Mathematical Linguistics

András Kornai

Foreword

Mathematical linguistics is rooted both in Euclid's ($\sim$325BCE – $\sim$265BCE) axiomatic method and in Pāṇini's ($\sim$520BCE – $\sim$460BCE) method of grammatical description. To be sure, both Euclid and Pāṇini built upon a considerable body of knowledge amassed by their precursors, but the systematicity, thoroughness, and sheer scope of the *Elements* and the *Ashṭādhyāyī* would place them among the greatest landmarks of all intellectual history even if we disregarded the key methodological advance they made.

As we will explain in greater detail in Chapter 2.2, the two methods are fundamentally very similar: the axiomatic method starts with a set of statements assumed to be true, and transfers truth from the axioms to other statements by means of a fixed set of logical rules, while the method of grammar is to start with a set of expressions assumed to be grammatical both in form and meaning and to transfer grammaticality to other expressions by means of a fixed set of grammatical rules.

Perhaps because our subject matter has attracted the efforts of some of the most powerful thinkers (of whom we single out A.A. Markov here, for reasons discussed in 1.3) from antiquity to the present day, there is no single easily accessible introductory text to mathematical linguistics. Indeed, to the mathematician the whole field of linguistics may appear to be hopelessly mired in controversy (see 1.1), and neither the formidable body of empirical knowledge about languages nor the standards of linguistic argumentation offer an easy entry point.

Those with a more postmodern bent may even go as far as to doubt the existence of a solid core of mathematical knowledge, often pointing at the false theorems and incomplete or downright wrong proofs that slip through the peer review process at a perhaps alarming rate. Rather than attempting to drown such doubts in rivers of philosophical ink, the present volume will simply proceed *ad more geometrico* in exhibiting this solid core of knowledge. In Chapters 3-7, a mathematical overview of the traditional main branches of linguistics, phonology, morphology, syntax, and semantics, are presented in a manner accessible to any reader with sufficient general mathematical maturity (graduate or advanced undergraduate).

Since no prior knowledge of linguistics is assumed on the part of the reader, those who enter the field through this book should be warned in advance that many branches of linguistics, in particular psycholinguistics and child language acquisition, are largely ignored here – not because they are viewed as inferior to other branches but simply because they do not offer enough grist for the mathematician's mill. Much of what the linguistically naive reader may find interesting about language turns out to be more pertinent to cognitive science, the philosophy of language, and sociolinguistics, than to linguistics proper, and the Introduction gives these issues the shortest possible shrift, discussing them only to the extent necessary for disentangling mathematical linguistics from other concerns.

Conversely, issues that linguists sometimes view as peripheral to their enterprise will get more discussion here simply because they offer such a rich variety of mathematical techniques and problems that no book on mathematical linguistics ignoring them could be considered complete. Thus we will devote Chapters 8-12 to phonetics, speech recognition, the recognition of handwriting and machine print, and in general to issues of linguistic signal processing and pattern matching. Though many of the same techniques are in wide use in information extraction, information retrieval, and statistical natural language processing, where the underlying parameter space is discrete, we can refer the reader to the excellent textbook by Manning and Schütze (1999) and concentrate more on the case of continuous parameters here.

From the early days of computers there was a great deal of overlap between the concerns of mathematical linguistics and computer science, and a surprising amount of work that began in one field ended up in the other, sometimes explicitly as part of computational linguistics, but often as general theory with its roots in linguistics largely forgotten. In particular, the basic techniques of syntactic analysis are now firmly embedded in the computer science curriculum, and the student can already choose from a large variety of textbooks that cover parsing, automata, and formal language theory. Here we will selectively cover only those aspects of the field that address specifically linguistic concerns, and again our guiding principle will be mathematical content, as opposed to algorithmic detail.

The book contains many exercises. These are, for the most part, rather hard (over level 30 in the system of Knuth 1971) but extremely rewarding. Especially in the later chapters, the exercises are often based on classical and still widely cited theorems, so the solutions can usually be found on the net quite easily. However, readers are strongly advised not to follow this route, unless they already spent several days attacking the problem. Unsolved problems presented as exercises are marked by an asterisk.

Contents

Chapter 1

Introduction

1.1 General concerns

1.1.1 Cumulative knowledge

It is hard to find any aspect of linguistics that is entirely uncontroversial, and to the mathematician less steeped in the broad tradition of the humanities it may appear that linguistic controversies are often settled on purely rhetorical grounds. Thus it may seem advisable, and only fair, to give both sides the full opportunity to express their views and let the reader be the judge. But such a book would run to thousands of pages, and would be of far more interest to historians of science than to those actually intending to learn mathematical linguistics. Therefore we will not accord equal space to both sides of such controversies, indeed often we will present a single view and will proceed without even attempting to discuss alternative ways of looking at the matter. Since part of our goal is to orient the reader not familiar with linguistics, typically we will present the majority view in detail and describe the minority view only tersely. For example, Chapter 4 introduces the reader to morphology, and will heavily rely on the notion of the morpheme – the excellent book by Anderson (1992) denying the utility, if not the very existence, of morphemes, will be relegated to footnotes. In some cases, when we feel that the minority view is the correct one, the emphasis will be inverted: for example Chapter 6, dealing with semantics, is more informed by the 'surface compositional' than the 'logical form' view.

In general, our goal is to present linguistics as a cumulative body of knowledge. In order to find a consistent set of definitions that offer a rational reconstruction of the main ideas and techniques developed over the course of millenia it will be often necessary to take sides in various controversies. But there is no pretense here that mathematical formulation will necessarily endow a particular set of ideas with greater verity, and often the opposing view could be formalized just as well.

One word of caution is in order: the fact that some idea is hard to formalize, or even seems so contradictory that a coherent mathematical formulation appears impossible, can be a reflection on the state of the art just as well as on the idea itself. Starting with Berkeley (1734) the intuitive notion of infinitesimals was subjected to all kinds of criticism, and it took over two centuries for mathematics to catch up and provide an adequate foundation in Robinson (1966). It is quite conceivable that equally intuitive notions, such as a *semantic theory of information*, which currently elude our mathematical grasp, will be put on firm foundations by later generations. In such cases, we content ourselves with explaining the idea informally, describing the main intuitions and pointing at possible avenues of formalization only programmatically.

1.1.2 Definitions

For the mathematician definitions are nearly synonymous with abbreviations: we say 'triangle' instead of the peculiar arrangement of points and lines that define it, 'polynomial' instead of going into a long song and dance about terms, addition, monomials, multiplication, or the underlying ring of coefficients, and so forth. The only sanity check required is to exhibit an instance, typically an explicit set-theoretic construction, to demonstrate that the defined object indeed exists.

In linguistics, there is rarely any serious doubt about the existence of the objects of inquiry. When we strive to define what is a 'word' we give a mathematical formulation not so much as to demonstrate that words exist, for we know it perfectly well that we use words both in spoken and written language, but rather to handle the odd and unexpected cases. The reader is invited to construct a definition now, and to write it down for comparison with the eventual definition that will emerge only after a rather complex discussion in Chapter 4.

In this respect mathematical linguistics is very much like the empirical sciences, where formulating a definition involves at least three distinct steps: an *ostensive* definition based on positive and sometimes negative examples (vitriol is an acid, lye is not), followed by an *extensive* definition delineating the intended scope of the notion (every chemical that forms a salt with a base is an acid), and the *intensive* definition that exposes the underlying mechanism (in this case, covalent bonds) emerging rather late, as a result of a long process of abstraction and analysis.

Throughout the book, the first significant instance of key notions will appear in *italics,* usually followed by ostensive examples (and counterexamples) in the next few paragraphs. (Italics will also be used for emphasis and for typesetting linguistic examples.) The observables associated with these notions are always discussed, but textbook definitions of an extensive sort are rarely given. Rather, a mathematical notion that serves as a stand-in will be defined in a rigorous fashion: in the defining phrase the same notion is given in **boldface**. Where an adequate mathematical formulation is lacking and we proceed by sheer analogy the key terms will be *slanted* – such cases are best thought of as open problems in mathematical linguistics.

1.1.3 Foundations

For the purposes of mathematical linguistics the classical foundations of mathematics are quite satisfactory: all objects of interest are sets, typically finite or, rarely, denumerably infinite. This is not to say that nonclassical metamathematical tools such as Heyting algebras find no use in mathematical linguistics (see Chapter 6....) but simply to assert that the fundamental issues of this field are not foundational, but definitional.

Given the finitistic nature of the subject matter, we will in general use the terms set, class, and collection interchangeably, drawing explicit cardinality distinctions only in the rare cases where we step out of the finite domain. Much of the classical linguistic literature of course predates Cantor, but even the modern literature typically conceives of infinity in the Gaussian manner of a potential as opposed to actual, Cantorian infinity. Because of immediate empirical concerns, denumerable generalizations of finite objects such as ω-words and Büchi automata are rarely used,[1] and in fact even the trivial step of generalizing from a fixed constant to arbitrary n is often viewed with great suspicion.

Aside from the tradition of Indian logic, the study of languages had very little impact on the foundations of mathematics. Rather, mathematicians have realized early on that natural language is a complex and in many ways unreliable construct, and created their own simplified language of formulas and the mathematical techniques to investigate it. As we shall see, some of these techniques are general enough to cover essential facets of natural languages, while others scale much more poorly.

There is an interesting residue of foundational work in the Berry, Richard, Liar, and other paradoxes which are often viewed as diagnostic of the vagueness, ambiguity, or even 'paradoxical nature' of natural language. Given the objective of developing a mathematical theory of language, once we have a firm notion of English as a formal system, the buck stops there, and questions like "what is the smallest integer not nameable in ten words" need to be addressed anew.

We shall begin with the seemingly simpler issue of the first number not nameable in *one* word. Since it appears to be one hundred and one, a number already requiring *four* words to name, we should systematically investigate the number of words in number names. There are two main issues to consider: what is a word (see Chapter 4) and what is a name (see Chapter 6). Another formulation of the Berry paradox invokes the notion of syllables: these are also discussed in Chapter 4. Eventually we will deal with the paradoxes in Chapter 6, but our treatment, along the standard Tarskian lines, offers little novelty to those interested in foundational issues.

1.2 The subject matter

What is *mathematical linguistics*? A classic book on the subject is (Jakobson, 1961), which contains papers on a variety of subjects including a catego-

[1]For a contrary view see Langendoen and Postal (1984)

rial grammar (Lambek 1961), formal syntax (Chomsky 1961, Hiż 1961), logical semantics (Quine 1961, Curry 1961), phonetics and phonology (Peterson and Harary 1961, Halle 1961), Markov models (Mandelbrot 1961), handwriting (Chao 1961, Eden 1961), parsing (Oettinger 1961, Yngve 1961), glottochronology (Gleason 1961), philosophy of language (Putnam 1961) as well as a number of papers that are harder to fit into our current system of scientific subfields, perhaps because there is a void now where once there was cybernetics and systems theory.

A good way to understand how these seemingly so disparate fields cohere is to proceed by analogy to mathematical physics. Hamiltonians receive a great deal more mathematical attention than, say, the study of generalized incomplete Gamma functions, because of their relevance to mechanics, not because the subject is, from a purely mathematical perspective, necessarily more interesting. Many parts of mathematical physics find a natural home in the study of differential equations, but other parts fit much better in algebra, statistics, and elsewhere. As we shall see, the situation in mathematical linguistics is quite similar: many parts of the subject would fit nicely in algebra and logic, but there are many others for which methods belonging to other fields of mathematics are more appropriate. Ultimately the coherence of the field, such as it is, depends on the coherence of linguistics.

