry in a matrix is either '+' or '−'. To illustrate, consider Halle's analysis of Modern Standard Russian (Halle 1959:45).[4] Here the systematic phoneme *a* would have the representation 1a (See next page).

Similarly, we could give a fully specified matrix representation of each Russian systematic phoneme. However, the representation in 1a obviously contains more than enough information to identify *a*. In fact, all we need is the set of features given in 1b. 1a is derivable from 1b by the segment structure rules of Russian (2, 3):[5]

[4] The term 'systematic phoneme' was not used in this work. Halle's 'fully specified morphonemes' correspond in our terminology to systematic phonemes in which all segmental redundancies are left blank. His 'incompletely specified morphonemes' correspond to systematic phonemes in morphemes in which neutralized feature values as well are left blank (see §1.5). Note that, while we refer often to Halle's book for illustrations, our framework differs in important ways from the one presented there (see §1.2).

[5] Halle does not happen to give explicitly any such rules in his Russian text, but nothing essential is involved in this omission.

(1)

	1a	1b
Vocalic	+	+
Consonantal	−	−
Diffuse	−	−
Compact	+	+
Low tonality	+	
Strident	−	
Nasal	−	
Continuant	+	
Voiced	+	
Sharped	−	
Accented	−	−

(2) $$\begin{bmatrix} -\text{Consonantal} \\ +\text{Compact} \end{bmatrix} \rightarrow [+\text{Low tonality}]$$

(3) $$[-\text{Consonantal}] \rightarrow \begin{bmatrix} -\text{Strident} \\ -\text{Nasal} \\ +\text{Continuant} \\ +\text{Voiced} \\ -\text{Sharped} \end{bmatrix}$$

The segment structure rules (2) and (3) can easily be extended to a full set of segment rules for Russian. As we saw in §1.1, such a set of rules can be interpreted both as an algorithm for predicting redundant feature values and as a set of statements of the constraints on the feature composition of systematic phonemes. In this case, if we interpret the set of rules as a predictive algorithm, then we can give all Russian systematic phonemes in a reduced form (as in 1b above and in Halle 1959:45, Table I-3).[6] Dictionary matrices will always have systematic phonemes with at least as many blanks as these reduced forms, since the segment structure rules suffice to predict the fully specified systematic phonemes from these reduced forms independently of the context.

[6] Note that the reduced form of *a* given in 1b (and in Table I-3 of Halle) is marked [−Diffuse], even though this is predictable from the fact that all [+Compact] segments are [−Diffuse]. This is unavoidable if there is to be a branching diagram or, equivalently, if 'distinctness' is required. We return to these questions below.

If, on the other hand, we interpret the set of segment structure rules as stating constraints on the feature composition of systematic phonemes, then this set defines exactly the set of Russian systematic phonemes. In fact, the set of Russian systematic phonemes is the unique maximum set of fully specified eleven-row, one-column matrices that obey all the constraints given in the set of segment structure rules. For if some such fully specified matrix which was not one of the systematic phonemes of Russian obeyed all the constraints given in these rules, then the rules would simply not be specific enough. They could (and should) be extended to rule out this matrix, since this would effect greater savings in the dictionary representations of the actual systematic phonemes. Thus the segment structure rules, though included in the grammar to characterize redundancy, provide as a by-product the definition of the set of systematic phonemes; this obviates the need for an ad hoc definition of this set by means of a list or a separate set of statements of some sort. This is an important point. It provides a certain amount of justification for deviating from past practice and including a set of segment structure rules in the MS rules (we will give further justification in §1.6.)

Notice that, since systematic phonemic matrices are fully specified, we obviously cannot gain information about what is distinctive in a language by looking at these matrices. For example, if in a systematic phonemic matrix we see a [+Nasal] segment which is also specified [−Compact], we cannot infer that compactness is distinctive for nasals and thus that there are [+Compact] nasals as well. It is the presence or absence of a segment structure rule [+Nasal] → [−Compact] that gives us the information about whether or not compactness is distinctive for nasals. Further, it seems natural to find such information in a set of rules, rather than having to search through the morphemes of the language for it. Finally, notice that we cannot infer from the absence, say, of a specification for the feature Compact in a nasal segment in a DICTIONARY matrix that compactness is not distinctive for nasals in the language, but only that it is not distinctive in that particular environment.

1.4. SEQUENCE STRUCTURE RULES. Segment structure rules give information about one kind of phonological redundancy, that of systematic phonemes in isolation. When systematic phonemes are combined into sequences to form morphemes, a new kind of redundancy arises, a redundancy due to sequential constraints. For example, in Russian a morpheme-initial glide must be followed by a vowel (Halle, 1959:58, rule MS1a). Thus we can give the sequence structure rule:[7]

$$(4)\qquad [\ ] \rightarrow \begin{bmatrix}+\text{Vocalic}\\ -\text{Consonantal}\end{bmatrix} / + \begin{bmatrix}-\text{Vocalic}\\ -\text{Consonantal}\end{bmatrix} \text{—}$$

Rule (4) can be extended to a full set of sequence structure rules for Russian, as Halle does in his presentation of the MS rules (1959:58–61). (For Halle, ALL MS rules are sequence structure rules since he does not give any segment struc-

[7] The '+' which stands outside the brackets in (4) stands for morpheme boundary ('&' is used in Halle's work). This '+' is not to be confused with the '+' of the feature values '+' and '−'.

ture rules.) If we interpret this set of rules as a predictive algorithm, then, in dictionary matrices, many segments can be represented with even fewer feature specifications than the reduced forms discussed in the preceding section. For example, a segment following a morpheme-initial glide need not be specified for the features Consonantal and Vocalic, since these features are predicted by rule (4). Thus segment structure rules make the predictions in systematic phonemes that are possible without considering the context, and sequence structure rules make the further predictions which are possible when contextual restrictions are considered.

If, on the other hand, we interpret the set of sequence structure rules as a statement of constraints on systematic phoneme sequences, then this set provides a characterization of the set of 'possible' morphemes of Russian. This fact is clear if, as is quite natural, we regard a 'possible' morpheme as a form which may or may not occur in the lexicon, but whose phonological structure violates no sequence structure rule (and, of course, no segment structure rule) of the language.[8] Thus the sequence structure rules, though included in the grammar to characterize redundancy, provide as a by-product a characterization of the notion 'possible morpheme'; this obviates the need for a separate set of statements to characterize this notion (cf. Halle 1964b: 341–2). The justification for the use of sequence structure rules given here is similar to that given for the use of segment structure rules in the preceding section. The reason that the latter kind of justification has not previously been considered sufficient cause to put the segment structure rules in the MS rules will be discussed below.

1.5. NEUTRALIZATION. Our decision to include ALL the phonological redundancy rules in a single set, the MS rules, may at first glance appear incorrect, for there are certain kinds of rules which must apply in the P rules but which, at least in some of the environments where they apply, function as redundancy rules. Rules of 'neutralization' have this property. A feature is said to be 'neutralized' in an environment if the value of the feature in this environment is determined by a sequential constraint. It is clear that if this environment is limited to sequences which are contained within a single morpheme and which cannot extend across morpheme boundaries, then the sequential constraint will be stated as a sequence structure rule and will be included in the set of MS rules. Generally, however, neutralization involves sequential constraints which hold across morpheme boundaries as well as within morphemes. In this case, any rule which reflects the neutralization not only makes a statement about the phonological structure of morphemes, as do MS rules, but also, like P rules, describes how this phonological structure is altered when morphemes are combined in sequence. The obvious place

[8] Of course, to obtain the desired results, we must guarantee that each sequence structure rule reflects a general systematic fact about the language, and not a fact which is due merely to the existence of accidental gaps in the lexicon. This guarantee is provided by the criterion of simplicity, which gives a well-motivated means of separating the accidental sequential constraints from the systematic ones; according to this criterion only the systematic ones can be stated in the sequence structure rules (Halle 1964b:340–2). This means that the sequence structure rules provide a means of distinguishing accidental gaps in the lexicon from gaps whose existence is due to a general fact about the language.

to put such a rule, and the place where it has been put in past practice, is in the P rules, since there it can not only predict the neutralized features when the neutralizing environment is contained within a single morpheme, but can also change features to their neutralized value when they are neutralized in environments which include surrounding morphemes. We will argue, however, that the proper place for such a rule is in the MS rules. Before showing how this is possible, let us give a concrete example of a rule of neutralization.

Consider a language which has both long and short vowels at the systematic phonemic level, but in which the feature Long is neutralized before two consonants (regardless of whether the VCC sequence is divided among one, two, or three morphemes), so that only [−Long] vowels appear phonetically in this position. The rule of neutralization which reflects this constraint is

(5) $V \rightarrow [-\text{Long}]\ /\ \_\_CC$

Such a rule is a redundancy rule when the VCC sequence is contained within a single morpheme, for in this case the vowel is always in the shortening environment and thus need not be marked for length in the dictionary. Since we have decided to include all redundancy rules in the MS rules (so that the output of the MS rules can be fully specified), it is clear that (5) must be an MS rule. However, we are assuming that (5) also applies across morpheme boundaries, having, for example, the effect $+C\bar{V}C+C+ \rightarrow +C\breve{V}C+C+$. Thus (5) must apply in the P rules as well.

We could just leave the discussion at this point, saying that rules of neutralization, such as (5), have a double function, appearing both in the MS rules and in the P rules. Moreover, this repetition of rules is not necessarily undesirable; it merely represents the kind of situation, quite typical in natural languages, in which certain constraints happen to typify both sequences within a single morpheme and sequences which extend across morpheme boundaries. Compare this to the situation in which constraints applying within morphemes are totally different from those applying across morpheme boundaries; we could make such a situation less highly valued in grammars by agreeing to attach less cost to a P rule which is identical with some MS rule than to a P rule which imposes some entirely new constraint. This is a possible and feasible proposal.

However, it may be possible to state a rule such as (5) just once, in the MS rules, and to say that by convention it retains its effect in the P rules. In this case, such a convention would say that at any point during the operation of the P rules where a vowel appears before two consonants, the vowel is automatically made short. This convention would work only if the order of (5), in its original statement in the P rules, was not crucial. We will leave the discussion of neutralization in this inconclusive state and go on, in the next section, to discuss a much clearer case where certain redundancy rules (the segment structure rules) must be stated in the MS rules but still retain their effect in the P rules.

1.6. THE POSITION OF THE SEGMENT STRUCTURE RULES. In this paper we have adopted the practice of putting the segment structure rules in the MS rules, so that all the redundancy rules of the language appear together in one set. The fact that the segment structure rules appear in the MS rules has two main advantages. First, they provide a definition of the set of systematic phonemes,[9] and, second, they fill in all blanks in matrices before the P rules begin to apply.[10]

There is, however, an argument in favor of putting the segment structure rules in the P rules, and it is this argument that has won out in the past. The argument is that there are redundancies in segments that arise during the operation of the P rules, and that we can account for these redundancies by having the segment structure rules appear in the P rules.

Let us consider an example. Suppose that some language has non-compact vowels *i e o u* but not *ï ë ü ö*. This restriction can be reflected in the segment structure rule:

$$(6) \qquad \begin{bmatrix} -\text{Consonantal} \\ -\text{Compact} \\ \alpha\ \text{Grave} \end{bmatrix} \rightarrow [\alpha\text{Rounded}]$$

Now it is likely that any epenthetic non-compact vowel will also obey the restriction stated in (6); suppose that this is the case. Then we would want rule (6) to FOLLOW any P rule that states such an epenthesis, so that (6) could apply also to the epenthetic segments. For if (6) were stated in the MS rules, or in the P rules before the rules of epenthesis, then we would have to specify the epenthetic segments for the redundant feature Rounded and thus miss a generalization. It was thus argued that we must let the segment structure rule (6) appear in the P rules; in particular it must follow rules which state the epenthesis of non-compact vowels.

To give another example, consider a language whose non-compact stops are non-strident and whose non-compact continuants are strident.[11] This restriction could be stated in the segment structure rule:

$$(7) \qquad \begin{bmatrix} +\text{Consonantal} \\ -\text{Compact} \\ \alpha\ \text{Continuant} \end{bmatrix} \rightarrow [\alpha\text{Strident}]$$

If this language has a P rule whereby non-Compact continuants become stops in some environment, then we would like to state this rule as (8):

$$(8) \quad \begin{bmatrix} +\text{Consonantal} \\ -\text{Compact} \end{bmatrix} \rightarrow [-\text{Continuant}] \;/\; \begin{matrix}\text{(in the appropriate} \\ \text{environment)}\end{matrix}$$

But this is only possible if the segment structure rule (7) follows the P rule (8).

[9] This is discussed in §1.3. It is clear that if we adopt the usual practice of letting the segment structure rules be scattered through the P rules, each rule not appearing until it is needed, then this means of defining the set of systematic phonemes is not available. For in this case the segment structure rules no longer formally represent redundancies and constraints at the systematic phonemic level. There could be no motivation for defining units (the systematic phonemes) of that level in terms of this scattered set of rules.

[10] The advantage of this has been mentioned briefly above and will be discussed fully in §3.3.

[11] Such a language would have the non-compact consonants *t d s z*, but not *c* or *ʒ*, which are [−Continuant, +Strident] and thus violate (7). This language could still have *č* and *ǯ*, which, though also [−Continuant, +Strident], are [+Compact].

If (7) had been stated in the MS rules, or had preceded (8) in the P rules, then we would have to make (8) more complicated, stating it as:

$$(9)\quad \begin{bmatrix}+\text{Consonantal}\\ -\text{Compact}\end{bmatrix} \rightarrow \begin{bmatrix}-\text{Continuant}\\ -\text{Strident}\end{bmatrix} / \begin{matrix}\text{(in the appropriate}\\ \text{environment)}\end{matrix}$$

This means, in effect, that we have had to state the restriction on the features Continuant and Strident in non-compact obstruents twice, once in (7) and once in (9), and thus again we miss a generalization. To avoid this, it was argued, we must let the segment structure rule (7) appear in the P rules, where it must, moreover, follow rules such as (8).

It seems that examples such as the above two show that the segment structure rules must appear in the P rules, and thus that they cannot be put in the MS rules. There is, however, a serious conflict involved in this practice of putting segment structure rules in the P rules in order to state redundancies in segments which arise there. According to such a practice, a segment structure rule such as (7) must FOLLOW any P rule which, like rule (8), changes a non-compact continuant to a stop (or any rule which changes a stop to a continuant), or which introduces an epenthetic non-compact stop or continuant. But it is also necessary that (7) PRECEDE any P rule 'R' that needs information about the feature Strident in non-compact consonants, since the stridency is inserted only by (7) and since insurmountable problems arise if blanks occur in matrices during the P rules (cf. §3.3). Yet these two conditions can both be met only if rules like (8) always precede rules like 'R', and clearly there is no way to guarantee, or even any reason to expect, that this will be the case. The conflict involved can be summarized as follows. For some purposes we want the segment structure rules to state their redundancies EARLY in the P rules, so that these redundant features can be used by other P rules. But for other purposes we want the segment structure rules to state their redundancies LATE in the P rules so that they can state the redundancies in segments introduced or changed by the P rules. Because of this conflict, neither the usual practice of letting the segment structure rules be scattered through the P rules, nor the practice we advocated earlier of letting the segment structure rules appear only in the MS rules, can be accepted as it stands.

There is a way, however, of keeping the segment structure rules in the MS rules while still allowing them to state redundancies in the P rules. We merely need to adopt the convention that the output of each P rule is automatically subjected to the segment structure rules.[12] Such a convention would imply that

[12] There are various ways in which such a convention could be made precise. One way is the following. Any P rule of the form $X \rightarrow [\beta_1 F_1, ..., \beta_n F_n]/Y\_Z$ (where X, Y, and Z are bundles of features, and each β_i is either '+' or '−') is interpreted as applying in n steps. First, the feature F_1 is assigned the value β_1 in any segment which has the features in X; then any segment structure rule which has $\beta_1 F_1$ on the left of the arrow is applied, if applicable. Then the feature F_2 is assigned the value β_2 in the output of the first step, and segment structure rules involving $\beta_2 F_2$ are applied. This is kept up until the feature β_n is assigned to F_n and segment structure rules involving $\beta_n F_n$ are applied. According to this convention, the order in which the features are stated in the right side of a P rule may affect the output of the P rule, a fact which may be of some significance.

segment structure rules represent constraints that hold throughout the P rules as well as at the systematic phonemic level; the above examples, being typical of natural language situations, show that this is just what we find.[13]

We will assume such a convention in what follows. We thus have all redundancy rules in one place in the grammar (in the MS rules), so that the input to the P rules consists of fully specified matrices.[14] Moreover, some of these redundancy rules (the segment structure rules) have their domain of influence extended to the output of (some) P rules. The examples in this section show the inadequacy of theories without this convention. Yet it is, at this point, not at all clear how the details of such a convention are to be worked out. An extremely interesting and fruitful line of research would be to obtain data bearing on this question from a variety of languages.

FURTHER DETAILS

2.1. PHONETIC REDUNDANCY RULES. We have stressed that MS rules represent redundancies at the systematic phonemic level; they say nothing about redundancies at the systematic phonetic level. In fact, the theory has no redundancy rules at the systematic phonetic level, rules which correspond to the MS rules at the systematic phonemic level (but cf. the discussion in §1.2). Moreover, the complexity and depth of ordering which in general exists in the P rules shows that it may be difficult to construct any natural set of phonetic redundancy rules, even in principle. This is because phonetic representations are only obtained as the output of this highly structured set of rules (the P rules), so that in general a relationship among the elements of these representations cannot be simply stated in terms of those elements alone, but must be referred back to the systematic phonemic representations which underlie them.

As an example, suppose that in a language with vowels *i e a o u*, back vowels are fronted but not unrounded in the environment __CV when the V is *e* or *i*. If a later P rule drops the *e* or the *i* in some environment, then *œ ö* and *ü* will contrast with the five underlying vowels at the systematic phonetic level, but in a rather special enviroment which is difficult, perhaps impossible, to characterize at the systematic phonetic level—namely before a C which at the systematic phonemic level was followed by an *i* or an *e*. Thus the distributional facts cannot be stated in a natural way at the systematic phonetic level. In general there will be more or less simple distributions of elements at the systematic phonemic level (given in the MS rules), and more or less simple statements

[13] It seems certain that we do not want all P rules, especially the late ones, to have their output subjected to the segment structure rules. It is still an open question how we are to identify, in a non-ad-hoc manner, just which P rules are so subjected.

[14] Note that the argument that the segment structure rules should be put in the P rules, and there as late as possible, on the grounds that then derivations would become 'more economical' since redundant feature values would be carried along for a shorter time, would be meaningless. More economical grammars are to be preferred only when the kind of economy achieved is that defined by an evaluation measure, and a particular evaluation measure is chosen only because of a relatively higher degree of explanatory power achieved in grammars evaluated highly by this measure. Clearly, delaying segment structure rules merely to obtain a more economical derivation explains nothing.

(in the P rules) which map representations at this level onto representations at the systematic phonetic level, but any direct statements of distribution of elements at the systematic phonetic level are likely to be complex and to involve loss of generalizations.

2.2. ORDER. It is possible, in fact usual, to present the MS rules as an ordered set. Rule B follows rule A in this ordering just in case B refers to feature values inserted by A; in particular, this means that all segment structure rules follow all sequence structure rules. However, since MS rules are redundancy rules, which only insert feature values and never change them, this ordering is of a rather trivial sort. A non-trivial kind of ordering can only be obtained in rules which, like P rules, can change feature values; if a P rule R changes the value of a feature f, different results will obtain if a rule affecting feature f is placed before R from those occurring if it is placed after R. Moreover, these effects of ordering of P rules are quite necessary, so that this ordering makes a significant claim about the nature of phonological structure in natural languages (cf. Halle 1964b:338, 343–4, Chomsky 1964:70–5). However, if a set of rules, such as the MS rules, never change feature values, then different orderings will NOT give different results. In fact, it should be clear that the MS rules can be presented as an unordered set with the general instruction 'keep applying rules over and over and in any order until all blanks are filled'. Further, they actually must be so presented, since to make the (stronger) claim that they are ordered gives the false impression that this order plays some role (as it does in the P rules). We will see in §4 that MS conditions, which we introduce to replace MS rules, provide a natural formalism for stating just what the above discussion shows that it is necessary to state; i.e. a set of unordered statements about the structure of morphemes.

2.3. INSERTION AND DELETION RULES. Notice that we have not allowed MS rules of the forms $\emptyset \rightarrow x/A\_\_B$ or $x \rightarrow \emptyset/A\_\_B$; i.e., the formalism, as we have interpreted it, does not permit us to use insertion rules or deletion rules as MS rules. This is a consequence of our requirement that each dictionary matrix be exactly like the corresponding systematic phonemic matrix except that redundant features in the latter are left blank in the former; in particular, the two matrices have the same number of columns. This requirement is made simply because we know of no cases where regularities of morpheme structure can be stated best in terms of insertion and/or deletion rules.[15] Of course there may be insertion and deletion rules which operate in a grammar, but in most cases it is clear that these are P rules, since they must follow other rules known to be P rules.

It has sometimes been suggested that insertion rules in the MS rules could be used to make specious savings of feature specifications in the dictionary, and that this constitutes a problem for the theory. For example, suppose we have a language with no consonant clusters. If the most common vowel in this language

[15] In fact, the theory of morpheme structure advanced in §4 below is incapable of even stating insertion or deletion rules which are to apply SOLELY in intra-morphemic environments. Thus, this theory represents a claim that such insertion or deletion rules cannot occur in natural languages. Not only does this seem intuitively correct, but it has not been refuted by any cases which have come to our attention.

is *e*, then we could leave *e* out of the dictionary wherever it occurs between two consonants, and have an MS rule $\emptyset \rightarrow e/\mathrm{C}\_\mathrm{C}$. This, it is claimed, will save features in the dictionary, even though this saving is wholly unnatural and undesirable. However, this claim is correct only if the MS rules are ordered and had the insertion rule as the first rule, for otherwise the other MS rules could not make use of the legitimate generalization that no consonant clusters occur. But we saw reason in §2.2 to require that the MS rules be unordered. Thus the undesirable insertion rule is not allowed. More generally, the requirement that the MS rules be unordered is sufficient to disallow any insertion or deletion rules.

2.4. Distinction between MS rules and P rules. In our presentation of the formalism involved in a theory of generative phonology, we have made very clear the distinction between MS rules and P rules. The former are exclusively redundancy rules and are totally different, in their function and their position in the grammar, from the latter, whose function is to derive the phonetic representations from the systematic phonemic ones. However, it should be noted that this distinction has not always been so sharply made. In fact, when MS rules were first introduced, they were viewed as being very nearly on a par with the P rules; there was one set of rules in the phonological component, the early rules of which were MS rules and the later of which were P rules. Further, there are many cases in the literature where one is at a loss to decide whether a rule is intended to be an MS rule or a P rule. In this paper, however, we have argued that MS rules are quite distinct from P rules, both in their logical form and in their linguistic function, and that it is both necessary and desirable to maintain this distinction. In fact, phonological descriptions would be much more valuable if they contained a set of MS rules clearly distinguished from the P rules.

We will show in §4 that the distinction between MS rules and P rules is so great that they are even best handled by different formal devices, instead of both being handled by 'rules'. P rules will remain rules, but we will introduce MS 'conditions' to replace MS rules.

2.5. Branching diagrams. Halle (1959:32, §1.53, and 46, fig. I-1) uses a branching diagram to represent the segments of Russian. In his terminology, these segments are called 'fully specified morphonemes'; for us they correspond to systematic phonemes in which the feature values predicted by the segment structure rules are left blank. Each descending path from the top node to a terminal node in a branching diagram represents the distinctive (non-redundant) feature values of a systematic phoneme of the language. However, it is important to notice that not all ways of choosing the non-redundant feature values leave open the possibility of constructing a branching diagram (Halle 1950:35). A branching diagram can only be constructed if there is a feature f_1 which is non-redundant in every segment; if there is a feature f_2 which is non-redundant in all $+f_1$ segments and a feature f_3 which is non-redundant in all $-f_1$ segments; if there are features f_4, f_5, f_6, and f_7 which are non-redundant in all $+f_2$, $-f_2$, $+f_3$, and $-f_3$ segments respectively, etc. We will see below that there are situations in natural languages which do not exhibit this kind of distribution of non-redundant features. In general, however, generative grammars of languages have been chosen so that there IS a branching diagram for the systematic phonemes. There are several reasons that branching diagrams have been regarded as important: (1) Giving segments in a branching diagram appears to be the most direct means, involving the fewest feature specifications, which will guarantee that each pair of segments is DISTINCT in the sense that for any pair of different segments there is at least one feature f such that one member of the pair is specified +f and the other −f. (2) The branching diagram gives a hierarchy of features which can be interpreted as meaning that the features at high nodes (such as Consonantal) are in some sense more basic than the features at low nodes (such as Voiced). (3) The branching diagram gives a way of formalizing the notion of archiphoneme. It is for these reasons that branching diagrams have been given in grammars. However, we will show that each of them is open to question.

Representing segments in a branching diagram may be quite natural if it is assumed that segments are to be distinct. However, distinctness is only one of the possible formal requirements for guaranteeing that the segments are all kept apart, i.e. are all distinguishable. In fact we will give independent evidence (§§3.5, 3.6) in favor of adopting a criterion of distinguishability of segments which is weaker than distinctness. If this evidence is accepted, then the notion of branching diagram must be abandoned, for non-distinct segments cannot be represented in such a diagram.