Since almost all social activity ultimately rests on linguistic communication, there is a great deal of temptation to reduce problems from other fields of inquiry to purely linguistic problems. Instead of understanding schizophrenia, perhaps we should first ponder what the phrase *multiple personality* means. Mathematics already provides a reasonable notion of ‘multiple’, but what is ‘personality’, and how can there be more than one per person? Can a proper understanding of the suffixes *-al* and *-ity* be the key? This line of inquiry, predating the Schoolmen and going back at least to the *cheng ming* (rectification of names) doctrine of Confucius, has a clear and convincing rationale (The Analects 13.3, D.C. Lau transl.):

> When names are not correct, what is said will not sound reasonable; when what is said does not sound reasonable, affairs will not culminate in success; when affairs do not culminate in success, rites and music will not flourish; when rites and music do not flourish, punishments will not fit the crimes; when punishments do not fit the crimes, the common people will not know where to put hand and foot. Thus when the gentleman names something, the name is sure to be usable in speech, and when he says something this is sure to be practicable. The thing about the gentleman is that he is anything but casual where speech is concerned.

1.3 Mesoscopy

Physicists speak of mesoscopic systems when these contain, say, fifty atoms: too large to be given a microscopic quantum-mechanical description but too small for the classical macroscopic properties to dominate the behavior of the system. Linguistic systems are mesoscopic in the same broad sense: they have thousands of rules and axioms, compared to the handful of axioms used in most branches of mathematics. Group theory goes a long way exploring the implications of five axioms, arithmetic and set theory gets along with five and twelve axioms respectively (not counting members of axioms schemes separately), and the most complex axiom system in common use, that of geometry, has less than thirty axioms.

It comes as no surprise that with their mesoscopic number of axioms, linguistic systems are never pursued microscopically to yield implications in the same depth as group theory or even less well developed branches of mathematics. What is perhaps more surprising is that we can get reasonable approximations of the behavior at the macroscopic level, using the statistical techniques pioneered by A.A. Markov (see Chapters 7-8).

Statistical mechanics owes its success largely to the fact that in thermodynamics only a handful of phenomenological parameters are of interest, and these are relatively easy to link to averages of mechanical quantities. In mathematical linguistics the averages that matter, e.g. the percentage of words correctly recognized or correctly translated, are linked only very indirectly to the measurable parameters, of which there is such a bewildering variety that it requires special techniques to decide which ones to employ and which ones to leave unmodelled.

Macroscopic techniques, by their very nature, can yield only approximations for mesoscopic systems. Microscopic techniques, though in principle easy to extend to the mesoscopic domain, are in practice also prone to all kinds of bugs, ranging from plain errors of fact (which are hard to avoid once we deal with thousands of axioms) to more subtle, and often systematic, errors and omissions. Readers may at this point feel very uncomfortable with the idea that a given system is only 70%, 95%, or even 99.99% correct: after all, isn't a single contradiction or empirically false prediction enough to render a theory invalid? Since we need a whole book to develop the tools needed to address this question, the full answer will have to wait until Chapter 12. But for the impatient we provide a short answer here: yes and no.

Chapter 2

The elements

A primary concern of mathematical linguistics is to effectively enumerate those sets of words, sentences, etc. that play some important linguistic role. Typically, this is done by means of *generating* the set in question, a definitional method that we introduce in 2.1 by means of examples and counterexamples that show the similarities and the differences between the standard mathematical use of the term 'generate' and the way it is employed in linguistics.

Because the techniques used in defining sets, functions, relations, etc. are not always directly useful for evaluating them at a given point, an equally important concern is to solve the membership problem for the sets, functions etc. of interest. In 2.2 we therefore introduce a variety of *grammars* that can be used, among other things, to create *certificates* that a particular element is indeed a member of the set, gets mapped to a particular value, etc.

Since *generative grammar* is most familiar to mathematicians and computer scientists as a set of rather loosely collected string rewriting techniques, in 2.3 we give a brief overview of this domain. We put the emphasis on context-sensitive grammars, both because they play an important role in phonology (see Chapter 3) and morphology (see Chapter 4), and because they provide an essential line of defense against undecidability in syntax (see Chapter 5).

2.1 Generation

To define a collection of objects it is often expedient to begin with a fixed set of primitive elements E, and a fixed collection of *rules* R that describe permissible arrangements of the primitive elements as well as of more complex objects. If x, y, z are objects *satisfying* a (binary) rule $z = r(x, y)$ we say that z **directly generates** x and y (in this order) and use the notation $z \to_r xy$. The smallest collection of objects closed under direct generation by any $r \in R$ and containing all elements of E is called the set **generated** from E by R.

Very often the simplest or most natural definition yields a superset of the real objects of interest, which is therefore supplemented by some additional conditions to narrow it down. In textbooks of algebra the symmetric group is invariably introduced before the alternating group, and the latter is presented simply as a subgroup of the former. In logic, closed formulas are typically introduced as a special class of well-formed formulas. In context-free grammars the sentential forms produced by the grammar are kept only if they contain no nonterminals (see 2.3), and we will see many similar examples.

Generative definitions, like any other definition, need to be supported by some notion of *equality* among the defined objects. Typically, the notion of equality we wish to employ will abstract away from the derivational history of the object, but in some cases we will need a stronger definition of identity that treats two objects the same only if they were generated the same way.

In mathematical linguistics the objects of interest are the collection of words in a language, the collection of sentences, the collection of meanings etc. Even the most tame and obviously finite collections of this kind present great definitional difficulties. Consider, for example, the set of characters (graphemes) used in written English. Are uppercase and lowercase forms to be kept distinct? How about punctuation? Digits? Zapf dingbats? If there will be a new character for the euro currency unit, as there is a special character for dollar and pound sterling, shall it be included or shall we wait until an English-speaking country joins the euro-zone? Before proceeding to words, meanings, and other more subtle objects of inquiry, we will therefore first refine the notion of a generative definition on some familiar mathematical objects.

Example 2.1.1 Wang tilings. Let C be a finite set of colors, S be a finite set of square tiles, each colored on the edges according to some function $e : S \rightarrow C^4$. We assume that for each coloring *type* we have an infinite supply of *tokens* colored with that pattern: these make up the set of primitive elements E. The four rules n, s, e, w that make up R express the fact that we are interested only in tilings where edge-adjacent tiles have the same color on their adjacent edges. Let $\mathbb{Z}$ be the set of integers, $'$ be the successor function and ‘ be its inverse. For any $i, j \in \mathbb{Z}$ we say that the tile u whose bottom left corner is at (i, j) has a correct neighbor to the north if the third component of $e(u)$ is the same as the first component of $e(v)$ where v is the tile at (i, j'). Denoting the ith projection by π_i, we can write $\pi_3(e(u)) = \pi_1(e(v))$ for v at (i, j'). Similarly, the west rule requires $\pi_4(e(u)) = \pi_2(e(v))$ for v at $(i‘, j)$, the east rule requires $\pi_2(e(u)) = \pi_4(e(v))$ for v at (i', j), and the south rule requires $\pi_1(e(u)) = \pi_3(e(v))$ for v at $(i, j‘)$. We define **first quadrant (plane) tilings** as functions from $\mathbb{N} \times \mathbb{N}$ ($\mathbb{Z} \times \mathbb{Z}$) to E that satisfy all four rules.

Discussion While the above may look very much like a generative definition, there are some crucial differences. First, the definition relies on a number of externally given objects, such as the natural numbers, the integers, the successor function, and cartesian products. In contrast, the definitions we will encounter later, though they may require some minimal set-theoretical scaffolding, are almost always *noncounting* in the broad sense of being free of arithmetic aspects.

Second, the rules are *well-formedness conditions* (WFCs, see 2.3) rather than *rules of production.* In many cases, this is a distinction without a difference, since production rules can often be used to enforce WFCs: for example, if tilings are defined by the successive placement of new tiles starting at the origin and proceeding so that only placements preserving well-formedness are allowed, obviously the only first quadrant tilings we encounter will be the well-formed ones. However, we will often see reason to consider broader production processes, where ill-formed intermediate structures are integral to the whole process, and on the whole production rules are not limited to *conspiracies* enforcing well-formedness.

Third, the rules as stated have no recursive aspect whatsoever. There is no notion of larger structures built via intermediate structures: we go from the atomic units (tiles) to the global structure (tiling of the first quadrant) in one leap. Linguistic objects, as we shall see repeatedly, have many intermediate layers which are of interest in and of themselves. In fact, Example 2.1.1 will show this lack of recursive structure not only in the presentation chosen above, but in any other presentation.

Theorem 2.1.1 (Berger, 1966) It is recursively undecidable whether a given inventory of tiles E can yield a Wang tiling.

Proof We encode the halting problem in tiles...

Example 2.1.2 Presentation of groups in terms of generators and relations. Let E be a set of generators $g_1, \cdots, g_k$ and their inverses $g_1^{-1}, \cdots, g_k^{-1}$, and R be the usual rules of cancellation $g_i^{-1}g_i = g_ig_i^{-1} = e$. Formal products composed from the g_i and g_i^{-1} define the **free group** (or, if we omit inverses and cancellation, the **free monoid**) over k generators, with the usual conventions that the empty word is the multiplicative unit of the group (monoid) and that formal products containing canceling terms are equivalent to those with the canceling terms omitted. If a broader set of formal products is defined as canceling, representatives of this set are called **defining relations** for the group being presented, which is the factor of the free group by the cancellation kernel.

Discussion As is well known, it is in general undecidable whether a formal product of generators and inverses is included in the kernel or not (ref...Sims) We will discuss the relationship of combinatorial group theory and formal languages in Chapter 7, but gave the example here to show that the *equality* clause of generative definitions can lead to just the same complications as the *rules* clause.

Example 2.1.3 Herbrand universes. The primitive elements are the object constants of the first order language (FOL) under study (or an arbitrary constant if no object constant was available initially) and there are as many rules to describe permissible arrangements of elements as there are function constants in the FOL under study: if f was such a constant of arity n, $f(x_1, \ldots, x_n)$ is in the Herbrand universe provided the x_i were.

Discussion It should come as no surprise that logic offers many *par excellence* examples of generative definition – after all, the techniques developed for formalizing mathematical statements grew out of the larger effort to render statements of all sorts formally. However, the definition of a FOL abstracts away from several important properties of natural language. In FOLs functions and relations of arbitrary arity are permitted, while in natural language the largest number of arguments one needs to consider is five. Also, in many important cases (see Chapter 3) the freedom to utilize an infinite set of constants or variables is not required.

Exercise 2.1.1 The Fibonacci numbers are defined by $f_0 = 0, f_1 = 1, f_{n+1} = f_n + f_{n-1}$. Is this a generative definition? Why?

2.2 Axioms and rules

There is an unbroken tradition of no holds barred argumentation running from the Greek sophists to the Oxford Union, and the axiomatic method has its historic roots in the efforts to regulate (some would say emasculate) the permissible methods of debate. Since it is the general experience that almost all statements are arguable, one first needs to postulate a small set of primitive statements that the parties agree on – those who will not agree are simply excluded from the debate. As there is remarkable agreement about the validity of certain kinds of inference, the stage is set for a fully formal, even automatic, method of verifying whether a given argument indeed leads to the desired conclusion from the agreed upon premises.

There is an equally venerable tradition of protecting the full meaning and exact form of sacred texts, both to make sure that mispronunciations and other errors that may creep in over the centuries do not render them ineffectual and that misinterpretations do not confuse those whose task is to utter them on the right occasion. Even if we ignore the phonetic issues related to "proper" pronunciation (see Chapter 8), writing down the texts is far from sufficient for the broader goals of preservation. With any material of great antiquity, we rarely have a single, fully preserved and widely accepted version – rather, we have several imperfect variants and fragments. What is needed is not just a frozen description of some texts, say the Vedas, but also a grammar that defines what constitutes a proper Vedic text. The philological ability to determine the age of a section and undo subsequent modifications is especially important because the words of earlier sages are typically accorded greater weight.

In defining the language of a text, a period, or a speech community, we propagate *grammaticality* the same way we propagate truth in an axiomatic system. Once we choose an initial set of grammatical expressions, such as the set of words W, one important method is to assign each word an element (or a disjunction of elements) from another structure G with well-understood rules of combination by some well-defined mapping $c : W \rightarrow 2^G$, and consider grammatical only those sequences of words for which the rules of combination yield a desirable result. Demonstrating that the assigned elements of G indeed combine in the desired manner constitutes a certificate of membership.