The second point is more important, since there is obviously some kind of hierarchical relationship among the features which must somehow be captured in the theory. However, an attempt to capture this hierarchy in a branching diagram seems somewhat strange in light of the fact that there may be considerable freedom in the way this branching diagram is constructed for a given set of systematic phonemes; a different choice of redundant feature values in this set will lead to a different branching diagram and thus to a different hierarchy of features. Obviously we would choose, if possible, that branching diagram which gives the hierarchy we feel is right; but this just means that we know beforehand what hierarchy we want and are simply choosing to represent it in a branching diagram. In this case, though, we might just as well describe the hierarchy separately since it has nothing essential to do with the branching diagram.

What we want, of course, is to find some characteristics of features and their interrelations that force us, on independent grounds, to assume a PARTICULAR hierarchy of features. That is, we want to find some definite formal property of features, perhaps stated in terms of the different ways in which different features behave in the P rules or in the MS rules, which points in an unambiguous way to the existence of a specific hierarchy. Only in this way can a feature hierarchy come to represent more than a vague set of intuitions about what features are more basic than others, and thus only in this way can we discover a real NECESSITY for incorporating the notion of feature hierarchy in the grammar.[16]

[16] Some of the ideas in the above paragraphs were suggested to me by Chin-W. Kim.

What such a formal property might be is an open question at this point. We have seen, however, that the relation of redundant to non-redundant feature specifications in segments, which relationship can sometimes be stated in a branching diagram, is a formal property of features that is NOT, in itself, well enough determined on independent grounds to justify its use in characterizing a hierarchy. Further, the fact that high-level features can sometimes be predicted in terms of low-level features makes it seem that the relation of redundant to non-redundant features cannot be expressed at all in terms of a branching diagram.

To illustrate this last point, consider the following example. In many languages which use the feature Strident, the only [+Strident] segments are true consonants. This means we can leave the features Consonantal and Vocalic blank in [+Strident] segments and predict them by a segment structure rule

$$(10) \qquad [+\text{Strident}] \rightarrow \begin{bmatrix} -\text{Vocalic} \\ +\text{Consonantal} \end{bmatrix}$$

Before the application of such a rule, of course, the [+Strident] segments may not be distinct from vowels, liquids, or glides which happen to have all other features in common with them (unless we specify all vowels, liquids, and glides as [−Strident] which, clearly, we don't want to do). But this lack of distinctness is no problem: there are still enough features specified to keep all segments distinguishable (for further discussion, see §3.5).

The third reason for having branching diagrams, mentioned at the beginning of this section, was that it was assumed that they formalize the notion of archiphoneme. This assumption, however, is false. The original idea was that just as the systematic phonemes (the 'fully specified morphonemes' for Halle) are represented at the bottom, terminal nodes of the diagram, so archiphonemes are represented at higher, non-terminal nodes. For example, the nodes labelled 10 (the nodes for the feature Sharped) by Halle (1959:46) represent archiphonemes of consonants from which the feature Sharped has been extracted, e.g. the archiphoneme /t t,/. Similarly the nodes labelled 9 (Voiced) represent archiphonemes of consonants from which both the features Sharped and Voiced have been extracted, e.g. the archiphoneme /t t, d d,/. However, it is easy to see that the branching diagram wrongly limits what the possible archiphonemes are. For example, letting the feature Sharped label the bottom nodes excludes the possibility of having archiphonemes with just the feature Voiced extracted. Yet obviously the notion of archiphoneme applies as well to those segments from which the feature Voiced has been extracted as to those from which the feature Sharped has been extracted. Thus branching diagrams fail to capture the notion of archiphoneme because of a quite basic and unavoidable fact of their structure.

THE FORMAL APPARATUS

3.1. IMPROPER USE OF BLANKS. We have made three assumptions about MS rules which differ from those made in past works. We have assumed that all redundancy rules are included in the MS rules so that their output, which is the input to the P rules, is fully specified (cf. §2.2); we have assumed that MS rules are exclusively redundancy rules, and never change feature values (cf. §1.2); finally, we have assumed that the MS rules are to be presented as an unordered set (cf. §2.2). Arguments in favor of adopting these assumptions have been given in the sections referred to. In the present section, which is a discussion of the formal apparatus involved in rules and their application, we will show that insurmountable formal problems arise if these assumptions are NOT made; i.e. for purposes of illustration, we will abandon these assumptions, and will discuss the phonological component as it is usually presented. Specifically, we posit that the phonological component consists of a single ordered set of rules which maps strings of morphemes in their dictionary form onto the phonetic representations of the sentences comprised by these strings. An initial subset of these rules is the set of MS rules, while the rest are the P rules. MS rules apply only within single morphemes, and may change feature values if desired. Finally, some redundancy rules may be stated in the P rules, so that the output of the MS rules need not consist of fully specified matrices. Our concern will be the difficulties which blank feature values cause in this, the usual conception of phonology.

Recall that the set of MS rules applies to each morpheme's dictionary matrix, filling in '0' entries in this matrix and producing a more fully specified matrix: the systematic phonemic matrix of the morpheme.[17] Thus we have chosen to indicate formally the fact that a feature value is redundant in the systematic phonemic matrix of a morpheme by replacing that feature value by '0' in the dictionary matrix of the morpheme and by including an MS rule which changes that '0' to the proper feature value. However, in using this system we must take care that the feature values remain BINARY, and that '0' in a dictionary matrix is never allowed to function as a third value, distinct from '+' and '−'. The correctness of any empirical claim that distinctive features are binary is, of course, not at issue here. The point is simply that, once we decide to use a binary system, we must be formally consistent. Unfortunately, it is all too easy to be formally inconsistent by letting '0' function as a third feature value, and this has often been done unknowingly in the writing of generative grammars. What is important is that we keep the meaning of '0' clearly in mind. It is not a feature value, but merely a mark which indicates that the feature value of the entry in which it appears has not yet been filled in.

There are two general ways in which '0' can achieve an improper status. One way is to let the ONLY difference between two dictionary matrices be that one has a '+' entry (or a '−' entry) where the other has a '0' entry; in this case the '0' entry is being used to keep the two matrices apart, contrary to its intended use. The other way to let '0' achieve improper status is to formulate rules which mention '0' in their environment; clearly a rule must never depend for its proper application on there being '0' entries as yet unfilled in matrices. In either of the above cases, '0' would be functioning as a feature value. Not only is this contrary to our intention to have a binary system, but it would even be improper

[17] Both the symbol '0' and the absence of a symbol, i.e. a blank, are used in the literature to indicate redundant feature values. We will use them interchangeably.

as a ternary system, since only '+' and '−', and not blanks, are counted in the evaluation procedure.

When '0' is given improper status, it is often said that a 'specious simplification' or a 'specious saving' has been made. Specific examples of these situations will be given in §3.3. Such distortions of the binary system are, of course, to be avoided, and to avoid them we must take great care in how we formulate MS rules and how we interpret them as applying to dictionary matrices. Considerable attention has been given in the past few years to this problem. The goal is a formalism for MS rules which allows one to state all legitimate generalizations but which prohibits one from making any specious simplifications: i.e. a formalism for MS rules which does not permit '0' to function as a value distinct from '+' and '−'. This goal has been partially achieved by the general adoption of conditions on MS rules such as the 'distinctness' condition and the 'well-formedness' condition, but these conditions have several defects. The present section is devoted to a discussion of these issues.

3.2. Rule definition. We have said that MS rules are redundancy rules: that they are instructions for filling in the proper values ('+' or '−') of features which have been left blank in dictionary matrices. In this section we will give a precise definition of MS rule, and will show just how such a rule is to fill in feature values. Following standard practice, we will say that a MS rule R consists of two parts: a structural description SD(R) and a structural change SC(R). SD(R) and SC(R) may be thought of as being a pair of incompletely specified, disjoint matrices with the same number of columns. The kinds of matrices we are concerned with here have entries '+', '−', or no entry (blank), and have a fixed number n of rows, each of which will correspond in an actual grammar to a distinctive feature. To say that a matrix is incompletely specified is simply to say that some of its entries are blank. To say that two matrices are disjoint is to say that one has a specification (a '+' or '−') in a position only if the other has no specification (a blank) in the corresponding position.

To illustrate these terms consider a rule R where SD(R) and SC(R) are as follows:

(11)
$$\mathrm{SD(R)}\ \begin{array}{c|c|c|c|}\cline{2-4} f_1 & + & & \\ \cline{2-4} f_2 & - & - & \\ \cline{2-4} f_3 & & & - \\ \cline{2-4} f_4 & & & \\ \cline{2-4}\end{array}$$

(12)
$$\mathrm{SC(R)}\ \begin{array}{c|c|c|c|}\cline{2-4} f_1 & & + & \\ \cline{2-4} f_2 & & & \\ \cline{2-4} f_3 & & & \\ \cline{2-4} f_4 & & - & \\ \cline{2-4}\end{array}$$

A rule such as R has usually been written

(13)
$$[-f_2] \rightarrow \begin{bmatrix}+f_1\\-f_4\end{bmatrix} / \begin{bmatrix}+f_1\\-f_2\end{bmatrix} \text{—} [-f_3]$$

We will sometimes use this notation, but will often write R in a form in which SD(R) and SC(R) are kept more clearly distinct:

(14)
$$\begin{array}{c} \mathrm{SD(R)} \begin{bmatrix} +f_1 \\ -f_4 \end{bmatrix} [-f_2]\,[-f_3] \\ \downarrow \\ \mathrm{SC(R)} \begin{bmatrix} +f_1 \\ -f_4 \end{bmatrix} \end{array}$$

The difference between the formulations in (13) and (14) is, of course, purely notational.

MS rules, as they have been formulated in actual practice, are often such that SC(R) lies entirely in one segment; that is, as in the above example, features are added in just one segment. However, there seems to have been no attempt to make this a formal requirement in MS rules (analogous to the requirement in the phrase structure rules that a single symbol be rewritten at a time), and in fact it has proved convenient to add features to more than one segment with a single rule. Thus Halle (1959:60–1, Rules MS 11d, 11e) seems to indicate that rules involving additions in two segments are necessary.[18] It is for this reason that we have set up our formalism so as to allow such rules. The disadvantage of this, of course, is that we have a quite powerful formal device which is needed in a relatively small number of cases of a limited sort.

Note that, by requiring that SD(R) and SC(R) be DISJOINT matrices, we are unable even to formulate any rule which mentions the same feature on both sides of the arrow, as in (15):

(15)
$$\begin{array}{c} X \begin{bmatrix} \vdots \\ +f_i \\ \vdots \end{bmatrix} Y \\ \downarrow \\ \begin{bmatrix} \vdots \\ +f_i \\ \vdots \end{bmatrix} \end{array} \quad \text{or} \quad \begin{array}{c} X \begin{bmatrix} \vdots \\ +f_i \\ \vdots \end{bmatrix} Y \\ \downarrow \\ \begin{bmatrix} \vdots \\ -f_i \\ \vdots \end{bmatrix} \end{array}$$

[18] They are not logically necessary, of course, since any MS rule of the form

(X)
$$\begin{array}{c} [A]\quad[B] \\ \downarrow \\ [C]\quad[D] \end{array}$$

(where A, B, C, and D stand for bundles of feature values) can be reformulated in two rules

(Y)
$$\begin{array}{cc} [A] & [B] \\ \downarrow & \\ [C] & \end{array} \quad \text{and} \quad \begin{array}{cc} [A] & [B] \\ & \downarrow \\ & [D] \end{array}$$

The point is that such a reformulation results in a loss of generality, since the environment SD(R) = [A] [B] must be stated twice. Note, however, that the reformulation of (X) as the two rules of (Y) is possible in general only if we are talking about MS rules—rules which do not change feature values. If the rules were P rules, then this reformulation would not be possible if the first rule in (Y) changed features in A in such a way as to render the second rule in (Y) inapplicable.

And, indeed, in actual practice, rules of the above form have not been used as MS rules.[19]

3.3. Rule application. Informally, an MS rule is an 'if-then' condition on matrices: each rule R says 'if a matrix M meets condition SD(R), then it must also meet condition SC(R)', or 'if a matrix M meets condition SD(R), then we are to fill in blanks in M until it also meets condition SC(R)'. To say more precisely how MS rules apply, we must introduce some preliminary definitions. Let us say that, given (two completely or incompletely specified) matrices M and N which have the same number of columns, then:

(1) M is a sub-matrix of N if, whenever M has a specification (a '+' or a '−') in a position, N has the same specification in the corresponding position, but not necessarily conversely.

(2) M is distinct from N (equivalently, N is distinct from M) if for some specification in M, N has the opposite specification in the corresponding position.

(3) M and N are incomparable if they are neither distinct nor in a sub-matrix relation.

When a matrix M is subjected to a rule R, we define the applicability of R to M as follows:

(4) If SD(R) is a sub-matrix of M, then M meets the conditions of R; thus R applies to M, and (the minimum required number of) features are added or changed in M until SC(R) is also a sub-matrix of M.

(5) If SD(R) is distinct from M, then M does not meet the conditions of R; thus R fails to apply to M, and M passes through unchanged.

In these two cases the definition of rule application is the obvious and natural one. Regarding the third possibility, that SD(R) is incomparable with M, it is not clear whether we should say that R applies to M or that R fails to apply to M.

Suppose that we say in general that a rule R fails to apply to (i.e. has no effect on) a matrix M if SD(R) is incomparable with M. Call this the sub-matrix interpretation of rule application: a rule R applies to M just in case SD(R) is a sub-matrix of M. Then given the three consecutive rules

(16)
$$\begin{array}{ccc} [\ \ ] & [+\text{f}] & [-\text{f}] \\ \downarrow & \downarrow & \downarrow \\ [-\text{g}] & [+\text{g}] & [+\text{g}] \end{array}$$

we see that segments blank for feature f will end up different both from segments marked +f and from segments marked −f after the application of these rules; segments blank for f end up −g, segments marked +f or −f end up +g. But then, as far as these rules are concerned, segments blank for feature f are neither +f nor −f; i.e. when blank, feature f functions as though it had a third value, different from both '+' and '−'. It is easy to see how the use of rules having the property of the rules in (16) can lead to specious simplifications of the kind mentioned above. For example, if we add a fourth rule

[19] Note, however, that the requirement of disjointness does not completely rule out the possibility that an MS rule can change feature values; see (21) in §3.5.

(17)
$$\begin{matrix}[-g]\\ \downarrow \\ [+f]\end{matrix}$$

to follow the above three, then the three 'dictionary matrices'

(18)
$$\begin{bmatrix}0f\\0g\end{bmatrix} \quad \begin{bmatrix}+f\\0\,g\end{bmatrix} \quad \begin{bmatrix}-f\\0\,g\end{bmatrix}$$

when operated on by the four rules of (16) and (17) become

(19)
$$\begin{bmatrix}+f\\-g\end{bmatrix} \quad \begin{bmatrix}+f\\+g\end{bmatrix} \quad \begin{bmatrix}-f\\+g\end{bmatrix}$$

But then, clearly, '0' has been used as a third feature value. Thus this interpretation of rule application (the sub-matrix interpretation) is not the desirable one.

Suppose, on the other hand, we decide to say that a rule DOES apply to (i.e. does have an effect on) a matrix M if SD(R) is incomparable with M. Call this the DISTINCTNESS INTERPRETATION OF RULE APPLICATION: a rule R applies to M just in case SD(R) is not distinct from M. Then, given the three consecutive rules

(20)
$$\begin{matrix}[\quad] & [+f] & [-f]\\ \downarrow & \downarrow & \downarrow\\ \begin{bmatrix}-g\\-h\end{bmatrix} & [+g] & [+h]\end{matrix}$$

we see again the blank (in feature f) is functioning as a third value: segments blank for feature f end up [+g, +h], segments marked '+' for f end up [+g, −h], segments marked '−' for f end up [−g, +h], after the rules in (20) have applied. It is easy to see how the use of rules having the property of the rules in (20) can lead to improper use of blanks; an example similar to the one just given is readily constructible. It follows that, under the distinctness interpretation of rule application, as well as under the sub-matrix interpretation of rule application, we are unable to avoid improper use of blanks. In short, when rules have to recognize blanks in any way,[20] the road is open for these blanks to function as a third value, and we see that this is true under either of the two possible interpretations of rule application.[21] Thus there is NO adequate way of defining the applicability of a rule R to a matrix M if SD(R) and M are incomparable. To avoid this difficulty, it seems we must require that, whenever a matrix M is subjected to a rule R, we must be able to tell from the SPECIFIED features in M whether SD(R) is a sub-matrix of M or is distinct from M; i.e., whenever a matrix M is subjected to a rule R, it must not be the case that SD(R) and M are incomparable. This requirement has come to be referred to as the well-formedness condition.

[20] That is, when we have to decide for a rule R and a matrix M, where SD(R) and M are incomparable, whether R is to apply to M or not.

[21] For an early statement of this problem (in which, we might add, no conclusions are reached), see Lightner 1963.

3.4. THE WELL-FORMEDNESS CONDITION. If there are MS rules $R_1, R_2, \dots, R_m$ in a language, then given a dictionary matrix D, the corresponding systematic phonemic matrix is formed by subjecting D to R_1, by subjecting the result to R_2, etc. Clearly, each time a matrix M is subjected to a rule R, we have to be able to decide in some way whether or not R is to apply to M, and the theory has to give us a constant, formal basis for making this decision. A set of dictionary matrices and MS rules is said to be WELL-FORMED if, whenever we subject a matrix M to a rule R_i in the course of this process, $SD(R_i)$ is distinct either from M or a sub-matrix of M;[22] i.e. a grammar is well-formed if the SITUATION NEVER ARISES that a matrix M is subjected to a rule R_i where $SD(R_i)$ and M are incomparable in the sense of §3.3 (3). This well-formedness condition was not given explicitly in Halle 1959,[23] but has been conformed to in practice and has been explicitly stated since 1959. (For a different but equivalent version of this condition, see McCawley 1964.) The reason for having this condition is that, as we have seen, specious simplifications can arise in grammars which are not well-formed. The problem involves the decision (which must be made once and for all in the definition of rule application) whether or not a rule R is to apply to a matrix M if SD(R) and M are incomparable; we have seen that either way of making this decision leaves open the possibility of using MS rules to make specious simplifications. This problem does not arise in well-formed grammars, since, by the definition of well-formedness, this decision does not have to be made.[24] It is on the basis of the above kind of reasoning that the well-formedness condition has been generally adopted, at least in practice.

It is a simple matter to check a set of MS rules for well-formedness: we need only to check something about how a small finite number of rules (the MS rules) apply to each of a finite number of matrices (the dictionary matrices). Moreover, well-formedness does not need to be a condition on how P rules apply if, as we suggested in §1.2, these rules operate on strings of matrices which are always fully specified.

Suppose, however, we adopt the usual practice of letting the segment structure rules be scattered through the P rules. Then the P rules operate on strings of matrices which have blanks in them, making it necessary to require that the P rules, as well as the MS rules, be subject to the well-formedness condition. This can be seen clearly in the following two examples, in which we arrive at specious simplifications (using each of the possible interpretations of rule application) in P rules which do not meet the well-formedness condition. The examples are actually concrete illustrations of the principles involved in (16)–(20) above.

[22] M will always be the output of rule R_{i-1} except when $i = 1$, in which case M will be a dictionary matrix.—Note that 'well-formed' names a specific, precisely defined formal property of grammars; it is NOT a general property which somehow reflects the intuitive meaning of the term.

[23] His section 2.15 (p. 57) seems to indicate that this condition was implicitly required.

[24] It might be argued that it doesn't matter if the formalism permits the use of rules which lead to specious simplifications as long as we never actually make use of such rules. This, however, is to misunderstand the purpose of a formal theory, which is, after all, to provide the most SPECIFIC and HIGHLY-STRUCTURED formalism with which it is possible to treat the phenomena in question. Only then can an explanation be reached of why the phenomena are organized in one way rather than another.

Note that, in these examples, we specifically assume that the segment structure rules are in the P rules, and thus that the input to the P rules contains blank entries.

Example A: Suppose we assume the sub-matrix interpretation of rule application, and suppose that in some language all [−Consonantal] segments are [+Voiced] and that, further, there is a P rule that affects only [+Voiced, +Consonantal] segments. If this P rule applies before the segment structure rule which makes all [−Consonantal] segments [+Voiced], and if we don't require well-formedness, then all we need to state in the structural description (SD) of this P rule is [+Voiced]: the rule will then apply to just the [+Voiced, +Consonantal] segments, since the [−Consonantal] segments, though really [+Voiced], have not yet been specified as [+Voiced]. But this is a specious simplification, an improper use of blanks. The formalism should require us to state [+Voiced, +Consonantal] in the structural description of this P rule (which is just what we do have to state if the whole grammar, including P rules, is well-formed). If we only state [+Voiced], then we are DEPENDING on the fact that the [−Consonantal] segments are blank for voicing at this point. If these segments had been specified beforehand with the correct value of the feature Voiced (i.e. [+Voiced]), the P rules would give the wrong results. This is exactly the same as saying that 'blank' (for the feature Voiced in [−Consonantal] segments) is functioning as a third value, distinct from '+' and '−'.[25]

Example B: The same kind of improper use of blanks in the P rules can be illustrated if we assume the distinctness interpretation of rule application. Consider a language in which the feature specification [−Voiced] is never predictable by a segment structure rule; this would be true if, for example, for every [−Voiced] segment there always occurred a corresponding [+Voiced] segment in the same environment. Suppose further that all vowels, liquids, and glides are [+Voiced]. Consider now a stage of the P rules before the [+Voiced] specifications have been inserted (by a segment structure rule) in vowels, liquids, and glides. If, at this stage, we want a rule R which applied to all the [+Voiced] segments (which constitute a natural class), then SD(R) will be [+Voiced], as expected. Yet if we wanted a rule R′ which applies to all the [−Voiced] consonants plus all the vowels, liquids, and glides (which constitute a much less natural class), we would still need only one feature specification in SD(R′), namely [−Voiced]. This is an intolerable situation: in any formalism where it can arise, the concept of natural class, otherwise so elegantly characterized in distinctive feature theory, cannot be captured. The problem, of course, is that a rule such

[25] It might be argued that it is correct to state a P rule such as the above, with only [+Voiced] in the structural description of the rule, on the basis of the following principle: a rule R should apply only to those segments which are non-redundantly specified for the features given in SD(R). As we can see from the above example, this principle could be put into effect by assuming the sub-matrix interpretation of rule application and by putting the segment structure rules at the end of the P rules. However, this would force us to have a system in which blank functions as a third feature value, distinct from '+' and '−'. Moreover it seems that in any actual grammar there are just as many cases in which we want to have a rule R apply to segments which are REDUNDANTLY SD(R).

as R′ applies to too many segments. Its structural description SD(R′) is [−Voiced], and yet it applies to vowels, liquids, and glides, which are really [+Voiced], as well as to [−Voiced] consonants. It can do this simply because these [+Voiced] specifications, being redundant, are not inserted until later. The result is that a rule such as R′, which supposedly represents generalizations about segments which have the feature specifications given in SD(R′), now applies to many segments which will never have these specifications, but which happen at this point to be blank for these specifications. Such a result destroys completely the meaning of phonological rules.

From these examples and many more like them, we see that as long as there are redundancy rules (the segment structure rules) scattered through the P rules, the problem of blank specifications arises in the P rules. To solve this problem, we could, of course, require well-formedness in the P rules, just as in §3.3 we solved a similar problem in the MS rules by requiring well-formedness. But then, to check a grammar for well-formedness, we must not only see how MS rules apply to dictionary matrices, but we must also see how P rules apply to the systematic phonemic representations of whole sentences. Since there is no upper bound on the length of sentences, it is clear that there is, in general, no finite procedure which will tell us whether a given grammar is well-formed; e.g. we never know when some transformational rule is going to produce a configuration of systematic phonemic matrices which is incomparable with the environment of some P rule.[26] It seems much more natural to make the requirement that the input to the P rules consist of fully specified matrices, for then the problem of well-formedness does not arise in the P rules. Well-formedness is then simply a condition on how MS rules apply and, as already mentioned, is easily checkable.

However, the problem of specious simplifications obtained through the improper use of blanks is not solved by the well-formedness condition. This condition, though it does prevent all improper use of blanks, is actually too strong. We will not show this directly, but will rather show that a weaker condition on grammars, the distinctness condition, is itself too strong.

3.5. The distinctness condition. The well-formedness condition has not been widely discussed. It has been much more common to assume the distinct-

[26] It should be noted that it is possible to define a condition on the P rules and systematic matrices which is finitely checkable and which solves the problems which the well-formedness condition is designed to solve. This condition is slightly stronger than the well-formedness condition. The well-formedness condition requires that no sequence of systematic phonemic matrices which is actually generated by the syntactic component of the grammar be incomparable with the structural description of the first P rule, that no configuration of matrices obtained by applying the first P rule to one of these sequences be incomparable with the structural description of the second P rule, etc. The new condition would require that the above conditions be met not only for all configurations of systematic phonemic matrices actually generated by the grammar, but for all possible configurations of systematic phonemic matrices. However, such a condition is too restrictive; it merely bypasses the problem of blank specifications, and does not come to terms with it. In fact we will see later that even the well-formedness condition is too restrictive in this sense.

NESS CONDITION on grammars as a means of preventing improper use of blanks (cf. Halle 1959:32, Chomsky 1965:81).[27] This condition requires that the dictionary matrices be pairwise distinct, i.e. that there be no pair D_1, D_2 of dictionary matrices such that D_1 is not distinct from D_2.[28] It is important to notice that 'distinct' is being used here in the technical sense defined in §3.3, and not as an informal equivalent of 'different' or 'distinguishable'. In particular, the distinctness condition is not to be taken as being a condition which, a priori, dictionary matrices must meet. In fact we will show below that this condition is too strong. For the moment we merely observe that every actual grammar that meets the well-formedness condition must also meet the distinctness condition; for, assuming well-formedness, it is apparent that any non-distinct pair of matrices must undergo exactly the same rules, and so could not emerge different from the MS rules. We also observe that a grammar which meets the distinctness condition need not meet the well-formedness condition; for example, the four rules of (16) and (17) violate well-formedness when they apply to the second and third matrices in (18), even though these matrices are distinct. Thus, well-formedness is a strictly stronger condition than distinctness. Finally, we note that the distinctness condition is easily checkable; we simply need to check something about each pair of members in a finite set, the set of dictionary matrices.