Example 2.2.1 Categorial grammar. If G is a free group as in Example 1.2 above, and we consider only those word sequences $w_1.w_2.\ldots.w_n$ for which there is at least one h_i in each $c(w_i)$ such that $h_1 \cdot \ldots \cdot h_n = g_0$ (i.e. the group-theoretical product of the h_i yields a distinguished generator g_0), we obtain the classical notion of *bidirectional categorial grammar* (Bar-Hillel 1953, Lambek 1958). If we take G as the free Abelian group, we obtain *unidirectional categorial grammar* (Ajdukiewitz 1935).

Example 2.2.2 Unification grammar. By choosing G to be the set of directed acyclic labelnode graphs, where the labels are first order variables and constants, and considering only those word sequences for which the assigned graphs will unify we obtain a class of *unification grammars* (ref).

Example 2.2.3 Link grammar. By choosing G to satisfy a generalized version of the (horizontal) tiling rules of Example 2.1.1, we obtain the *link grammars* of Sleator and Temperley (1993).

We will investigate a variety of such systems in detail in Chapter 5 – here we concentrate on the major differences between truth and grammaticality. First, note that systems such as the above are naturally set up to define not only one distinguished set of strings, but its cosets as well. For example, in a categorial grammar we may inquire not only about those strings of words for which group multiplication of the associated categories yields the distinguished generator, but also about those for which the yield contains another generator or any specific word of G. This corresponds to the fact that e.g. *the house of the seven gables* is grammatical, but only as a noun phrase and not as a sentence while *the house had seven gables* is a grammatical sentence but not a grammatical noun phrase. There is no analogy with n-valued logics either, since the various stringsets defined by a grammar may overlap, and will in fact irreducibly overlap in every case that a primitive element is assigned more than one disjunct by c.

Second, the various calculi for propagating truth values by specific rules of inference can be supported by an appropriately constructed theory of model structures. In logic, a model will be unique only in degenerate cases: as soon as there is an infinite model, by the Löwenhein-Skolem theorems we have at least as many non-isomorphic models as there are cardinalities. In grammar, the opposite holds: as soon as we fix the period, dialect, style, and possibly other parameters determining grammaticality, the model is essentially unique. We may be tempted to set up the system so that e.g. British English and American English or Old English and Modern English are models of a single 'abstract English'. But from a linguistic standpoint this makes little sense inasmuch as grammars are intended as abstract models of the native speaker's competence, and the lack of cross-dialectal or historical data are never an impediment in the process of children acquiring their native language. Since children can perfectly well construct their internal grammar without access to such data, it would raise serious methodological problems for the grammarian to rely on facts outside the normal range of input available to children.

The fact that in the interesting sense there is only one model structure M gives rise to two notions peculiar to mathematical linguistics: *overgeneration* and *undergeneration*. If there is some string $w_1.w_2.\ldots.w_n \notin M$ that appears in the yield of G we say that G **overgenarates** (with respect to M), and if there is a $w_1.w_2.\ldots.w_n \in M$ that does not appear in the yield of G we say that G **undergenerates**. It is quite possible, indeed typical, for working grammars to have both kinds of error at the same time. We will develop quantitative methods to compare the error of different grammars in Chapter 7, but note here that neither undergeneration nor overgeneration is a definitive diagnostic of some fatal problem with the system. In many cases, overgeneration is benign in the sense that the usefulness of a system that e.g. translates English sentences to French is not at all impaired by the fact that it is also capable of translating an input that lies outside the confines of fully grammatical English. In other cases, the aim of the system may be to shed light only on a particular range of phenomena, say on the system of intransitive verbs, to the exclusion of transitive, ditransitive, etc. verbs: in the tradition of Montague grammar (see Chapter 6) such systems are explicitely called *fragments*.

In spite of these major differences, the practice of logic and grammar have a great deal in common. First, both require a systematic ability to analyze sentences in component parts so that generalizations involving only some part can be stated, and the ability to construct new sentences from ones already seen. Chapter 5 will discuss such *syntactic* abilities in detail, but we note here that the practice of logic is largely *normative* in the sense that constructions outside those explicitly permitted by its syntax are declared ill-formed, while the practice of linguistics is largely *descriptive* in the sense that it takes the range of existing constructions as given, and strives to adjust the grammar so as to match this range.

Second, both logic and grammar are largely driven by an overall consideration of economy. As the reader will have no doubt noticed, having a separate WFC for the northern, southern, eastern, and western edges of a tile in Example 2.2.1 is quite unnecessary: any two orthogonal directions would suffice to narrow down the range of well-formed tilings. Similarly, in context-free grammars we often find it sufficient to deal only with rules that yield only two elements on the right-hand side (Chomsky normal form), and there has to be some strong reason for departing from the simplest binary branching structure (see Chapter 5).

Finally, the all-important boundary between recursive and recursively enumerable (r.e.) is drawn the same way by certificates (derivation structures), even though the systems of interest congregate on different sides of this boundary. In logic, proving the negation of a statement requires the same kind of certificate (a proof object rooted in the axioms and terminating in the desired conclusion) as proving the statement itself – the difficulty is that most calculi are r.e. but not recursive (decidable). In grammar, proving the ungrammaticality of a form requires an apparatus very different from proving its grammaticality: for the latter purpose an ordinary derivation suffices but for the former we typically need to exhaustively survey all forms of similar and lesser complexity, which can be difficult, even though most grammars are not only r.e. but in fact recursive.

2.3 String rewriting

Given a set of atomic symbols Σ called the **alphabet,** the simplest imaginable operation is that of **concatenation** whereby a complex symbol xy is formed from x and y by writing them in succession. Applying this operation recursively, we obtain **strings** of arbitrary *length.* Whenever such a distinction is necessary, the operation will be denoted by . (dot), but the result of the operation is viewed as having no internal punctuation: $u.v = uv$ both for atomic symbols and for more complex strings, corresponding to the fact that concatenation is associative. Of special interest is the **empty string** λ which serves as a two-sided multiplicative unit of concatenation: $\lambda.u = u.\lambda = u$. The whole set of strings generated from Σ by concatenation is denoted by Σ^+ (**λ-free Kleene closure**) or, if the empty string is included, by Σ^* (**Kleene closure**). If we define the **length** of the empty word as 0, that of atomic symbols as 1, and require length to be a homomorphism from Σ^* to the additive semigroup of nonnegative integers, we end up with the standard definition of length counting multiple occurrences of the same symbol as separate. In particular, the semigroup of nonnegative integers (with ordinary addition) is isomorphic to the Kleene closure of a one-symbol alphabet (with concatenation): the latter is called integers in **base one** notation.

Subsets of Σ^* are called **stringsets** or **languages**. In addition to the standard Boolean operations, we can define the **concatenation** of strings and languages U and V as $UV = \{uv|u \in U, v \in V\}$, suppressing the distinction between a string and a one-member language, writing xU instead of $\{x\}U$ etc. The (λ-free) Kleene closure of strings and languages is defined analogously to the closure of alphabets. Of particular interest is the finitely generated case: we call **regular** all finite languages, and all languages that can be obtained from these by repeated application of the above operations: these have the same distinguished status among languages that the rationals in $\mathbb{Q}$ have among numbers in $\mathbb{R}$. The basic facts about regular languages, finite automata, and Kleene's theorem are covered in most textbooks about formal language theory or foundations of computer science – for an in-depth discussion see Eilenberg (1974). Some generalizations of particular interest to linguistics, regular relations, and finite k-automata, are discussed in 3.4.

Exercise 2.3.1 Let F be the language of Fibonacci numbers written base one. Is F finitely generated?

The generative grammars defining stringsets typically use an alphabet V that is a proper superset of Σ which contains the symbols of interest. Elements of $N = V \setminus \Sigma$ are called **nonterminal symbols** or just **nonterminals** to distinguish them from elements of Σ (called **terminal symbols** or **terminals**). Nonterminals play only a transient role in generating the objects of real interst, inasmuch as the yield of a grammar is explicitely restricted to terminal strings – the name nonterminal comes from the notion that a string containing these corresponds to a stage of the derivation that has not (yet) terminated. In **context-free grammars** (CFGs) we use a **start symbol** $S \in N$ and **pro-**

ductions or **rewrite rules** of the form $A \rightarrow v$ where $A \in N$ and $v \in V^*$.

Example 2.3.1. A CFG for base ten integers. We use nonterminals `SIGN` and `DIGIT` (treated as atomic symbols rather than strings of Latin letters) and posit the rules $S \rightarrow$ `SIGN DIGIT`; $S \rightarrow$ `DIGIT`; `DIGIT` $\rightarrow$ `DIGIT DIGIT`; `DIGIT` $\rightarrow$ 0; `DIGIT` $\rightarrow$ 1; ... `DIGIT` $\rightarrow$ 9; `SIGN` $\rightarrow$ `+`; `SIGN` $\rightarrow$ `-`. At the first step of the derivation we can only choose the first or the second rule (since no other rule rewrites S) and we obtain the string `SIGNDIGIT` or `DIGIT`. Taking the first option, and using the last rule to rewrite `SIGN` we obtain `-DIGIT`, and using the third rule n times we get `-DIGIT`$^{n+1}$. By eliminating the nonterminals we obtain a sequence of $n+1$ decimal digits preceded by the minus sign.

Discussion Needless to say, base ten integers are easy to define by simpler methods, and the CFG used above is overkill also in the sense that strings with three or more digits will have more than one derivation. But **context free languages** (languages generated by a CFG) are a proper superset of regular languages. For example, consider the CFG with nonterminal S, terminals a, b and rewrite rules $S \rightarrow aSa$; $S \rightarrow bSb$; $S \rightarrow a$; $S \rightarrow b$; $S \rightarrow \lambda$. It is easily seen that this grammar defines the language of *palindromes* over $\{a, b\}$, which contains exactly those strings that are their own reversal (mirror image). The proof that this language is not regular is deferred to Chapter 5, but we note here that in general CFLs play the same role among languages that algebraic numbers play among the reals.

If a string $i = x.y$ we say that x is a **prefix** and y is a **suffix** in i. If a context-free rewrite rule $A \rightarrow v$ is applied to a string iAj and l is a suffix of i (r is a prefix of j) we say that the rule applied **in the left context** i (**in the right context** j). **Context-sensitive** rewrite rules are defined as triples (p, l, r) where p is a context-free production as above, and l and r are (possibly empty) strings defining the left and the right context of the rule in question. Traditionally, p is called the **structural change**, the context, written l_r, is called the **structural description,** and the triple is written as the structural change separated from the structural description by / (assumed to be outside the alphabet P). For example, a rule that deletes a leading zero from an unsigned decimal could be written $0 \rightarrow \lambda/\#\_$, and the more general rule that deletes it irrespective of the presence of a sign could be written $0 \rightarrow \lambda/\{\#, +, -\}\_$. The right context of these rules is empty (it does not matter what digit, if any, follows the leading 0), but on the left context "edge of string" needs to be explicitely marked by the # symbol conventionally reserved for this purpose.

When interpreted as WFCs, the contect statements simply act as filters on derivations: an otherwise legitimate rewriting step $iAj \rightarrow ivj$ is blocked (deemed ill-formed) unless l is a suffix of i and r is a prefix of j. This notion of context-sensitivity adds nothing to the generative power of CFGs: the resulting system is still capable only of generating CFLs.

Theorem 2.3.1 (McCawley, 1968) Context-free grammars with context checking generate only context-free languages.

Proof

However, if context-sensitivity is part of the generation process, we can obtain context-senstive languages (CSLs) that are not CFLs. If λ-rules (rewriting nonterminals as the empty string in some context) are permitted, every r.e. language can be generated, but if such rules are disallowed, we obtain the CSL family proper (the case when CSLs contain the empty string has to be treated separately). As is well known, the family of languages that can be generated by λ-free context-sensitive productions is the same as the family of languages that can be generated by using only lenght-increasing productions (i.e. productions of the form $u \to v$ where u is not longer than v) and the same as the family of languages computable by **linear bounded automata** (LBA). LBA are Turing machines that accept on the empty tape, with the additional restriction that at all stages of the computation, the reading head must remain on the portion of the tape that was used to store the string whose membership is to be decided.

These results are traditionally summarized in the *Chomsky hierarchy*: assigning regular languages to Type 3, CFLs to Type 2, CSLs to Type 1, and r.e. languages to Type 0, (Chomsky, 1956) demonstrated that each type is properly contained in the next lower one. These proofs, together with examples of context-free but not regular, context-sensitive but not context free, and recursive but not context sensitive languages, are omitted here, as they are discussed in many excellent textbooks of formal language theory such as (Salomaa, 1973) or (Harrison, 1978). To get a better feel for CSLs, we prove the following results:

Theorem 2.3.2 (Karp, 1972) The membership problem for CSLs is PSPACE-complete.

Theorem 2.3.3 (Szelepcsényi, 1987) (Immerman, 1988) The complement of a CSL is a CSL.

Exercise 2.3.2 Construct three CSGs that generate the language F of Fibonacci numbers in base one, the language F_2 of Fibonacci numbers in base two, and the language F_{10} of Fibonacci numbers in base ten. Solve the membership problem for 117467.

Chapter 3

Phonology

The fundamental unit of linguistics is the *sign* which, as a first approximation, can be defined as a conventional pairing of sound and meaning. By *conventional* we mean both that signs are handed down from generation to generation with little modification, and that the pairings are almost entirely arbitrary, just as in bridge where there is no particular reason for a bid of two clubs in response to one no trump to be construed as an inquiry about the partner's major suits.

One of the earliest debates in linguistics, dramatized in Plato's *Cratylus* concerns the arbitrariness of signs. One school maintained that for every idea there is a true sound that expresses it best, something that makes a great deal of sense for *onomatopoeic* words (describing e.g. the calls of various animals) but is hard to generalize outside this limited domain. Ultimately the other school prevailed (see Lyons 1968 Ch. 1.2 for a discussion) at least as far as the word-level pairing of sound and meaning is concerned, though the issue still raises its head time and again in various debates concerning natural word order and the Sapir-Whorf Hypothesis (S-WH). We will return to these matters in Chapter 5 and 6, and here restrict ourselves to noting that the S-WH itself is clearly conventionally named, since neither Sapir's nor Whorf's published work shows much evidence for either of them having ever posed it.

In Section 3.1 we describe the concatenative building blocks of sound structure called 'phonemes'. Some subatomic components called 'distinctive features' and the formal linguistic mechanisms required to handle them are discussed in 3.2. Both subatomic and larger units are treated as 'autosegments' in 3.3. Two generalizations of regular languages motivated by phonological considerations, regular transducers and regular k-languages, are introduced in 3.4. The notions of prosodic hierarchy and optimality, being equally relevant for phonology and morphology, are deferred to Chapter 4.

3.1 Phonemes

It is desirable to build up the theory of sounds without reference to the theory of meanings both because the set of atomic units of sound promises to be considerably simpler than the set of atomic units of meanings, and because sounds as linguistic units appear to possess clear physical correlates (acoustic waveforms, see Chapter 8) while meanings, for the most part, appear to lack any direct physical embodiment. There is at least one standard system of communication, Morse code, which gets by with only two units, dot (short beep) and dash (long beep). To be sure, Morse code is parasitic on written language, which has a considerably larger alphabet, but the enormous success of the alphabetic mode of writing itself indicates clearly that it is possible to analyze speech sounds into a few dozen atomic units, while efforts to do the same with meaning (such as Wilkins 1668) could never claim similar success. However, we need not postulate the existence of some alphabetic system for transcribing sounds, let alone a meaning decomposition of some given kind: rather, we will start with easily observable entities called *utterances* and defined as maximal pause-free stretches of speech.

We owe the program of eliminating references to meaning from operational definitions of linguistic phenomena to the structuralist school, conventionally dated to begin with Saussure (1881). The first detailed axiom system is Bloomfield (1926): the version we present here owes a great deal to later developments, in particular Harris (1951). We are investigating the very complex *interpretation relation* that obtains between certain structured kinds of sounds and certain structured kinds of meanings: our eventual goal is to define it in a generative fashion. At the very least, we must have some notion of identity that tells us whether two signs sound the same and/or mean the same. The key idea is that we actually have access to more information, namely, whether two utterances are *partially similar* in form and/or meaning. To use Bloomfield's original examples:

> A needy stranger at the door says *I'm hungry.* A child who has eaten and merely wants to put off going to bed says *I'm hungry.* Linguistics considers only those vocal features which are alike in the two utterances (...) Similarly, *Put the book away* and *The book is interesting* are partly alike *(the book).*

That the same utterance can carry different meanings at different times is a fact we shall not explore until we introduce *disambiguation* in Chapter 6 – the only burden we now place on the theory of meanings is that it be capable of (i) distinguishing meaningful from meaningless and (ii) determining whether the meanings of two utterances share some aspect. Our expectations of the observational theory of sound are similarly modest: we assume we are capable of (i') distinguishing pauses from speech and (ii') determining whether the sounds of two utterances share some aspect.

We should emphasize at the outset that the theory developed on this basis does not rely on our ability to exercise these capabilities to the extreme. It is not necessary to be able to decide whether a 20 millisecond stretch that contains exactly 1.001 times the physiological minimum of audible sound energy constitutes a pause or not. If this stretch is indeed a pause we can always produce another instance, one that will have a 2000 millisecond pause and one tenth of the previous energy, which will show quite unambiguously that we had two utterances in the first place. If it was not a pause, but rather a functional part of sound formation such as a stop closure, the new 'utterances' with the artificially interposed pause will be deemed ill-formed by native speakers of the language. Similarly, we need not worry a great deal whether *Colorless green ideas sleep furiously* is meaningful, or what it exactly means: the techniques described here are robust enough to perform well on the basis of ordinary data.

The domain of the interpretation relation I is the set of *forms* F, and the codomain is the set of *meanings* M, so we have $I \subset F \times M$. In addition, we have two *overlap* relations $O_F \subset F \times F$ and $O_M \subset M \times M$ that determine partial similarity of form and meaning respectively. O_F is traditionally divided into *segmental* and *suprasegmental* overlap: we will discuss mostly segmental overlap here and defer suprasegmentals like tone and stress to 3.3 and 4.2 respectively. Since speech happens in time, we can define two forms α and β to *segmentally overlap* if their temporal supports as intervals on the real line can be made to overlap, as in the *the book* example above. In the segmental domain at least, we therefore have a better notion than mere overlap: we have a partial ordering defined by the usual notion of interval containment. In addition to O_F we will therefore use sub- and superset relations (denoted by $\subset_F, \supset_F$) as well as intersection, union and complementation operations in the expected fashion, and we have

$$\alpha \cap_F \beta \neq \emptyset \Rightarrow \alpha O_F \beta \tag{3.1}$$

In the domain of I we find obviously complex forms such as a full epic poem and some that are atomic in the sense that

$$\forall x \subset_F \alpha : x \notin dom(I) \tag{3.2}$$

– these are called *minimum forms.* A form that can stand alone as an utterance is a *free form*, the rest (e.g. forms like *ity* or *al* as in *electricity, electrical*), which can not normally appear between pauses, are called *bound forms.*

Typically, utterances are full phrases or sentences, but when circumstances are right, e.g. because a preceding question sets up the appropriate context, forms much smaller than sentences can stand alone as complete utterances. Bloomfield in fact defines a *word* as a minimum free form. For example, *electrical* is a word because it is a free form (can appear e.g. as answer to the question *What kind of engine is in this car?*) and it can not be decomposed further into free forms (*electric* would be free but *al* is bound). We will have reason to revise this definition in Chapter 4, but for now we can provisionally adopt it here, because in defining phonemes it is sufficient to restrict ourselves to free forms.

For the rest of this section we will only consider the set of words $W \subset F$, and we are in the happy position of being able to ignore the meanings of words entirely. We may know that forms like *city* and *velocity* have nothing in common as far as their meanings are concerned, and that we can not reasonably analyze the latter as containing the former, but we also know that the two rhyme and as far as their forms are concerned *velocity* = *velo* . *city.* Similarly, *velo* and *kilo* share the form *lo* so we can isolate *ve, ki,* and *lo* as more elementary forms.

In general, if pOq, we have a nonempty u such that $p = aub, q = cud$. a, b, c, d, u will be called **word fragments** obtained from comparing p and q, and we say p **is a subword of** q, denoted $p \prec q$, if $a = b = \lambda$. We denote by $\tilde{W}$ the smallest set containing W and closed under the operation of taking fragments : $\tilde{W}$ contains all and only those fragments that can be obtained from W in finitely many steps.

By successively comparing forms and fragments we can rapidly extract a set of short fragments P that is sufficiently large for each $w \in \tilde{W}$ to be a concatenation of elements of P and sufficiently small that no two elements of it overlap. A **phonemic alphabet** P is therefore defined by (i) $\tilde{W} \subset P^*$ and (ii) $\forall p, q \in P : pO_Fq \Rightarrow p = q$. We emphasize here that the procedure for finding P does not depend on the existence of an alphabetic writing system: all it requires is an informant (oracle) who can render judgements about partial similarity, and in practice this person can just as well be illiterate. Though the number of unmapped languages is shrinking, to this day the procedure is routinely carried out whenever a new language is encountered.

For an arbitrary set W endowed with an arbitrary overlap relation O_F there is no guarantee that a phonemic alphabet exists: for example, if W is the set of intervals $[0, 2^{-n}]$ with overlap defined in the standard manner, $P = \{[2^{-(i+1)}, 2^{-i}] : i \geq 0\}$ will enjoy (ii) but not (i). But in actual word inventories W and their extensions $\tilde{W}$ we never see the phenomenon of an infinite descending chain of words or fragments $w_1, w_2, \ldots$ such that each w_{i+1} is a proper part of w_i, nor can we find a large number (say $> 2^8$) words or fragments such that no two of them overlap. We call such statements of contingent facts about the real world *postulates*, to distinguish them from ordinary axioms, which are not generally viewed as subject to falsification.

Postulate 3.1.1 Foundation. Any sequence of words and word fragments $w_1, w_2, \ldots$ such that each $w_{i+1} \prec w_i, w_{i+1} \neq w_i$, terminates after a finite number of steps.

Postulate 3.1.2 Dependence. Any set of words or word fragments $w_1, w_2, \ldots$ w_m contains two different but overlapping words or fragments for any $m > 256$.

From these two postulates both the existence and uniqueness of phonetic alphabets follows. Foundation guarantees that every $w \in W$ contains at least one atom under $\prec$, and dependence guarantees that the set P of atoms is finite. Since different atoms can not overlap, all that remains to be seen is that every word of $\tilde{W}$ is indeed expressible as a concatenation of atoms. Suppose indirectly that q is a word or fragment that could not be expressed this way: either q itself is atomic or we can find a fragment q_1 in it which is not expressible. Repeating the same procedure for q_1, we obtain $q_2, \ldots, q_n$ – because of postulate 3.2.1 the procedure terminates in an atomic q_n. But by definition of P, q_n is a member of it, a contradiction which proves the indirect hypothesis false.

Discussion Nearly every communication system that we know of is built on a finite inventory of discrete symbols. There is no law of nature that would forbid a language to use measure predicates such as *tall* that take different vowel length in proportion to the tallness of the object described: in such a hypothetical language we could say *It was taaaaaaall* to express the fact that something was 7 times as tall as some standard of comparison, and *It was taaall* to express that it was only 3 times as tall. Yet even though analog signals are available, we find that in actual languages these are used only to convey a discrete set of possible values (see Chapter 8).

Postulates 3.1.1-2 go some way toward explaining why discretization of continuous signals must take place: we can speculate that foundation is necessitated by limitations of perception (it is hard to see how a chain could descend below every perceptual threshold), and dependence is caused by limitations of memory (it is hard to see how an infinite number of totally disjoint atomic units could be kept in mind). But no matter how valid these explanations turn out to be, the postulates have a clear value in helping us distinguishing linguistic systems from non-linguistic ones. For example, the dance of bees, where the direction and size of figure-8 movements is directly related to the direction and distance from the hive to where food can be collected (von Frisch, 1967), must be deemed non-linguistic, while the genetic code, where information about the composition of proteins is conveyed by DNA/RNA strings, can at least provisionally be accepted as linguistic.