The distinctness condition was originally adopted to rule out the following particular kind of improper use of blanks. Suppose we have a language in which the feature Tense is predictable in consonants, [+Voiced] consonants being [−Tense] and [−Voiced] consonants being [+Tense]. Then we can let the ONLY difference between pairs of systematic phonemes like *t* and *d* in dictionary matrices be as follows:

$$(21)\qquad \begin{array}{l|c|c|} & t & d \\ \hline \text{Consonantal} & + & + \\ \hline \vdots & \vdots & \vdots \\ \hline \text{Voiced} & & + \\ \hline \text{Tense} & & \\ \hline \end{array}$$

Clearly, we are not using enough information to keep *t* and *d* apart in (21). Yet, if we use the distinctness interpretation of rule application and the rules in (22), we can predict the correct values of the features Voiced and Tense in *t* and *d* in (21).[29]

[27] The distinctness condition is not to be confused with the 'distinctness interpretation of rule application' (§3.3); both concepts, however, are based on the definition of 'distinct' pair of matrices given at the beginning of §3.3. Halle uses the term 'different from'.

[28] The distinctness condition is correct in not ruling out a situation where non-distinct SEGMENTS become distinct by the application of the MS rules if these segments occur in distinct matrices. In fact, this is exactly the situation when neutralized segments are represented by archiphonemes.

[29] This 'grammar', of course, does not obey the well-formedness condition, since it was chosen expressly to violate the weaker distinctness condition. This will be true of all the examples of this section.

(22)
$$[+\text{Consonantal}] \rightarrow [+\text{Tense}]$$
$$\begin{bmatrix} +\text{Consonantal} \\ +\text{Voiced} \end{bmatrix} \rightarrow [-\text{Tense}]$$
$$\begin{bmatrix} +\text{Consonantal} \\ +\text{Tense} \end{bmatrix} \rightarrow [-\text{Voiced}]$$

This intolerable situation is ruled out by the distinctness condition, since *t* and *d* were not kept distinct in the dictionary matrices. However, it is important to notice that there are other ways of ruling out this kind of case. For example, we could require that MS rules not be allowed to change feature values; since the second rule in (22) involves changing features in an essential way, this requirement would suffice. Alternatively, we could require that MS rules must use the sub-matrix interpretation of rule application, rather than the distinctness interpretation as in (22). In fact, we will show in §3.6 that these latter two requirements are essentially correct, whereas the distinctness requirement is not.

It is true that the distinctness condition happens to rule out many types of improper use of blanks in the MS rules (such as the one just discussed). However, unlike the stronger well-formedness condition, it is not sufficient to rule out ALL improper use of blanks. This follows since (using either of the two possible interpretations of rule application) the kind of specious simplification obtained in (16)–(20) could still be obtained if the matrices involved were merely subparts of distinct matrices. The same point can be made by viewing the rules in examples A and B of the preceding section as MS rules. The reason that the distinctness condition fails is that it has not come to terms with the problem, discussed in §3.3, that there IS no adequate definition of how rules apply if the rules must recognize blanks in any way.

Thus the distinctness condition is not strong enough. However, in another sense, the distinctness condition is too strong. These two facts indicate that it is a wholly inadequate condition to place on grammars.

There are many ways of showing that distinctness is too strong. In fact we have already pointed out (§2.2) that 'high level' features may be predictable in terms of 'low level' features, and that this prediction can only be made at the expense of violating distinctness; rule (10) is an example of such a prediction. The essence of this kind of case can be illustrated by the following example. Consider the matrices

(23)
$$\begin{array}{c|c|} \cline{2-2} f_1 & + \\ \cline{2-2} f_2 & \\ \cline{2-2} \end{array} \qquad \begin{array}{c|c|} \cline{2-2} f_1 & \\ \cline{2-2} f_2 & + \\ \cline{2-2} \end{array}$$

and the rules

(24)
$$[+f_1] \rightarrow [-f_2]$$
$$[+f_2] \rightarrow [-f_1]$$

Then, using the sub-matrix interpretation of rule application and the rules (24), the non-distinct matrices (23) become the distinct matrices

(25)
$$\begin{array}{c|c|}\cline{2-2} f_1 & + \\ \cline{2-2} f_2 & - \\ \cline{2-2}\end{array} \qquad \begin{array}{c|c|}\cline{2-2} f_1 & - \\ \cline{2-2} f_2 & + \\ \cline{2-2}\end{array}$$

But in no sense do we have here an improper use of blanks. That is, we are not using blanks in keeping the matrices in (23) apart, but rather the specifications $+f_1$ in the one and $+f_2$ in the other. In other words, we would a priori rule out (21), but not (23), as being instances where matrices are being kept apart with too little information; yet in each case the matrices involved are not distinct. Thus, the distinctness condition rules out desirable and undesirable cases indiscriminately.

It is possible to give a concrete case which illustrates the fact, suggested by the above remarks, that the distinctness condition is too strong. Consider Modern Standard Russian (MSR). According to Halle (1959), a grammar of MSR includes the following two MS rules:

(26)
$$+\begin{bmatrix}-\text{Consonantal}\\ -\text{Vocalic}\end{bmatrix} \begin{array}{c}\begin{bmatrix}\phantom{-\text{Consonantal}}\\ \phantom{+\text{Vocalic}}\end{bmatrix}\\ \downarrow \\ \begin{bmatrix}-\text{Consonantal}\\ +\text{Vocalic}\end{bmatrix}\end{array}$$

(27)
$$\begin{array}{c}\begin{bmatrix}\phantom{-\text{Consonantal}}\\ \phantom{+\text{Vocalic}}\end{bmatrix}\\ \downarrow \\ \begin{bmatrix}-\text{Consonantal}\\ +\text{Vocalic}\end{bmatrix}\end{array} \begin{bmatrix}-\text{Consonantal}\\ -\text{Vocalic}\end{bmatrix} \begin{bmatrix}+\text{Consonantal}\\ -\text{Vocalic}\end{bmatrix}+$$

Rule (26) says that a morpheme-initial glide must be followed by a vowel. Rule (27) says that a morpheme-final glide-consonant cluster must be preceded by a vowel. But now consider the forms *ijv* and *jiv*, which certainly are possible morphemes in MSR. What would the dictionary matrices for such forms be? Whatever these matrices are, we would like to be able to use rules (26) and (27) to make savings in them. If we did, then these matrices would be:

(28)
$$\begin{array}{l|c|c|c|} & i & j & v \\ \cline{2-4} \text{Consonantal} & & - & + \\ \cline{2-4} \text{Vocalic} & & - & - \\ \cline{2-4} \vdots & & & \end{array} \qquad \begin{array}{l|c|c|c|} & i & j & v \\ \cline{2-4} \text{Consonantal} & - & & + \\ \cline{2-4} \text{Vocalic} & - & & - \\ \cline{2-4} \vdots & & & \end{array}$$

(We ignore all other features, which are the same for each form since *j* and *i* differ only in the feature Vocalic.) However, the two matrices in (28) are not distinct. Since there are reasons to give rule (26) first, we conclude that to maintain distinctness we must write the dictionary matrix for *ijv* as:

(29)
$$\begin{array}{l|c|c|c|} & i & j & v \\ \cline{2-4} \text{Consonantal} & & - & + \\ \cline{2-4} \text{Vocalic} & + & - & - \\ \cline{2-4}\end{array}$$

But then we miss being able to make use of the generalization that ALL morpheme-final glide-consonant clusters, even those in three segment morphemes, are preceded by a vowel. If rule (27) had been first, a perfectly analogous problem would have arisen.

This example supports the conclusion that the distinctness condition is too strong, since this condition prevents us from making savings in the dictionary where they seem desirable. We are therefore led to reject both the distinctness condition and the stronger well-formedness condition as being inadequate devices for preventing the improper use of blanks in the MS rules. The distinctness condition is, of course, especially inadequate, since in some cases it is too restrictive while in others it is not restrictive enough. Clearly, however, SOME condition is necessary. In the next section we will see that there is a rather natural and obvious condition that has so far been overlooked.

3.6. THE TRUE GENERALIZATION CONDITION. In stating the new condition we will assume that the sub-matrix interpretation of rule application is used exclusively in the MS rules. Evidence in favor of this interpretation, rather than the distinctness interpretation, will be given below. We will also assume that all the redundancy rules are in the MS rules, so that the systematic phonemic matrices which emerge from these rules are fully specified. The problems involved when this assumption is abandoned have been fully discussed (§§1.6,3.4). Finally we will assume that each dictionary matrix D is simply a less fully specified version of the corresponding completely specified systematic phonemic matrix S; i.e., we assume that in each case D is a sub-matrix of S. These assumptions are merely corollaries of our decision to identify the concepts of redundancy rule and MS rule.

A grammar is said to meet the TRUE GENERALIZATION CONDITION if each MS rule represents a true generalization about the (fully specified) systematic phonemic matrices of the language. That is, given any (fully specified) systematic phonemic matrix S and any MS rule R, it must not be the case that SD(R) is a sub-matrix of S when SC(R) is not, for in such a case the rule R would be making a false statement about S.

The motivation for such a condition becomes clear when we recall that MS rules are intended to state redundancies in systematic phonemic matrices; we are simply requiring that each such statement be true of these matrices. Surely such a requirement must be implicit in any statement of redundancies, and to violate it would be highly unnatural. In fact, put in this form, the requirement is so obvious that it hardly needs to be stated at all.[30]

Despite the simplicity and naturalness of the true generalization condition, it and it alone suffices to prevent all specious simplifications and improper use of blanks. This can be illustrated in many ways. In grammars meeting the true generalization condition, a rule can never DEPEND for its proper application on

[30] The reason the naturalness and usefulness of this requirement have not been noticed previously is probably due to the fact that redundancy rules and feature changing rules have generally been intermingled in the P rules, and have not been formally distinguished. However, once the suggestion of this paper is adopted and the redundancy rules are considered as a separate set of rules (the MS rules), quite distinct from the P rules, then it becomes obvious that the true generalization requirement is merely part of what we mean by redundancy rules (cf. the historical discussion in §1.2).

there being blanks in matrices at the stage at which it applies, since we require that the rule also apply correctly to the systematic phonemic matrices, in which there are no blanks. Thus we note that this condition rules out the improper use of blanks illustrated in §3.4, example A.[31] In §3.3, (16)–(19), we gave an example, using the sub-matrix interpretation, where '0' functioned as a third feature value. The true generalization condition clearly rules out such a case, since the second rule in (16) does not represent a true generalization about the matrices in (19). In §3.5 we showed by means of several examples that the distinctness condition was too strong. The true generalization condition does not share this defect. In particular we note that the non-distinct matrices of (23) are now allowed, as desired, since the rules of (24) represent true generalizations about the matrices of (25). Also, we may now have the matrices of (28) in the Russian example, which are the desirable ones but which are not distinct; this follows since the rules of (26) and (27) each represent legitimate generalizations about Russian systematic phonemic matrices. Finally, notice that the true generalization condition does not allow the kind of improper use of blanks illustrated in (21) and (22), since the first rule in (22) is not true of all consonants.

In summary, the true generalization condition prevents improper use of blanks, but is not so strong that it prevents us from making certain generalizations. The condition is also easily checkable; we simply need to take each systematic phonemic matrix of the language and check to see that no MS rule makes a false statement about this matrix.

It is easy to see that, if the true generalization condition is met, MS rules will not change feature values, since each dictionary matrix is a sub-matrix of the corresponding fully specified systematic phonemic matrix, and the MS rules, which fill in the former to produce the latter, must make true statements about the latter. It is natural to ask if the true generalization condition might be replaced by the (weaker) requirement that MS rules do not change feature values. The following example shows that this is not possible. Consider the matrices A and B below:

(30)
$$\begin{array}{c|c|c|}
 & \text{A} & \text{B} \\
\hline
\text{f} & + & - \\
\hline
\text{g} & + & + \\
\hline
\text{h} & - & + \\
\hline
\end{array}
\qquad
\begin{array}{|c|c|}
\text{A}' & \text{B}' \\
\hline
+ & - \\
\hline
+ & \\
\hline
 & \\
\hline
\end{array}$$

If we use the sub-matrix interpretation of rule application and the rules [+g] → [−h] and [−f] → [+g, +h], applied in this order, then we see that we can let A′ and B′ be the dictionary matrices for A and B. However, the first rule applies properly only because the second has not applied and feature g is still blank in B′. Thus we have a specious simplification: we are able to get away with too

[31] There we used the sub-matrix interpretation of rule application and obtained specious simplifications in the segment structure rules. In that example, these rules were viewed as P rules; but in the present section we have returned to our assumption that such rules, being redundancy rules, are in the MS rules. The essence of the example is the same in either case.

general a statement of the first rule, which should have to be stated as [+f, +g] → [−h]. This shows that merely preventing rules from changing feature values is not sufficient to prevent specious simplifications, for the above rules do not change feature values. However, the first rule violates the true generalization condition since it is not true of the segment B. Thus this condition rules out the above case, as desired.