Following the tradition of Chomsky (1965), memory limitations are often grouped together with mispronunciations, lapses, hesitations, coughing, and other minor errors as *performance* factors, while more abstract and structural properties are treated as *competence* factors. Though few doubt that some form of the competence vs. performance distinction is valuable, at least as a means of keeping the noise out of the data, there has been a great deal of debate about where the line between the two should be drawn (refs). Given the orthodox view that limitations of memory and perception are matters of performance, it is surprising that such a deeply structural property as the existence of phonetic alphabets can be derived from postulates rooted in these limitations.

3.2 Natural classes and distinctive features

Isolating the atomic segmental units is a significant step toward characterizing the phonological system of a language: using the phonemic alphabet P we can write every word as a string $w \in P^*$, and by adding just one extra symbol # to denote the pause between words, we can write all utterances as strings over $P \cup \{\#\}$. Since in actual *connected* speech pauses between words need not be manifest, we need an interpretative convention that # can be *phonetically realized* either as silence or as the empty string (zero realization). Silence, of course, is distinctly audible and has positive duration (usually 20 miliseconds or longer) while λ could not be heard and has 0 duration.

In fact, similar interpretative conventions are required throughout the alphabet, e.g. to take care of the fact that in English word-initial t is *aspirated* (released with a puff of air similar in effect to h but much shorter) while in many other positions t is *unaspirated* (released without an audible puff of air), compare *ton* to *stun.* The task of relating the abstract units of the alphabet to their audible mainifestations is a complex one, and we defer the details to Chapter 9, but we note here that the interpretation process is by no means trivial, and there are many unassailable cases, such as aspirated vs. unaspirated t, silenceful vs. empty #, where we permit two or more alternative realizations for the same segment. (Here and in what follows we reserve the term **segment** to alphabetic units i.e. strings of length one.)

Since λ can be one of the alternatives, an interesting technical possibility is to permit cases when it is the only choice, i.e. to declare elements of a phonemic alphabet that never get realized. The use of such *abstract* or *diacritic* elements *(anubandha)* is already pivotal in Pāṇini's system, and remains characteristic of phonology to this day. This is our first example of the linguistic distinction between *underlying* (abstract) versus *surface* (concrete) elements – we will see many others later.

Because in most cases alternative realizations of a symbol are governed by the symbols in its immediate neighborhood, the mathematical tool of choice for dealing with most of segmental phonology is string rewriting by means of context sensitive rules. This is not to say that the set of words W, viewed as a language over P, or over a larger alphabet Q that includes abstract elements as well, will be context sensitive in the sense of formal language theory: to the contrary, we have good reasons to believe that W is in fact regular. We defer this issue to 3.4, and for now emphasize only the convenience of context sensitive rules, which offer an easy and well-understood mechanism to express the phonological regularities or *sound laws* that have been discovered over the centuries.

Example 3.2.1 Final devoicing in Russian. The nominative form of Russian nouns can be predicted from their dative forms by removing the dative suffix u and inspecting the final consonant: if it was b or p, the final consonant of the nominative form will be p. This could be expressed in a phonological rule of *final b devoicing:* $b \rightarrow p/\_\#$ Most remarkably, we find that a similar rule links d to t, g to k and in fact any voiced obstruent to its voiceless counterpart.

The phenomenon that the structural description and/or the structural change in rules extends to some disjunction of segments is extremely pervasive. Those sets of segments that frequently appear together in rules are called *natural classes:* for example, the class $\{p, t, k\}$ of *unvoiced stops* or the class {b, d, g} of *voiced stops* are both natural, but the class $\{p, t, d\}$ is not: phonologists would be truly astonished to find a language where some rule or regularity affects *p, t,* and d but no other segment.

The linguist has no control over the phonemic alphabet of a language: P is computed as the result of a specific (oracle-based, but otherwise deterministic) algorithm. Since the set $N \subset 2^P$ of natural classes is also externally given by the phonological patterning of the language, over the millenia a great deal of effort has been devoted to the problem of properly characterizing it, both in order to shed some light on the structure of P and to help simplify the statement of rules.

So far, we treated P as an unordered set of alphabetic symbols. In the Ashṭādhyāyī, Pāṇini arranges elements of P in a linear sequence (the *śivasūtras*) with some abstract (phonetically unrealized) symbols *(anubandha)* interspersed. Simplifying his treatment somewhat (for a fuller discussion, see Staal 1962), natural classes *(pratyāhāra)* are defined in his 1.1.71 as those subintervals of the *śivasūtras* which end in some *anubandha*. If there are k symbols in P, in principle there could be as many as 2^k natural classes. However, the Pāṇinian method will generate at most $k(k+1)/2$ subintervals (or even fewer, if diacritics are used more sparingly) which is in accordance with

Postulate 3.2.1 In any language, the number of natural classes is small.

We do not exactly spell out what "small" means here: certainly it has to be polynomial, rather than exponential, in the size of P. The European tradition reserves names for many important natural classes such as the *apicals, aspirates, bilabials, consonants, continuants, dentals, fricatives, glides, labiodentals, linguals, liquids, nasals, obstruents, sibilants, stops, spirants, unaspirates, velars, vowels* etc. etc. – all told, there could be a few hundred, but certainly not a few thousand, such classes. As these names suggest, the reason why a certain class of sounds is natural can often be found in sharing some aspects of production (e.g. all sounds crucially involving a constriction at the lips are *labials*, and all sounds involving turbulent airflow are *fricatives*) but often the justification is far more complex and indirect. In some cases, the matter whether a particular class is natural is heavily debated: for a particularly hard chesnut, the *ruki* class, see Collinge's (1985) discussion of Petersen's Law I and the references cited therein.

For the mathematician the first question to ask about the set of natural classes N is neither its size nor its exact membership, but rather its algebraic structure: what operations is N closed under? To the extent Pāṇini is right the structure is not fully Boolean: the complement of an interval typically will not be expressible as a single interval. but the intersection of two intervals *(pratyāhāra)* will again be an interval. We state this as

Postulate 3.2.2 In any language, the set of natural classes is closed under intersection.

This postulate already makes N a meet semilattice, but we can go beyond this observation. Let us define, following Ehrenfeucht (pc), the notion of (semi)independence: two subsets X and Y of some domain D are **independent** (denoted $X \not\sim Y$) iff none of the sets $X \setminus Y, Y \setminus X, X \cap Y, \overline{X \cup Y}$ is empty. Two sets X and Y are **semi-independent** (denoted $X \not- Y$) iff none of the sets $X \setminus Y, Y \setminus X, X \cap Y$ is empty. Informally, independence means that no Bayesian inferences can be made: knowing that some $p \in D$ is or is not a member of

X gives us no information about its membership in Y. Semi-independence is a slightly weaker, but for our purposes more useful notion.

Definition 3.2.1 A set $E \subset 2^D$ is an **Independent Boolean Algebra** (IBA) iff (i) $X, Y \in E, X \not\sim Y \Rightarrow X \cup Y, X \cap Y, X \setminus Y, Y \setminus X \in E$; (ii) $X \in E \Rightarrow \overline{X} \in E$; and (iii) $\emptyset, D, \{a\}, \overline{\{a\}} \in E$ (singleton sets and their complements).

Definition 3.2.2 A set $S \subset 2^D$ is a **SemiIndependent Boolean Ring** (SIBR) iff (i') $X, Y \in S, X \not\vdash Y \Rightarrow X \cup Y, X \cap Y, X \setminus Y, Y \setminus X \in S$ and (ii') $\emptyset, D, \{a\} \in S$ (singleton sets).

Let us consider a few simple examples of IBAs and SIBRs. First of all, the systems of sets listed in clause (iii) and (ii') of the above definitions are obviously IBAs and SIBRs respectively – let us denote them by **A**. Second, traditional boolean algebras are of course IBAs and SIBRs – let us denote them by **B**. The third example (and the first nontrivial one) is the IBA built on GF(2,2) i.e. the 2-dimensional cube with points `a=(0,0), b=(0,1), c=(1,0), d=(1,1)` by including all subsets *except* {`b,c`} and {`a,d`}. As the reader can easily verify, this 14-member set of sets is an IBA but not a SIBR. The key idea is to view these sets from the center of the square, so to speak, as segments in a cyclically ordered set of n points. If all such segments are included, we get an IBA, and if we break up the circle and use linear ordering we get a SIBR. Let us denote the class of such interval structures by **C**. For the sake of completeness we gave both definitions here, but for the case of natural classes only the second one will be relevant: because the complement set of a segment is hardly ever natural, N fails to meet condition (iii) of Definition 3.2.1 above.

To see that intervals (pratyāhāra) form a SIBR, consider two intervals $[AB]$ and $[CD]$. If they are are semi-independent, their intersection is non-empty, so there is a segment x such that $x \leq B$ and $C \leq x$. Therefore, $C < B$ and by similar appeals to the nonemptiness of $[AB] \setminus [CD]$ and $[CD] \setminus [AB]$ it follows that $A \leq C \leq B \leq D$, and thus $[AC)$, $(BD]$, and $[AD]$ are also intervals. (The open intervals can be replaced by closed ones because there are only finitely many points involved.)

Another way of weakening the Boolean structure is to consider meet semilattices of linear subspaces. The modern treatment of natural classes begins with (Trubetskoi, 1939), and Jakobson (Jakobson et al., 1952), who assumed that the defining properties of natural classes are all orthogonal. Their method (see also Cherry et al. (1953), Cherry 1956) is to embed P in a hypercube so that natural classes correspond to hyperplanes parallel to the axes. The basis vectors that give rise to the hypercube are called **distinctive features** and are generally assumed to be binary: a typical example is the *voiced/unvoiced* distinction that is defined by the presence/absence of periodic vocal fold movements.

It is debatable whether the field underlying this vector space construct should be $\mathbb{R}$ of GF(2). The first position is explicitely taken by Cherry (1956), Stevens and Blumstein (1981), who assume the coordinates in feature space to be directly measurable properties of the sounds. The second, implicit in the more abstract outlook of Trubetzkoy and Jakobson, and explicit in the single most influential work on the subject, Chomsky and Halle (1968), is the route we follow here, but we will have reason to return to the notion of real-valued features in Chapter 8. Thus we define a **feature assignment** as an injective mapping C from the set Q of segments into the linear space GF(2,n). In this theory, segments are defined as *bundles* (vectors) of features, and **natural classes** are those sets of segments that can be expressed by fewer features than their individual members (see Halle 1964:328).

Exercise 3.2.1. Verify that with this definition postulates 3.2.1-2 are satisfied.

In addition to using *pratyāhāra,* Pāṇini employs a variety of other devices, most notably, the concept of 'homogeneity' *(sāvarṇya)* as a means of cross-classification (see Cardona 1965). This device enables him to to treat quality distinctions in vowels separately from length, nasality, and tone distinctions, as well as to treat place of articulation distinctions in consonants separately from nasality, voicing and aspiration contrasts. Another subsidiary concept, that of *antara* 'nearness' is required to handle the details of mappings between natural classes. Since Pāṇinian rules always map classes onto classes, the image of a segment under a rule is decided by 1.1.50 *sthāne 'ntaratamaḥ* 'in replacement, the nearest'. The modern equivalent of P1.1.50 is is the convention that features unchanged by a rule need not be explicitely mentioned, so that the Russian final devoicing rule that we began with may simply be stated as [+obstruent] $\rightarrow$ [-voice] / _#.