Though it is not necessary for rules to change feature values in order to obtain the specious simplification in the last paragraph, it is necessary that the two rules be ordered with respect to one another. If we require BOTH that MS rules be unordered AND that they do not change feature values, then no specious simplifications are possible. Moreover, it is not hard to see that these two requirements together are equivalent to the true generalization condition. This, in turn, is just a way of saying that the MS rules are exclusively redundancy rules, which was precisely what we argued in §1. Thus we have given two independent kinds of justification for this position, a formal one (in this section), and a linguistic one (in §1).

Note further that the true generalization condition gives an obvious and natural criterion for determining when two dictionary matrices are different enough to be kept apart; two dictionary matrices are properly distinguishable just in case the MS rules render them distinct, where 'distinct' has the formal meaning of §3.3. This follows since MS rules meeting the true generalization condition can never make improper use of blanks in rendering non-distinct matrices distinct.

In closing this section, we will give evidence supporting our decision to use the sub-matrix interpretation of rule application, rather than the distinctness interpretation. Recall that, in the sub-matrix interpretation of rule application, a rule R applies to a matrix M just in case the environment SD(R) of R is a submatrix of M. This interpretation means that R would still have applied to M if the blanks in M had been filled in beforehand in any arbitrary way; in other words, THE APPLICATION OF R IS INDEPENDENT OF ANY UNSPECIFIED FEATURES IN M. Such an interpretation seems intuitively more natural than the distinctness interpretation of rule application.

In this latter interpretation, rule R applies to a matrix M just in case SD(R) is not distinct from M. Thus a rule R might apply to a matrix M even if M would be distinct from SD(R) if it had all its redundant features filled in. All that is required is that M not be distinct from SD(R) at the point of application of R. However, as we noted in §3.4, example B, to have a rule R apply to matrices which are distinct (even though redundantly so) from SD(R) is an entirely unnatural situation. It not only places an artificial importance on the ORDER in which redundant features are filled in, but also destroys the crucial idea that a rule R represents a generalization about, and thus applies to, exactly those matrices which have the features of SD(R). To illustrate this point further, consider the following example of improper use of blanks which would be attainable under the distinctness interpretation of rule application, even if the true generalization condition were met. Suppose there is a language whose strident segments are exactly the continuant consonants; and suppose we mark the continuant consonants in the dictionary [0Continuant, +Strident] (i.e. as blank for continuancy), and the stop consonants as [0Continuant, 0Strident] (i.e., we do not mark the stops for either continuancy or stridency). Then if we have rules

(31)
(1) [−Strident] → [−Continuant]
(2) [−Continuant] → [−Strident]
(3) [+Continuant] → [+Strident]
(4) [+Strident] → [+Continuant]

(each of which represents a true generalization about consonants), and if we use the distinctness interpretation of rule application, we can have the following derivation:

(32)

	D		1		2		3		4	
	s	*t*	*s*	*t*	*s*	*t*	*s*	*t*	*s*	*t*
Continuant				−		−		−	+	−
Strident	+		+		−	−	+	−	+	−

Here the entry under 'D' represents the values of the features Continuant and Strident in the segments *s* and *t* in the dictionary; the entry under '1' represents the same information after the first rule in (31) has applied, etc. Thus we see that, even if the true generalization condition is met, the distinctness interpretation of rule application allows us to keep two segments apart in the dictionary using only one specified feature value—clearly an improper use of blanks. This supports our decision to use the sub-matrix interpretation of rule application, in which such an improper use of blanks is not possible.

MORPHEME STRUCTURE CONDITIONS

4.1. THE GENERAL APPROACH. The point of this paper so far is that we must take seriously the idea that the MS rules of a language are the rules which state the redundancies at the systematic phonemic level. We have seen that the MS rules are most naturally viewed as an unordered set of statements which represent true generalizations about the fully specified systematic phonemic matrices, and which may be used (as an unordered set) to fill in the redundant feature values in dictionary matrices, each of which is a less fully specified version (a sub-matrix) of its corresponding systematic phonemic matrix. In this section we will see that the most natural way of formulating such statements is not in terms of rules at all, and we will show how MS rules can be replaced by a new device, MS CONDITIONS; we use rules (like the P rules) only to map one level onto another, and 'conditions' to state redundancies at a given level. MS conditions not only avoid, in a natural way, all the formal problems discussed in §3, they also allow us to state situations which arise in natural languages but which are not easily stated in terms of rules.

In many ways, MS rules and MS conditions are very similar. In particular, we can talk about segment structure CONDITIONS and sequence structure CONDITIONS in the same way that we talked about the corresponding kinds of RULES in §1.3

and §1.4. The discussion in §§1.5, 1.6 will also carry over essentially without change into MS conditions, and we will retain the decisions made there.

Briefly, the proposed system consists of the following two parts. First, there is an UNORDERED set M of MS conditions which defines, in a manner specified in detail in §§4.2–4.6 a set M(U) of FULLY SPECIFIED matrices, where each matrix in M(U) has a number of rows equal to the number of distinctive features in the language in question,[32] and where each matrix in M(U) has i columns for some $1 \leqq i \leqq \lambda$, where λ is the length, in segments, of the longest morpheme of the language. The set M(U) contains the systematic phonemic matrix of every morpheme of the language; it also contains matrices for forms which, though not in the language, violate no constraints of the language. In brief, M(U) contains matrices for all the 'possible' morphemes of the language (cf. §1.4).

The second part of the proposed system is a PROCESS OF SELECTION. For each morpheme m, the incompletely specified dictionary matrix D_m of m SELECTS, in a manner described in §4.7, the completely specified systematic phonemic matrix S_m of m from the set M(U). Thus the process of selection provides a method for filling in redundant information in dictionary matrices. Recalling the discussion in §1.1 regarding the relation between redundancy and constraints, we can summarize the proposed system by saying that the MS conditions give statements of constraints, while the process of selection uses these statements to predict redundant feature values. Thus, statement of constraints and prediction of redundancies, though intimately related, are given as separate processes.

We will now describe the system of MS conditions in detail, first showing how the MS conditions define the set M(U), and then showing how the dictionary matrix of each morpheme selects the corresponding fully specified systematic phonemic matrix of the morpheme from this set.

4.2. MS CONDITIONS. Suppose we are dealing with a language with n distinctive features, in which the longest morpheme has λ segments. Let U be the set of all FULLY SPECIFIED matrices with n rows, with entries '+' and '−' and with no more than λ columns. Then U has 2^n members with one column, $(2^n) \times (2^n) = 2^{2n}$ members with two columns ..., and $2^{\lambda n}$ members with λ columns. Thus U is finite. In general, an MS CONDITION is a finite statement of a property shared by some, but not all, matrices in U. Thus 'has two segments (columns)', 'begins with a [+Consonantal] segment', 'has no two consecutive [−Vocalic] segments', etc., are all examples of MS conditions. It is easy to see that each MS condition divides the set U into two parts, one part consisting of all matrices which possess the property stated in the MS condition, and another part consisting of all other matrices in U. We will say that an MS condition C ACCEPTS a matrix in U if M has the property stated in C, and that C REJECTS M otherwise.

Instead of allowing any statement of properties of matrices to serve as an MS condition, we will restrict ourselves in what follows to MS conditions expressed in rather restricted standard form.[33] Specifically we will be considering three

[32] That is, systematic phonemic matrices, dictionary matrices, and matrices in M(U) all have the same number of rows.

[33] As always, the particular 'standard form' chosen is intended to represent the most restricted form in which all the relevant generalizations can be expressed.

kinds of MS conditions: 'if-then' conditions, 'positive' conditions, and 'negative' conditions, and we will require that all MS conditions be expressible in one of these three forms.

4.3. If-then conditions. An if-then condition C is a pair of matrices I(C) and T(C), the 'if' and the 'then' part of the condition respectively, where I(C) and T(C) are each incompletely specified matrices which have n rows (one for each distinctive feature) and entries '+', '−', or no entry (blank). Further, I(C) and T(C) have the same number of columns and are disjoint (for definition, see §3.2). The if-then condition C has the following interpretation: for all matrices M in U (see §4.2) such that I(C) is a sub-matrix of M, C accepts M if T(C) is also a sub-matrix of M, and C rejects M if T(C) is distinct from M; if I(C) is distinct from M, then C accepts M regardless of what T(C) is. This interpretation is well defined, since any incompletely specified matrix such as I(C) or T(C) is either distinct from or a sub-matrix of any matrix M in U; the matrices in U are fully specified, and thus it is not possible for either I(C) or T(C) to be incomparable with M. Intuitively, the if-then condition C says that if a matrix M in U meets condition I(C), then M must also meet condition T(C) if it is to be accepted by C; the condition C says nothing about matrices M which don't meet condition I(C) since it accepts all such matrices indiscriminately. The formal similarity between the definition of 'if-then condition' and the definition of 'MS rule' in §3.2 should be apparent; in fact, every MS rule R has a direct interpretation as an if-then condition C and conversely, with SD(R) corresponding to I(C) and SC(R) corresponding to T(C). However, the functional difference between an if-then condition and an MS rule should be kept clearly in mind. The former is a statement which defines a subset of the set U of fully specified matrices, namely the subset consisting of those matrices in P that it accepts. The latter is an instruction for filling in blanks in matrices with '+' and/or '−'.

We show, by example, how we will write an if-then condition:

$$
(33)\qquad
\begin{array}{lll}
\mathrm{I(C)} \quad + \ [+\text{Consonantal}] & & \begin{bmatrix} +\text{Consonantal} \\ -\text{Vocalic} \end{bmatrix} \\
 & & \quad\downarrow \\
\mathrm{T(C)} & \begin{bmatrix} -\text{Vocalic} \\ -\text{Grave} \\ -\text{Compact} \\ +\text{Continuant} \end{bmatrix} & [-\text{Continuant}]
\end{array}
$$

The condition will reject all matrices in U which begin with a [+Consonantal] segment followed by a true consonant and which do not also meet the condition T(C). This condition is valid in a language such as English, where, in any initial cluster of true consonants, the first consonant is *s* and the second is a stop ('sphere' and 'sphinx' being exceptions).

As another example, an if-then condition of the form

$$(34)\qquad \begin{array}{rl} \text{I(C)} & [-\text{Consonantal}] \\ & \downarrow \\ \text{T(C)} & \begin{bmatrix} +\text{Voiced} \\ +\text{Continuant} \\ -\text{Strident} \end{bmatrix} \end{array}$$

would reject all matrices in U containing [−Consonantal] segments which are [−Voiced], or [−Continuant], or [+Strident]. (33) and (34) are sequence structure and segment structure conditions, respectively.

4.4. Positive conditions. The second kind of MS condition is a 'positive' condition. Each positive condition consists simply of an incompletely specified matrix P(C). Its interpretation as a MS condition is straightforward: all matrices in U of which P(C) is a sub-matrix are accepted, all other matrices in U are rejected. Thus, for example, suppose all the morphemes in some language are of the form

$$(35)\qquad \text{C(L)V}\left(\begin{Bmatrix}\text{C}\\ \text{L}\end{Bmatrix}\right)$$

(where L represents a liquid). Then we would have the positive redundancy condition

$$(36)\ \text{P(C)} + \begin{bmatrix}+\text{Consonantal}\\ -\text{Vocalic}\end{bmatrix}\left(\begin{bmatrix}+\text{Consonantal}\\ +\text{Vocalic}\end{bmatrix}\right)\begin{bmatrix}-\text{Consonantal}\\ +\text{Vocalic}\end{bmatrix}[(+\text{Consonantal}])+$$

The condition (36) actually stands for the four positive conditions obtainable from it by considering all possible combinations of optional elements. It accepts only those matrices in U which have the permitted form. Note that (36) is a sequence structure condition. A positive segment structure condition would never be used, since, of two features mentioned in such a condition, one would necessarily be completely dependent on the other; thus it would be non-distinctive and not mentioned in the grammar.

4.5. Negative conditions. The third and final kind of MS condition is a 'negative' condition. Like positive conditions, each negative condition C consists of a single incompletely specified matrix; in this case we will denote the matrix by the symbol N(C). The interpretation of a negative condition C is that all matrices in U of which N(C) is a subset are rejected, all other matrices are accepted. As an example, consider a language which has systematic phonemes č š and ǰ but no ž. This situation is described by the negative condition

$$(37)\qquad \text{N(C)}\ \sim\begin{bmatrix} -\text{Vocalic}\\ +\text{Compact}\\ -\text{Grave}\\ +\text{Continuant}\\ +\text{Voiced}\end{bmatrix}$$

Here the symbol (∼) is the sign of negation. The condition (37) accepts just those matrices in U which contain no voiced continuant palatal segments. It is a segment structure condition. We will give an example below (§5.5) of a negative sequence structure condition.