For very much the same empirical reasons that forced Pāṇini to introduce additional devices like *sāvarṇya*, the contemporary theory of features also relaxes the requirement of full orthogonality. One place where the standard (Chomsky and Halle, 1968) theory of distinctive features shows some signs of strain is the treatment of vowel height. Phonologists and phoneticians are in broad agreement that vowels come in three varieties: *high, mid* and *low*, which form an interval structure: we often have reason to group high and mid vowels together, or to group mid and low vowels together, but we never see a reason to group high and low vowels together to the exclusion of mid vowels. The solution adopted in the standard theory is to use two binary features, [$\pm$ high] and [$\pm$ low] and to declare the conjunction [+high, +low] ill-formed. Similar issues arise in many other corners of the system (e.g. in the treatment of *place of articulation* features).

While in principle we could replace the two underutilized GF(2) factors that define vowel height by a single GF(3)-valued feature, for reasons to be discussed presently this is not a very attractive solution. What the system really needs to express is the fact that some features tend to occur together in rules to the exclusion of others, a situation somewhat akin to that observed among the segments. The first idea that leaps to mind would be to utilize the same solution, using features of features *(metafeatures)* to express natural classes of features. But the cartesian product operation that is used in the feature decomposition (subdirect product form) of P is associative, and therefore it makes no difference whether we perform the feature-decomposition twice in a metafeature setup, or just once at the segment level. The solution now widely accepted in phonology (Clements, 1985; McCarthy, 1988) is to arrange the features in a tree structure, using intermediate **class nodes** to express the grouping together of some features to the exclusion of others.

Figure 3.2.1 Feature geometry tree. Rules that required the special principle of *sāvarṇya* can be stated using the *supralaryngeal class node.*

This solution, now permanently (mis)named **feature geometry,** is in fact a generalization of both the pratyāhāra and the standard feature decomposition methods. The linear intervals of the Pāṇinian model are replaced by generalized (lattice-theoretic) intervals in the subsumption lattice of the tree (Kornai, 1993), and the cartesian product appearing in the feature decomposition corresponds to the special case when the feature geometry tree is a star (one distinguished root node, all other nodes are leaves). In Ehrenfeucht's theory of generalized Boolean structures, the classes **A**,**B**, and **C** introduced above are primitive, and all other SIBRs (IBAs) can be constructed from these by arranging these in a suitable tree structure (Ehrenfeucht,). This representation theorem guarantees that using SIBRs adds no generality to the mathematical linguistic theory, which already uses feature decomposition (**B**), Pāṇinian type sets of intervals (**C**) and trivial sets (**A**).

Discussion. The segmental inventories P developed in 3.1 are clearly different from language to language. As far as natural classes and feature decomposition are concerned, many phonologists look for a single, universal inventory of features arranged in a universally fixed geometry. Since the cross-linguistic identity of features such as [nasal] is anchored in their phonetic (acoustic and articulatory) properties, rather than in some combinatorial subtleties of their intra-language phonological patterning, this search can lead to a single object, unique up to isomorphism, that will, much like Mendeleyev's periodic table, encode a large number of regularities in a compact format.

Among other useful distinctions, (Chomsky, 1965) introduces the notion of *formal* vs. *substantive universals*: using this terminology SIBRs are a formal, and a unique feature geometry tree such as the one in Fig. 3.2.1 would be a substantive universal. To the extent that phonological research suceeds in identifying a unique feature geometry, every framework, such as SIBRs, that permits a variety of geometries overgenerates. That said, any theory is interesting to people other than its immediate developers only to the extent that it can be generalized to problems other than the one it was originally intended to solve. Phonology, construed broadly as an abstract theory of linguistic form, applies not only to speech but to other forms of communication (handwritten, printed, signed, etc.) as well: in fact, phonemes, distinctive features, and feature geometry are widely used in the study of sign language, (see e.g. Sandler (1989) Liddell and Johnson (1989)): where substantive notions like nasality may lose their grip, the formal theory remains valuable.[1]

Exercise 3.2.2. What are the phonemes in the genetic code? How would you define feature decomposition and feature geometry there?

3.3 Suprasegmentals and autosegments

In 3.1 we noted that words can be partially alike even when they do not share any segments. For example, *blackbird* and *whitefish* share the property that they have a single stressed syllable, a property that was used by Bloomfield (1926) to distinguish them from multiword phrases such as *black bird* or *white fish* which will often be pronunced without an intervening pause, but never without both syllables stressed. In addition to stress, there are other *suprasegmentals* such as *tone* which appear to be capable of holding constant over multi-segment stretches of speech, typically over syllables.

Traditionally, the theory of suprasegmentals is considered harder than that of segmental phenomena for the following reasons. First, their physical correlates are more elusive: stress is related to amplitude, and tone to frequency, but the relationship is quite indirect (see Lehiste 1970). Second, informant judgements are harder to elicit: native speakers of a language often find it much harder to judge e.g. whether two syllables carry the same degree of stress than to judge whether they contain the same vowel. Finally, until recently a notation as transparent as the alphabetic notation for phonemes was lacking. In this section we will deal mainly with tone and tone-like features of speech, leaving the discussion of stress and prosody to 4.2

Staring in the seventies, phonological theory had abandoned the standard string-based theory and notation in favor of a generalization called *autosegmental* theory, which initially grew out of research on African tone languages (Leben, 1973; Goldsmith, 1976; Williams, 1976). The basic insight that tones and segments need not be fully aligned with one another but rather must be placed on separate *tiers* was soon generalized from tone to vowel harmony (Clements,

[1]However, as the abstract theory is rooted in the study of sound, we will keep on talking about 'utterances' 'phonemes' 'syllables' etc. rather than using 'gestures', 'graphemes' or other narrow terms.

1976), aspiration (Thráinsson, 1978), nasality (Hyman, 1982) and eventually, by placing all distinctive features on separate tiers, to the theory of feature geometry described in 3.2.

Autosegmental theory (so named because it encompasses not only suprasegmental but also subsegmental aspects of sound structure) generalizes the method of using a string over some alphabet P the following way: it uses k-tuples of strings connected by *association relations* that spells out which segments in the two strings are overlapping in time. We will begin with the simplest case of *bistrings* composed of two strings and an association relation. First, the strings are placed on *tiers,* which are very much like Turing-machine tapes, except the number of blank squares between nonblank ones can not be counted.

Definition 3.3.1 A **tier** is an ordered pair ($\mathbb{Z}$,N) where $\mathbb{Z}$ is the set of integers equipped with the standard identity and ordering relations '=' and '<' and N is the name of the tier.

Each tier N has its own **tier alphabet** T_N and we can assume without loss of generality that the alphabets of different tiers are disjoint except for a distinguished **blank** symbol G (purposedly kept distinct from the pause symbol #) that is adjoined to every tier alphabet. Two tiers bearing identical names can only be distinguished by inspecting their contents. We define a tier containing a string $t_0t_1...t_n$ starting at position k by a mapping that maps k on t_0 , $k+1$ on t_1 ,..., $k+n$ on t_n and everything else on G. Abstracting away from the starting position, we have

Definition 3.3.2 A tier N **containing** a string $t_0t_1...t_n$ over the alphabet $T_N \cup^* G$ is defined as the class of mappings F_k that take $k+i$ into t_i for $0 \leq i \leq n$ and to G if i is outside this range. Unless noted otherwise, this class will be represented by the mapping F_0. Strings containing any number of successive G symbols are treated as equivalent to those strings that contain only a single G at the same position. G-free strings on a given tier are called **melodies**.

Between strings on the same tier and within the individual strings temporal ordering is encoded by their usual left-to-right ordering. The temporal ordering of strings on different tiers is encoded by association relations.

Definition 3.3.3 An **association relation** between two tiers N and M containing the strings n = $n_0n_1...n_k$ and m = $m_0m_1...m_l$ is a subset of $\{0,1,...,k\} \times \{0,1,...,l\}$. An element which is not in the domain or range of the association relation is called **floating**.

Note that the association relation, being an abstract pattern of synchrony between the tiers, is one step removed from the content of the tiers: association is defined on the *domain* of the representative mappings, while content also involves their *range*. By Definition 3.3.3, there are 2^{kl} association relations possible between two strings of length k and l. Of these relations, the *No Crossing Constraint* (Goldsmith, 1976) rules out as ill-formed all relations that contain pairs (i,v) and (j,u) such that $0 \leq i < j \leq k$ and $0 \leq u < v \leq l$ are both true. We defince the **span** of an element x as with respect to some association relation A as those elements y for which (x,y) is in A. Rolling the above definitions in one, we have

Definition 3.3.4 A **bistring** is an ordered triple (f, g, A), where f and g are strings not containing G, and A is a well-formed association relation over two tiers containing f and g.

In the general case we have several tiers arranged in a tree structure called the geometry of the representation (see 3.2). Association relations are permitted only among those tiers that are connected by an edge of this tree, so if there were k tiers there will be $k-1$ relations. Thus, in the general case, we define a k-**string** as a $(2k-1)$-tuple $(s_1, ..., s_k, A_1, \ldots, A_{k-1})$ where the s_i are strings and the A_i are association relations.

Theorem 3.3.1. The number of well-formed association relations over two tiers, each containing a string of length n, is asymptotically $(6+4\sqrt{2})^n$.

Proof Let us denote the number of well-formed association relations with n symbols on the top tier and k symbols on the bottom tier by $f(n,k)$. By symmetry, $f(n,k) = f(k,n)$, and obviously $f(n,1) = f(1,n) = 2^n$. By enumerating relations according to the pair (i,j) such that no $i' < i$ is in the span of any j' and no $j'' > j$ is in the span of i, we get

$$f(n+1, k+1) = \sum_{i=1}^{k+1} f(n,i) 2^{k+1-i} + f(n, k+1) \tag{3.3}$$

From (3.3) we can derive the following recursion:

$$f(n+1,k+1) = 2f(n+1,k) + 2f(n,k+1) - 2f(n,k) \tag{3.4}$$

For the first few values of $a_n = f(n,n)$ we can use (3.4) to calculate forward: $a_1 = 2$, $a_2 = 12$, $a_3 = 104$, $a_4 = 1008$, $a_5 = 10272$, $a_6 = 107712$, and so on. Using (3.4) we can also calculate backwards and define $f(0,n) = f(n,0)$ to be 1 so as to preserve the recursion. The generating function

$$F(z,w) = \sum_{i,j=0}^{\infty} f(i,j) z^i w^j \tag{3.5}$$

will therefore satisfy the equation

$$F(z,w) = \frac{1 - \frac{z}{1-z} - \frac{w}{1-w}}{1 - 2z - 2w + 2zw} \tag{3.6}$$

If we substitute $w = t/z$ and consider the integral

$$\frac{1}{2\pi i} \int_C \frac{F(z, t/z)}{z} dz \tag{3.7}$$

this will yield the constant term $\sum_{n=0}^{\infty} f(n,n) t^n$ by Cauchy's formula. Therefore, in order to get the generating function

$$d(t) = \sum_{i=0}^{\infty} a_n t^n \tag{3.8}$$

we have to evaluate

$$\frac{1}{2\pi i}\int_C \frac{1-\frac{z}{1-z}-\frac{t/z}{1-t/z}}{z(1-2z-2t/z+2t)}dz \tag{3.9}$$

which yields

$$d(t)=1+\frac{2t}{\sqrt{1-12t+4t^2}} \tag{3.10}$$

$d(t)$ will thus have its first singularity when $\sqrt{1-12t+4t^2}$ vanishes at $t_0 = (3-\sqrt{8})/2$, yielding the desired asymptotics

$$a_n \approx (6+4\sqrt{2})^n \tag{3.11}$$

The base 2 logarithm of this number, $n \cdot 3.543$, measures how many bits we need to encode a bistring of length n. Note that this number grows linearly in the length of the bistring, while the number of (possibly ill-formed) association relations was 2^{n^2}, with base 2 log growing quadratically. Association relations in general are depicted as bipartite graphs (pairs in the relation are called **association lines**), and encoded as two-dimensional arrays (the incidence matrix of the graph). However, the linear growth of information content suggests that well-formed association relations should be encoded as one-dimensional arrays or strings. Before turning to this matter in 3.4, let us first consider two particularly well-behaved classes of bistrings. A bistring is **fully associated** if there are no floating elements, and **proper** if the span of any element on one tier will form a single substring on the other tier (Levin, 1985). Proper relations are well-formed but not necessarily fully associated.

Let us define $g(i,j)$ as the number of association relations containing no unassociated (floating) elements, and define b_n as $g(n,n)$. By similar counting arguments as those used above, we get the recursion

$$g(n+1,k+1)=g(n+1,k)+g(n,k+1)+g(n,k) \tag{3.12}$$

Using this recursion the first few values of b_n can be computed as 1, 3, 13, 63, 321, 1683, 8989, and so on. Using (3.12) we can calculate backwards and define $g(0,0)$ to be 1 and $g(i,0)=g(0,i)$ to be 0 (for $i>0$) so as to preserve the recursion. The generating function

$$G(z,w)=\sum_{i,j=0}^{\infty} g(i,j)z^i w^j \tag{3.13}$$

will therefore satisfy the equation

$$G(z,w)=\frac{1-z-w}{1-z-w-zw}=1+\frac{zw}{1-z-w-zw} \tag{3.14}$$

Again we substitute $w=t/z$ and consider the integral

$$\frac{1}{2\pi i}\int_C \frac{G(z,t/z)}{z}dz \tag{3.15}$$

which will yield the constant term $\sum_{n=0}^{\infty} g(n,n)t^n$ by Cauchy's formula. Therefore, in order to get the generating function

$$e(t) = \sum_{i=0}^{\infty} b_n t^n \tag{3.16}$$

we have to evaluate

$$\frac{1}{2\pi i}\int_C \frac{1}{z} + \frac{t}{z(1-z-t/z-t)} dz = 1 - \frac{t}{2\pi i}\int_C \frac{dz}{(z-p)(z-q)} \tag{3.17}$$

which yields

$$e(t) = 1 + \frac{t}{\sqrt{1-6t+t^2}} \tag{3.18}$$

Notice that

$$e(2t) = 1 + \frac{2t}{\sqrt{1-6\cdot 2t+(2t)^2}} = d(t) \tag{3.19}$$

and thus

$$\sum_{i=0}^{\infty} b_n(2t)^n = \sum_{i=0}^{\infty} a_n t^n \tag{3.20}$$

Since the functions $d(t)$ and $e(t)$ are analytic in a disk of radius $1/10$, the coefficients of their Taylor series are uniquely determined, and we can conclude

$$b_n 2^n = a_n \tag{3.21}$$

meaning that fully associated bistrings over n points are only an exponentially vanishing fraction of all well-formed bistrings. In terms of information content, the result means that fully associated bistrings of length n can be encoded using *exactly* one bit less per unit length than arbitrary well-formed bistrings.

Exercise* 3.3.1 Find a 'bijective' proof establishing (3.21) by direct combinatorial methods.

Now for proper representations, denoting their number by $h(n,k)$ the generating function $H = H(z,w)$ will satisfy a functional equation

$$H - zH - wH - 2zwH + zw^2H + z^2wH - z^2w^2H = r(z,w) \tag{3.22}$$

where $r(z,w)$ is rational. Using the same diagonalizing substitution $w = t/z$ we have to evaluate

$$\frac{1}{2\pi i}\int_C \frac{s(z,t)}{z(1-z-t/z-2t+t^2/z+tz-t^2)} dz \tag{3.23}$$

Again, the denominator is quadratic in z, and the radius of convergence is determined by the roots of the discriminant

$$(t^2+2t-1)^2 - 4(t-1)(t^2-t) = t^4 + 10t^2 - 8t + 1 \tag{3.24}$$

The reciprocal of the smallest root of this equation, approximately 6.445 gives the base for the asymptotics for c_n, the number of proper bistrings over n points. By taking base 2 logarithm, we have

Theorem 3.3.2 The information content of a fully associated (proper) well-formed bistring is 2.543 (2.688) bits per unit length.

Exercise 3.3.2. Count the number of well-formed (fully-associated, proper) k-strings of length n assuming each tier alphabet has only one element besides G.

Sets of well-formed (fully associated, proper) bistrings will be called well-formed (fully associated, proper) **bilanguages**: these can undergo the usual set-theoretic operations of **intersection, union** and **complementation** (relative to the 'universal set' of well-formed, fully associated, resp. proper bistrings). **Reversal** (mirror image) is defined by reversing the constituent strings together with the association relation. The concatenation of bistrings is defined by concatenating both the strings and the relations:

Definition 3.3.5 Given two bistrings (f, h, A) and (k, l, B) on tiers N and M their **concatenation** (fk, hl, AB) is constructed via the tier-alphabet functions $F_0, H_0, K_{|f|}$, and $L_{|g|}$ as follows. $FK_0(i) = F(i)$ for $0 \leq i < |f|$, $K_{|f|}(i)$ for $|f| \leq i < |f| + |k|, G$ otherwise. $HL_0(j) = H(j)$ for $0 \leq j < |k|$, $L_{|k|}(j)$ for $|k| \leq j < |f| + |k|$, G otherwise. Finally, $AB = A \cup \{(i + |f|, j + |k|) : (i, j) \in B\}$

Notice that the concatenation of two connected bistrings will not be connected (as a bipartite graph). This is remedied by the following

Definition 3.3.6 Given two bistrings as in 3.3.5, their **t-catenation (b-catenation)** is defined as (fk, hl, AtB) (fk, hl, AbB) where $AtB = AB \cup \{(|f| - 1, |k|)\}$ $(AbB = AB \cup \{(|f|, |k| - 1)\})$

Using phonological terminology, in t-catenation the last element of the *t*op tier of the first bistring is *spread* on the first element of the bottom tier of the second bistring, and in b-catenation the last element of the *b*ottom tier of the first string is spread on the first element of the top tier of the second bistring.

The only autosegmental operation that is not the straightforward generalization of some well-known string operation is that of **alignment**. Given two bistrings $x = (f, g, A)$ and $y = (g, h, B)$, their alignment $z = x \parallel y$ is defined to be (f, h, C), where C is the relation-composition of A and B: in other words, the pair (i, k) will be in C iff there is some j such that (i, j) is in A and (j, k) is in B. Now we are in a position to define projections: these involve some subset S of the tier alphabet T. A **projector** $P_S(h)$ of a string $g = h_0h_1...h_m$ with respect to a set S is the bistring (h, h, Id_S), where (i, j) is in Id_S iff $i = j$ and h_i is in S. The **normal bistring** $I(h)$ corresponding to a string h is simply its projector with respect to the full alphabet: $I(h) = P_T(h)$. A **projection** of a string with respect to some subalphabet S can now be defined as the alignment of the corresponding normal bistring with the projector.

The alignment of well-formed bistrings is not necessarily well-formed, as the following example shows. Let f = ab, g = c, h = de and suppose that the following associations hold: $(0, 0)$ and $(1, 0)$ in x; $(0, 0)$ and $(0, 1)$ in y. By definition, C should contain $(0, 0), (0, 1), (1, 0)$, and $(1, 1)$ and will thus violate the No Crossing Constraint. Note also that a projector, as defined here, will not be necessarily proper. In order to capture the phonologically relevant sense of properness is useful to relativize the above definition to 'P-bearing units' (Clements and Ford 1979). We will say that a bistring (f, h, A) is **proper with respect to a subset** S of the tier-alphabet T underlying the string h, iff $(f, h, A) \parallel P_S(h)$ is proper.

3.4 Phonological computation

The 'standard' theory of phonology (Chomsky and Halle 1968) enumerates the well-formed strings in a generative fashion (see 2.1) by selecting a set E of *underlying forms* and some context-sensitive rules R that manipulate the underlying forms to yield the permissible *surface forms.* Because features are an effective (though imperfect, see 3.2) means of expressing natural classess, rules that typically arise in the phonology of natural languages can be stated more economically directly on features, and in fact phonologists rarely have any reason to manipulate strings of phonemes (as opposed to strings of feature bundles). Nevertheless, in what follows we can assume without loss of generality that the rules operate on segments, because a rule system employing features can always be replaced by a less economical, but equivalent rule system that uses only segments.

Exercise 3.4.1 Post-stress destressing. In our example language there are five unstressed vowels a e i o u and five stressed vowels A E I O U. Whenever two stressed vowels would come into contact, the second one loses its stress: [+stress] $\rightarrow$ [-stress]/[+stress]_. How many string-rewriting rules are needed to express this regularity without using feature decomposition?

In many problems, such as speech recognition, we are more interested in the converse task of computing the underlying form(s) given some surface form(s). Because of the context-sensitive character of the rules, the standard theory gave rise to very inefficient implementations: although in principle generative grammars are neutral between parsing and generation, the membership problem of CSGs is PSPACE-complete (see Theorem 2.3.2) and in practice no efficient parsing algorithm was found. Context-sensitive phonological rule systems, though widely used for generation tasks (Hunnicutt, 1976; Hertz, 1982), were too inefficient to be taken seriously as parsers.

The key step in identifying the source of the parsing difficulty was Johnson's (1970) finding that, as long as phonological rules do not reapply within their output, it is possible to replace the context sensitive rules by finite state transducers (FSTs). That such a condition is necessary can be seen from the following example: $S \rightarrow ab$; $\lambda \rightarrow ab/a\_b$. Starting from S these rules would generate $\{a^n b^n | n \in \mathbb{N}\}$, a language known not to be regular (see Theorem 3.4.1 below). To show that once the condition is met, context sensitive rules can be replaced by FSTs we first need to establish some facts.

We define **regular relations** analogously to the case of regular languages: given two alphabets P, Q a relation $R \subset P^* \times Q^*$ is regular iff it is finitely generated from finite sets by the operations of union, concatenation, and Kleene-closure. (These operations are defined componentwise on relations). Regular relations are in the same relationship to FSTs as regular languages to FSAs: in fact, it is convenient to think of languages as unary relations, and FSA as n-tape automata (transducers) with $n = 1$. In such automata, all tapes are read only, and the automaton can change internal state when no tape is advanced (λ-move) or when one or more of the tapes is advanced by one square.

Definition 3.4.1 A **finite state transducer (FST)** is a quadruple (S, s, F, T) where S is a finite set of states, $s \in S$ is the starting state, $F \subset S$ is the set of final (accepting) states, and T is a set of **transitions** of the form $(b, a, l_1, \cdots, l_n)$ where b and a are the states *b*efore and *a*fter the move, and the l_j are letters scanned on the jth tape during the move or λ. When for every b and $l_1, \cdots, l_n$ there is at most one $a \in S$ such that $(b, a, l_1, \cdots, l_n) \in T$ the transducer is **deterministic**, and when λ never appears in any transition it is called **length-preserving**

An n-tuple of words is **accepted** by an FST iff, starting with n tapes containing the n words, the reading heads positioned to the left of the first letter in each word, and the FST in the initial state, the automaton has a sequence of legal moves (transitions in T) that will advance each reading head to the right of the word on that tape, with the automaton ending in a final (accepting) state. When two words (or two n-toples of words) land an FST in the same state (or in the case of nondeterninistic FSTs, the same set of states) starting from the initial state, we call them *right congruent*. Significantly, this notion can be defined without reference to the automaton, based solely on the language it accepts.

Definition 3.4.2 Let L be a language (of n-tuples), x, y are **right congruent** iff for all z either both xz and yz are in L or both of them are outside L. We define **left congruence** analogously, by requiring for all z both zx and zy to be (or both not to be) in L. Finally, the **syntactic congruence** (also known as **Nerode equivalence**) is defined as the smallest equivalence that is finer than both left and right congruence.

Exercise 3.4.1 Prove that right congruence, as defined by FST landing sites, is the same relation as defined through the accepted language L. Prove that left and right congruence are equivalence relations. What can we say about the paths through an FST and left congruent elements?

The key property of regular languages (and relations) is that the (right) congruence they define has finitely many equivalence classes (this is also called having **finite index**). If a language (of n-tuples) has finite index, we can use the equivalence classes as states of the accepting FST, with transitions defined in the obvious manner. Conversely, languages accepted by some FST obviously have finite index. Finite languages have finite index, and if two languages have finite index, so will their union, concatenation, and Kleene-closure. Thus we have

Theorem 3.4.1 Kleene's Theorem: an n-place relation is regular iff it is accepted by an n-tape FST.