4.6. THE PROCESS OF SELECTION. We have shown the formal nature of if-then conditions, positive conditions, and negative conditions, and also how they are to be interpreted. Specifically, we have shown how each of the three kinds of conditions ACCEPTS certain kinds of matrices in P. A grammar of each natural language will have, in place of a set of MS rules, an unordered finite set M of MS conditions. This set will include, in general, conditions of each of the three types. The set of all matrices m in U, such that m is accepted by EVERY MS condition in M, is well defined; we call this set M(U).

Since each MS condition in M represents a generalization about the morphemes of the language, it follows that the set M(U) represents all matrices which violate none of these generalizations. Moreover, an evaluation measure will guarantee that, for every significant generalization that can be made about the morphemes of the language, there will be a corresponding MS condition.[34] In short, the set M(U) is exactly the set of POSSIBLE morphemes of the language. The segment structure conditions in M will guarantee that M(U) contains only those matrices whose columns are systematic phonemes of the language; the sequence structure conditions in M will guarantee that no sequential constraints of the language are violated in matrices of M(U). The set M of MS conditions may thus be thought of as filtering out, from the set U of all matrices, those matrices which do not form possible morphemes of the language, leaving the set M(U).

We said in§4.1 that a system of redundancy conditions, as applied to a language L, consists of two parts: first, a set M of MS conditions which defines a set M(U) of fully specified matrices, where M(U) contains the systematic phonemic matrix of every possible morpheme of L; and second, a process of selection whereby the dictionary matrix D_m of each morpheme m of L selects the completely specified systematic phonemic matrix S_m of m from the set M(U). In §§4.1–4.5 we have outlined the operation of the first part. We must now proceed to the second part, the description of the process of selection.

Suppose we are given a language L and a set M of MS conditions written for the morphemes of L. Then the set M(U) is well defined; in particular, it contains the systematic phonemic matrix S_m of every morpheme m in L. We will show how the set M(U) takes the place of the MS rules of the previous theory in filling in blanks in dictionary matrices. Consider a partially specified matrix D with any number of columns and with the same number of rows as matrices in M(U); consider also any matrix X in M(U). Such a matrix X is fully specified. D is said to SELECT X if D is a sub-matrix of X.[35] Clearly, it is possible for D to select more than one matrix from M(U), and, in general, the more blanks D has, the more matrices it will select. In the limiting case where D has no specifications at all (is completely blank), it will select every matrix in M(U), since in this case

[34] This evaluation measure is similar to the one discussed in §1.4 for MS rules; see also §5.2.

[35] Notice that, if D selects X, then D and X must have the same number of columns; cf. §3.3. Since X is fully specified, an equivalent definition of selection would be that D selects X if D and X are not distinct.

D is a sub-matrix of every matrix. On the other hand, if D is fully specified and is itself a member of M(U), then it will select just one matrix from M(U), namely the one identical to itself. If D is fully specified and is not a member of M(U), then it will select no members of M(U).

We are interested in those partially specified matrices D which select exactly one matrix from M(U). Such matrices D will serve as dictionary matrices, and the process by which they select a fully specified matrix from M(U) will be the process, analogous to the application of the MS rules, by which the blanks in dictionary matrices are filled.

As an illustration of this process, consider a language which has the if-then condition (34). For this language the only [−Consonantal] segments that appear in M(U) are also [+Voiced], [+Continuant], and [−Strident]; all [−Consonantal] segments that are [−Voiced] or [−Continuant] or [+Strident] are rejected by (34). Thus, in representing [−Consonantal] segments in dictionary matrices, we enter a value for the feature Consonantal (and for any other non-redundant features) but can leave blank the values of the features Voiced, Continuant, and Strident. To see in detail why this is possible, consider any dictionary matrix D which has a [−Consonantal] segment as, say, its first segment. Suppose D selects the matrix S from M(U). By the definition of selection, D must be a sub-matrix of S, and thus the first segment of S is also [−Consonantal]. Since S is in M(U), the condition (34) guarantees that this [−Consonantal] segment is also [+Voiced, +Continuant, −Strident], and thus that these three features have been correctly selected for D.

To illustrate further, consider a language which has the if-then condition (33). In any morpheme of this language that begins with the feature configuration I(C) of (33), we can leave blank the features of T(C) in the dictionary matrix, since the condition (33) guarantees that such a dictionary matrix will select a matrix from M(U) which has exactly the features of T(C).

Any language which has the positive condition (36) has morphemes which are two, three, or four segments long. In the dictionary matrix of a two-segment morpheme, we can leave the features Consonantal and Vocalic completely blank, since (36) guarantees that M(U) for this language has the proper value for these features in each of its two-segment matrices. In the dictionary matrix of a four-segment morpheme, we need only to indicate the value of the feature Vocalic in the last segment (to distinguish between a C and an L, both of which occur in this position); all remaining values of Consonantal and Vocalic may be left blank. Since (36) allows three types of three-segment morphemes, CLV, CVC, and CVL, we must indicate the values of the features Vocalic and Consonantal in sufficient places to distinguish these three types, all of which occur in M(U). There are several ways of doing this; for example, either a [−Consonantal] in the third segment or a [+Consonantal] in the second segment tells us that we are dealing with a CLV morpheme, and for such a morpheme no other indications for the features Consonantal or Vocalic need be made in the dictionary.

In a language which has the negative condition (37), we need to indicate š for the feature Continuant but not for the feature Voiced in dictionary matrices;

(37) guarantees that each [+Continuant] palatal in M(U) is [−Voiced]. Similarly, ǰ must be indicated for the feature Voiced but not for the feature Continuant.

Finally, note that non-distinct dictionary matrices can select distinct matrices from M(U). For example, the non-distinct matrices in (23) would select the distinct matrices (25) if we included, in the set M of MS conditions, if-then conditions corresponding to the rules of (24). As pointed out in §3.5, the lack of a distinctness requirement is an advantage.

These examples, which illustrate the process of selection, complete the description of the system of MS conditions. We have seen how MS conditions replace MS rules as a device for predicting redundant information in systematic phonemic matrices. In the next section we will discuss further details of the new system; we will also attempt to demonstrate its superiority over MS rules.

PROPERTIES OF THE SYSTEM

5.1. THE FORMALISM. We saw in §3 that one of the major problems in MS rule theory was to prevent the improper use of blanks; this problem was solved by imposing the true generalization condition on MS rules. Now that we have described MS condition theory, it is natural to ask whether or not improper use of blanks can be obtained through MS conditions. It is easy to see that the answer to this question is no, by looking at the definition of the set M(U) by the MS conditions and at the process of selection of fully specified matrices from M(U) by dictionary matrices.

Consider a set M of MS conditions for some language. The function of M is to define, from the set U of all matrices, the set M(U) of matrices which violate none of the constraints of the language. However, since U consists of matrices which have no blanks in them at all, it is clear that this definition can involve no improper use of blanks. Further, it is clear that each MS condition in M must state a true generalization about the morphemes of the language; violating this requirement would never lead to specious simplifications, only to an incorrect grammar. Thus, the true generalization condition of §3.6 is automatically met by MS conditions.

Consider now the process of selection from M(U). Recall that, by the definition of dictionary matrix in a system of MS conditions, each dictionary matrix must select exactly one (fully specified) matrix from M(U). Since the set M(U) consists only of the possible morphemes of the language, it follows that each dictionary matrix must contain precisely the amount of information needed to identify a morpheme GIVEN THAT IT IS A POSSIBLE MORPHEME OF THE LANGUAGE. This, of course, is precisely what is desired. It means, on the one hand, that dictionary matrices need not contain extra specifications in order to meet requirements such as distinctness or well-formedness, which we saw in §3 to be unnatural and unmotivated. On the other hand, it means that dictionary matrices cannot contain so few specifications that they are inadequately kept apart, for a matrix with too few specifications would select more than one matrix from M(U).

5.2. THE EVALUATION MEASURE. Each MS condition C in a set M of MS conditions in a grammar must contribute to making the set M(U) smaller; if C did not so contribute, it would be of no use in the grammar and would be discarded. We can see how each MS condition decreases the size of M(U): the more MS conditions there are, the fewer matrices are accepted by all conditions, and M(U) is just the set of matrices accepted by every condition. Also, the smaller the set M(U) is, the smaller is the number of specifications which have to be made in dictionary matrices: when M(U) is small, then dictionary matrices have less to select from. Thus, for every MS condition C, we can talk about how many feature specifications C allows us to save in any dictionary matrix D: this is just the number of additional specifications that D would require to select a single matrix from the set M′(U) where M′ is the set M with the condition C removed. Since we can talk about the number of feature specifications each MS condition saves in the dictionary, it follows that we can use the same kind of evaluation procedure as that used for MS rules, since this procedure is based on the number of feature specifications that each MS rule saves us in the dictionary.[36]

For any subset M′ of the full set M of MS conditions in any language, we can talk about the set M′(U). This set contains the set M(U) and also those matrices rejected only by the MS conditions in M but not by those in M′. This suggests a way of handling exceptions to the MS conditions. If a morpheme is an exception to the MS condition C_n, then we can simply mark this morpheme [−Condition C_n] in the dictionary, and agree that a morpheme so marked will select not from the set M(U), but rather from M′(U), where M′ is the set M minus the condition C_n. Selecting from the larger set M′(U) will require that a dictionary matrix be more fully specified, and this is exactly as desired: exceptional morphemes should require fuller specification. The treatment of morphemes which are exceptions to more than one MS condition can be handled in a perfectly analogous fashion.

5.3. The need for positive conditions. We give here an example of a kind of morpheme structure which is highly unlike that of any natural language, but which is describable in a quite natural way by MS rules. Such an example is surely devastating to a theory of MS rules. Thus, consider a 'language' which has the following four types of morphemes:

(38) LCVG CVGL VGLC GLCV

where L = Liquid = [+Consonantal, +Vocalic], C = Consonant = [+Consonantal, −Vocalic], V = Vowel = [−Consonantal, +Vocalic], G = Glide = [−Consonantal, −Vocalic]. That is, L, unless final, is always followed immediately by C; C, unless final, is always followed directly by V; etc. Such cyclical structure surely does not occur in natural languages. However, observe how naturally a set of MS rules can describe this structure:

(39)
$$[\alpha\text{Consonantal}]\quad[\qquad\qquad]\quad\begin{array}{c}[\qquad\qquad]\\ \downarrow\\ [-\alpha\text{Consonantal}]\end{array}$$

[36] See Halle 1964b:338–40. This evaluation procedure is the means by which we determine what the significant generalizations of a language are. Of course to say, as this evaluation measure does, that feature values in rules have the same cost as feature values in dictionary matrices, is a strong claim, and is by no means obviously correct. Perhaps some other ratio of importance of feature values in rules to feature values in the dictionary is needed.

$$(40) \qquad [\alpha\text{Vocalic}] \quad \underset{\substack{\downarrow \\ [-\alpha\text{Vocalic}]}}{[\qquad\quad]}$$

The MS rules (39) and (40) completely characterize the four structures in (38) and only these. Indeed, the dictionary entries for these four structures can be represented as follows for the features Consonantal and Vocalic:

$$(41) \quad \begin{array}{l}
+\begin{bmatrix}+\text{Consonantal}\\+\text{Vocalic}\end{bmatrix}\begin{bmatrix}+\text{Consonantal}\\ \ \end{bmatrix}\begin{bmatrix} \ \\+\text{Vocalic}\end{bmatrix}\begin{bmatrix}\qquad\quad\\ \ \end{bmatrix}+\\
+\begin{bmatrix}+\text{Consonantal}\\-\text{Vocalic}\end{bmatrix}\begin{bmatrix}-\text{Consonantal}\\ \ \end{bmatrix}\begin{bmatrix} \ \\-\text{Vocalic}\end{bmatrix}\begin{bmatrix}\qquad\quad\\ \ \end{bmatrix}+\\
+\begin{bmatrix}-\text{Consonantal}\\+\text{Vocalic}\end{bmatrix}\begin{bmatrix}-\text{Consonantal}\\ \ \end{bmatrix}\begin{bmatrix} \ \\+\text{Vocalic}\end{bmatrix}\begin{bmatrix}\qquad\quad\\ \ \end{bmatrix}+\\
+\begin{bmatrix}-\text{Consonantal}\\-\text{Vocalic}\end{bmatrix}\begin{bmatrix}+\text{Consonantal}\\ \ \end{bmatrix}\begin{bmatrix} \ \\-\text{Vocalic}\end{bmatrix}\begin{bmatrix}\qquad\quad\\ \ \end{bmatrix}+
\end{array}$$

The rules (39) and (40) would fill in all the other feature values (for the two features considered here) in each of the forms in (41).

Thus we see that MS rules treat the unnatural situation (38) in a simple way. On the other hand, no POSITIVE MS condition could possibly describe the structure in (38) (other than by listing in a four-way disjunction each of the possibilities, a solution which must be disallowed since, using similar techniques, positive conditions can describe ANY situation). However, it is true that the theory of MS conditions could handle the structure in (38) using IF-THEN conditions equivalent to (39) and (40) (since, as we have seen, if-then conditions and MS rules are always interconvertible), and this seems to take the force out of our argument. Yet this can be avoided if, in the theory of MS conditions, we REQUIRE that positive conditions be used in stating restrictions on syllable structure, that is, in stating restrictions involving the features Consonantal, Vocalic and perhaps Obstruent. This requirement is motivated by the fact that these features are typically interrelated in different ways than the other features, a fact which implies that they should be treated formally in different ways.[37] In summary, then, we may accept the above argument that positive MS conditions succeed where MS rules fail. This argument seems to be substantiated by the fact that the 'syllable structure' of many languages can only be described by a relatively complex set of MS rules which not only miss generalizations but which also cannot capture any restrictions on the length of morphemes which may exist. Positive MS conditions on the other hand, seem ideally suited to describing such syllabic structure.

5.4. THE NEED FOR NEGATIVE CONDITIONS. In this section we show that negative MS conditions, but not MS rules, can provide a solution to a well-known problem in Proto-Indoeuropean (Lehmann 1952:17). Morphemes of the type

[37] This difference in the behavior of different features is clearly the kind of 'formal property' which, as we noted in §2.4, would be necessary in any non-ad-hoc characterization of a hierarchy among the features. Perhaps we could even look for a formal definition of each feature in terms of the particular way it behaves in morpheme structure and in the P rules. This would be similar to the formal definitions of Noun and Verb suggested by Chomsky (1965:115–6), in terms of the different ways nouns and verbs behave with respect to subcategorization.

{obstruent, vowel, obstruent} were severely restricted; using *e* as a sample vowel, *p b bh* as sample initial obstruents, and *t d dh* as sample final obstruents, these restrictions can be summarized in the following table:

(42)	*pedh	ped	pet
	bedh	*bed	bet
	bhedh	bhed	*bhet

That is, if the initial consonant is a voiceless non-aspirate, then the final consonant cannot be an aspirate, etc. Consider the negative MS condition

$$(43)\quad \sim + \begin{bmatrix}\alpha\text{Voiced}\\ \beta\text{Aspirate}\end{bmatrix}\begin{bmatrix}\quad\quad\quad\quad\end{bmatrix}\begin{bmatrix}-\beta\text{Voiced}\\ -\alpha\text{Aspirate}\end{bmatrix}+$$

This condition rules out all forms where the voicing of the initial segment disagrees with the aspiration of the final segment and where the voicing of the final segment disagrees with the aspiration of the initial segment; that is, it rules out all forms which show NEITHER an assimilation of the aspiration of the final segment to the voicing of the initial segment NOR an assimilation of the aspiration of the initial segment to the voicing of the final segment. And, in fact, the forms ruled out are the desired ones, i.e. the forms starred in (42).[38] Thus the negative condition (43) accounts exactly for the facts summarized in (42). Further, it is not hard to see why it is in principle impossible to write MS rules to account for these facts. This follows since, as noted above, the generalization involved in (42) is that EITHER one OR another (or possibly but not necessarily both) of two assimilations must take place (that is, that all forms in which neither of the assimilations takes place are ruled out); yet, even if we wrote an MS rule which corresponds to each of these two assimilations, we would have no way of stating that at least one but not necessarily both of them must apply.

It is of course necessary that many examples of negative conditions be given if their introduction into the theory is to be motivated. An important line of research would be to discover to what extent negative conditions describe situations which occur in natural languages.

5.5. THE ROLE OF BLANKS. MS rules and MS conditions have one characteristic in common; in each we have chosen to formalize the notion of 'redundant' feature value by saying that a feature value is redundant in a certain environment just in case the feature value, when left blank in this environment, can be determined by some finite procedure. In the case of MS rules this procedure consists of applying all the rules that fit this environment, until finally one of these rules inserts the value of the feature in question; in the case of MS conditions, the procedure consists simply of looking at the set of matrices 'accepted' by the MS condition, since every matrix in this set which has the environment in question also has the proper value of the feature in question. Thus, 'redundant' for us has meant 'can be left blank and later predicted'.

Suppose we proceed differently and abandon the idea of having a dictionary matrix for each morpheme which is less fully specified than the systematic pho-

[38] The condition (43) also rules out forms such as *pheth*, but this is all right since voiceless aspirates must be ruled out anyway.

nemic matrix of the morpheme. In fact, let us abandon (but for the moment only) the whole notion of dictionary matrix, and talk only about the fully specified systematic phonemic matrices, regarding these both as the matrices to be listed in the dictionary and as the matrices which enter the P rules. Still, let us retain in each grammar the set of MS conditions that state (in the manner shown in the previous section) the constraints that exist on the morphemes of the language. Now at first glance it may seem that we cannot do this, since we could have no way of knowing what the right set of MS conditions for a given language is. As long as there are dictionary matrices with blanks, then we have an evaluation measure which tells us what the best set of MS conditions is; it is, essentially, the shortest set of MS conditions that allows us to leave the greatest number of blanks in dictionary matrices (but cf. fn. 36). If there are no dictionary entries with blanks, then this measure means nothing.

Looking deeper into the matter, however, we see that even if we are to talk only about sets of fully specified systematic phonemic matrices and sets of MS conditions which state the constraints on these matrices, it is still possible to talk about the 'right' set of conditions for a given set of matrices; that is, there is still a meaningful evaluation procedure. Suppose we define the WEIGHT of MS condition C, with respect to a systematic phonemic matrix S, as the maximum number of feature specifications that could be removed from S provided only that, from the resulting reduced matrix R and the MS condition C, we be able to reconstruct S.[39] Clearly, if an MS condition C does not apply to a systematic phonemic matrix S, then the weight of C with respect to S is zero.[40] If an if-then MS condition C applies to a systematic phonemic matrix S, then the weight of C with respect to S is the number of specifications in T(C), the 'then' part of C. If a negative MS condition C applies to a systematic phonemic matrix S, then the weight of C with respect to S is always the number '1'. If a positive MS condition C applies to a systematic phonemic matrix S, then the weight of C with respect to S is the number of specifications in P(C).[41] The crucial idea is that the weight of an MS condition with respect to the various systematic phonemic matrices tells us how general a statement this condition makes about these matrices. This notion of generality of MS conditions (and, previously, of MS rules) is really the important one, and with it we can define an evaluation measure for MS conditions that does not count blanks in dictionary matrices. This is done as follows.

[39] The meaning of 'reconstruct' should be clear given the discussion in §4.6. To be explicit, S should be the unique matrix selected by R from the set C(U) (the set of all matrices in U accepted by C).

[40] The notion of a MS condition applying (or not applying) to a systematic phonemic matrix S is intuitively clear. To be precise, we can say: (1) an if-then condition C applies to S just in case I(C) is a sub-matrix of S; (2) a positive condition C applies to S just in case P(C) is a sub-matrix of S; (3) a negative condition C applies to S just in case we can switch the value of ONE feature in N(C) (any one feature) and arrive at a matrix $N^1(C)$ which is a sub-matrix of S.

[41] Actually, this is the number m_1 of specifications in P(C) minus the number m_2 of specifications needed to keep P(C) distinct from all other positive conditions, if any, having the same number of columns as P(C), since clearly we can save at most m_1–m_2 specifications in a matrix using P(C). See (36) and the later discussion of (36) in §4.6.

Suppose we have a language L with the set SP of systematic phonemic matrices and with a proposed MS condition C. Define the GENERALITY INDEX of C with respect to SP as the sum of the weights of C with respect to each member of SP. Also, define the STATEMENT COST of C as the number of feature values needed to state C. Then we can agree to include C in the grammar just in case its statement cost is less than its generality index (cf. fn. 36). This provides a meaningful evaluation measure.[42] Further, it is not hard to see that this measure gives essentially the same results as the usual evaluation measure, which depends on the existence of dictionary matrices in which the redundant feature values have been left blank.

The proposed solution also removes a certain kind of arbitrariness which was inherent in the practice of leaving redundant feature values blank in the dictionary. For example, if, in some environment E, the value [+f] implies the value [+g] and the value [+g] implies the value [+f], then it would be arbitrary which value we actually choose to indicate in the dictionary; since the MS conditions would rule out both of the feature combinations [+f, −g] and [−f, +g] in the environment E, we would need to indicate either [+f] or [+g] but not both.[43] The arbitrariness stems from the decision to commit ourselves as to whether f is predictable from g or g from f, this decision being involved in any solution where redundant feature values are left blank in dictionary matrices. Actually, however, this decision is not well motivated, since the correct statement may be simply that the values of f and g are interrelated. In fact, we can profitably draw a general conclusion regarding redundancy from this discussion. That is, to say that a certain fully specified matrix is highly redundant in some language is actually to say that many of its feature values are interrelated in ways determined by the constraints of the language, and it is simply the statement of these constraints (in the MS conditions) which constitutes the most natural characterization of the redundancy of the language. Once these constraints have been stated, it is true that they may be utilized (as described in §4.6) in giving dictionary representations their most economical form; but this is a secondary fact, and these redundancy-free representations play no real role in a theory of redundancy.[44]

[42] To make this evaluation procedure meaningful, we must make the obvious requirement that each MS condition actually contributes to making the set M(U) smaller, and does not merely repeat the effect of other MS conditions.

[43] Of course in MS RULE theory there is this same arbitrariness, but here this is based on a more serious arbitrariness, namely the choice between $[+f] \rightarrow [+g] / E$ and $[+g] \rightarrow [+f] / E$ as the rule for formulating the constraint in question.

[44] I owe a special debt to Morris Halle, who gave the encouragement and direction necessary for the writing of this paper. Also extended comments on an earlier draft by Barbara Hall Partee, David Stampe, and John Ross have led to considerable improvements. Finally, I am grateful to Manfred Bierwisch, Richard Carter, Noam Chomsky, Erica Garcia, Kalon Kelley, Chin-W. Kim, Paul Kiparsky, George Lakoff, G. Hubert Matthews, P. Stanley Peters, and Paul Postal for comments and discussion.

This work was supported in part by the Joint Services Electronics Program under Contract DA36-039-AMC-032000E; in part by the National Science Foundation (Grant GP-2495), the National Institutes of Health (Grant MH-04737-05), the National Aeronautics and Space Administration (Grant SsG-496), and the U.S. Air Force (ESD Contract AF19 (628-2487).

REFERENCES

CARROLL, JOHN B. 1961. The study of language. Cambridge, Mass., Harvard University Press.

CHERRY, COLIN. 1961. On human communication. New York, Wiley.

CHOMSKY, NOAM. 1957. Review of Jakobson and Halle, Fundamentals of language. IJAL 23.234–42.

——. 1964. Current issues in linguistic theory. The Hague, Mouton.

——. 1965. Aspects of the theory of syntax. Cambridge, Mass., MIT Press.

——, and GEORGE A. MILLER. 1963. Introduction to the formal analysis of natural languages. Handbook of mathematical psychology, ed. by R. D. Luce, R. Bush, and E. Galanter, 2.269–321. New York, Wiley.

GLEASON, H. A., JR. 1961. An introduction to descriptive linguistics. New York, Holt.

HALLE, MORRIS. 1959. The sound pattern of Russian. The Hague, Mouton.

——. 1964a. On the bases of phonology. The structure of language, ed. by J. Katz and J. Fodor, 324–33. Englewood Cliffs, N. J., Prentice-Hall.

——. 1964b. Phonology in generative grammar. The structure of language, ed. by J. Katz and J. Fodor, 334–52. Englewood Cliffs, N. J., Prentice-Hall.

HOCKETT, CHARLES F. 1958. A course in modern linguistics. New York, Macmillan.

JAKOBSON, ROMAN, GUNNAR FANT, and MORRIS HALLE. 1951. Preliminaries to speech analysis. Cambridge, Mass., MIT Press.

——, and MORRIS HALLE. 1956. Fundamentals of language. The Hague, Mouton.

LEHMANN, WINFRED P. 1952. Proto-Indo-European phonology. Austin, University of Texas Press and L.S.A.

LIGHTNER, T. M. 1963. A note on the formation of phonological rules. Quarterly progress report, no. 68, 187–89. Research Laboratory of Electronics, MIT.

MCCAWLEY, JAMES. 1964. The accentual system of standard Japanese. MIT Ph.D. thesis.

MILLER, GEORGE, and NOAM CHOMSKY. 1963. Finitary models of language users. Handbook of mathematical psychology, ed. by R. D. Luce, R. Bush, and E. Galanter, 2.419–91. New York, Wiley.