Other properties also studied in formal language theory, such as closure under intersection or complementation, will not necessarily generalize from unary to n-ary relations. For example, the binary relations $\{(a^n, b^n c^*)|n \in \mathbb{N}\}$ and $\{(a^n, b^* c^n)|n \in \mathbb{N}\}$ intersect to yield $\{(a^n, b^n c^n)|n \in \mathbb{N}\}$ which is not regular. However, the length-preserving relations are a special case where closure under intersection and set-theoretic difference holds. The method of expressing context sensitive rules by regular binary relations exploits this fact by adding a new symbol, ϵ, to the alphabet, and using it as padding to maintain length wherever needed. The key idea is that the composition of regular relations (lenght-preserving or otherwise) is again regular, so the workings of a context sensitive rule $A \rightarrow u/l\_r$ can be decomposed into simple regular (and when needed, lenght-preserving) steps such as the introduction and eventual deletion of temporary brackets that keep track of the locus of rule application (see Kaplan and Kay (1994) for a detailed discussion).

Although the method of cascaded FST, as well as the related method of parallel FSTs (Koskenniemi, 1983) play a central role in modern computational phonology, we will not pursue their development here, as it is covered in most newer textbooks of computational linguistics and there are volumes such as Roche and Schabes (1997) and Kornai (1999) devoted entirely to this subject. Rather, we turn our attention to the formalization of the autosegmental theory described in 3.3, where regular languages are generalized so as to include the *association relation.* Because of the central role of the finitely generated case, our first task is to identify the family of *regular* or *finite state* bilanguages. We do this based on the finite index property:

Definition 3.4.3 The **syntactic congruence** $\equiv_L$ generated by a bilanguage L contains those pairs of bistrings (α, β) which are freely substitutable for one another i.e. for which $\gamma\alpha\delta \in L \Leftrightarrow \gamma\beta\delta \in L$. When γ (δ) is fixed as the empty string, we will talk of right (left) congruence. A k-language is **regular** iff it gives rise to a (right)congruence with finitely many classes.

Example 3.4.1 One-member tier alphabets $\{a\} = T_N, \{b\} = T_M$ and empty association relations. Let us denote the bistring $(a^i, b^j, \emptyset)$ by $< i, j >$. The bilanguage $B = \{< i, i > |i > 0\}$ is not regular, because the bistrings $< 1, k >$ all belong in different (right)congruence classes for $k = 1, 2, ...,$ as can be seen using $\delta =< l, 1 >$: if $k \neq l$ then $< 1, k >< l, 1 >\notin B$ but $< 1, l >< l, 1 >\in B$.

Discussion B can be expressed as the Kleene-closure $^+$ of the one-member bilanguage $\{< 1, 1 >\}$. However, the appropriate closure operation for bilanguages involves not only concatenation but t-catenation and b-catenation as well.

Definition 3.4.4 A **biautomaton** is defined as a 6-tuple (S, U, V, i, F, T) where S is a set of states, U and V are the alphabets of the two tapes, i is the initial state, F is the set of final (accepting) states, and T is the transition function. If we denote the square under scan on the upper tape by `x`, the square under scan on the lower tape by `y`, the transition function from a given state will depend on the following factors:

(i) Is there a symbol on square `x`, and if so, what symbol? (If not, we use the special symbol G introduced in 3.3.)

(ii) Is there a symbol on square `y`, and if so, what symbol?

(iii) Are the squares associated?

(iv) Are there further association lines from `x` to some symbol after `y`?

(v) Are there further association lines from `y` to some symbol after `x`?

The transition function T, depending on the present state, the letters under scan, and the presence of association between these letters, will assign a new state, and advance the tapes in accordance with the following rule:

(3.25) If there are no further association lines from `x` and `y`, both tapes can move one step to the right, if there are further association lines from `x`, only the bottom tape can move, and if there are further association lines from `y`, only the top tape can move.

In other words, the current position of the reading heads can always be added to the association relation without violating the No Crossing Constraint. We will shortly define *coders* as automata in which (3.25) is augmented by the requirement of moving the tapes as fast as possible, but for the moment we will leave the advancement pattern of the tapes nondeterministic. To specify which tape will move, it is best to separate out the transition function into three separate components: one that gives the new state provided a top move t was taken, one that gives the new state provided a bottom move b was taken, and one that gives the new state provided a full move f was taken. Here and in what follows $x[y,t]$ denotes the result state of making a top move from state x upon input y, and similarly for $x[y,b]$ (bottom move) and $x[y,f]$ (full move). In general there can be more than one such state, and we do not require that only a top, bottom, or full move be available at any given point – it might still be the case that only one of these moves is available, because that is what the association pattern dictates, but there is no general requirement enforcing uniqueness of next move.

The transition function at state u $\in$ S is **scanning independent** iff for every possible scanning of a string α takes the machine from u to the same state x=u[α]. In particular, the machine must be able to perform a full move as a top move followed by a bottom move, or as a bottom move followed by a top move. Two full moves should be replaceable by *ttbb*, *bbtt*, *tbtb*, *bttb*, *tbbt*, *btbt* and similarly for fff and longer sequences of moves. A biautomaton will be called a **finite autosegmental automaton** iff its transition function is scanning independent at every state. It will be called a **coder automaton** if advancement is by the following deterministic variant of the rule given above:

(3.26) If there are no further symbols on either tape, the machine stops. If there are no further symbols on one tape, the other tape is advanced by one.

> If there are no further association lines from `x` and `y`, both tapes move one step to the right, if there are further association lines from `x`, only the bottom tape moves, and if there are further association lines from `y`, only the top tape moves, provided the move does not result in scanning G. (The case when there are further lines both from `x` and `y` can not arise, since such lines would cross.)

Coders can be used to assign a unique linear string, called the *scanning code* in Kornai (1991 ch 1.4), to every association relation. Let us denote a top move by t, a bottom move by b, the presence of an association line by 1 and its absence by 0. To assign a unique linear string over the alphabet $\{t, b, 0, 1\}$ to every association relation, it is sufficient to record the top and bottom moves of the coder, leaving full moves unmarked, together with the association lines or lack thereof encountered during the scan. Since the scanning code of a bistring of length n can contain at most $4n$ symbols of this alphabet, we have a constructive proof that the information content of well-formed bistrings is at most 8 bits per unit length. While this is very inefficient compared to the optimum 3.543 bits established by Theorem 3.3.1, the constructive nature of the encoding lends further support to the claim in 3.3 that multilinear representations are linear (rather than quadratic or other polynomial) data types.

So far we have four families of bilanguages: those accepted by finite autosegmental automata (deterministic or nondeterministic) will be collected in the families **RD** and **RN**, those accepted by biautomata will be collected in the family **BA**, and those accepted by coders will be collected in the family **CA**. Clearly we have **RN** $\subset$ **BA**, **RD** $\subset$ **BA**, **CA** $\subset$ **BA**, since both scanning independent and coder automata are special cases of the general class of biautomata. Let us first establish that nondeterminism adds no power – for further "geographic" results see Theorem 3.4.4.

Theorem 3.4.2 RD = **RN**.

Proof Same as for the 1-string case. Instead of the state set S of the nondeterministic automaton consider its power set 2^S and lift the nondeterministic transition function T to a deterministic transition function D the following way: for $i \in$ S define D($\{i\}$) as $\{$T(i)$\}$, and for $X \subset S$, $D(X) = \bigcup_{i \in X} D(\{i\})$.

Note that the proof will not generalize to coders, because different nondeterministic options can lead to different positioning of the heads. However, if the transition function is scanning independent, different positionings of the heads can always be exchanged without altering the eventual state of the machine. Now we can prove Kleene's Theorem that the family of languages **R** characterized by the finite index property is the same as **RN** and **RD**.

Theorem 3.4.3 A bilanguage L is regular iff it is accepted by a regular autosegmental automaton.

Proof ($\Leftarrow$) If L is accepted by a regular autosegmental automaton, it is also accepted by a deterministic regular autosegmental automaton (which can be constructed by the method outlined above) and further it can be accepted by a reduced automaton in which no two states have exactly the same transition function (for such states can always be collapsed into a single state). We claim that there will be as many right congruence classes in $\equiv_L$ as there are states in a minimal (reduced, deterministic, regular) autosegmental automaton $A = (S, U, V, i, F, T)$.

To see this, define $\alpha \equiv_A \beta$ iff for every scanning of α starting in the initial state i and ending in some state j there is a scanning of β starting in i and also ending in j and vice versa. Clearly, $\equiv_A$ is an equivalence relation, and $\alpha \equiv_A \beta \Rightarrow \alpha \equiv_L \beta$. If $\alpha \not\equiv_A \beta$, there must exist a state j such that at least one scanning of one of the bistrings, say α, will lead from i to j, but no scanning of β will ever lead from i to j. Since A is deterministic, scanning β will lead to some state k≠j. We will show that there exists a string δ such that from j we get to an accepting state by scanning δ and from k we get to a non-accepting state (or conversely), meaning that $\alpha\delta \in L$ but $\beta\delta \notin L$ (or conversely), so in either case $\alpha \not\equiv_L \beta$.

Call to states p and q *distinguishable* iff there exists a string δ such that starting from p, scanning δ leads to an accepting state, but starting from q, scanning δ leads to a rejecting state or vice versa. *In*distinguishability, denoted by I, is an equivalence relation: clearly pIp for every state p, and if pIq, also qIp. For transitivity, suppose indirectly that pIq and qIr, but p and r are distinguishable, i.e. there is a string δ for which p[δ] is accepting but r[δ] is not. Now, q[δ] is either accepting or rejecting: in the former case, qIr was false, and in the latter, pIq was false, contradiction. Further, in a minimal automaton there can be no two (or more) indistinguishable states, for such states could be collapsed into a single state without altering the accepted bilanguage. Since j and k above are not equal, they are distinguishable by some δ, QED.

($\Rightarrow$) To prove the "only if" part of the theorem, we have to show that if a bilanguage L gives rise to a finite right congruence, it is accepted by some regular autosegmental automaton. We will construct the states of the automaton from the congruence classes of the equivalence relation. Let us denote the congruence class of a bistring α under $\equiv_L$ by (α). The initial state of the machine is the congruence class of the empty bistring, ( ), and the transition function from state (α) is defined for top transitions from (α) as the congruence class (αtT) (t-catenation) and similarly the result state of a bottom transition from (α) will be the congruence class (αbB) (b-catenation). Thus top (bottom) transitions are nondeterministic: there are as many result states as there are congruence classes for each member of the top (bottom) tier alphabet. For ordinary concatenation of a bistring β, the result is defined by the class ($\alpha\beta$), in order to guarantee scanning independence.

Finally, the accepting states of the automaton are defined as those congruence classes that contain the members of L – this is well-defined because if $\alpha \equiv_L \beta$, both must be members of L or both must be outside L, meaning that L is a union of congruence classes. What remains to be seen is that the bilanguage M accepted by the automaton defined here is the same as the bilanguage L we started with. First let us take a bistring α included in L – since (α) is an accepting state, it follows that α is also in M. Next let us take a bistring β not in L – since (β) is not an accepting state it would follow that β is not in M if we can show that no scanning path would lead to any state other than (β). This can be done by induction on the length (defined as the maximum of the length of the top and bottom strings) of β (see Kornai 1991, ch 1.6).

Discussion The class **R** of regular k-languages is closed under all Boolean operations, while the class of regular k-relations is not: this is due to the fact that finite autosegmental automata can always be determinized (Theorem 3.4.2) while FSTs in general can not. Determinization creates a machine where complementing the set of accepting states leads to accepting the complement language – if a language is only accepted by a nondeterministic acceptor this simple method fails, and the complement language may not have an equally simple characterization. Since both families are closed under union, (non)closure under complementation will guarantee, by De Morgan's law, (non)closure under intersection. In this respect, the bilanguage families **BA** and **CA** are closer to regular relations than the family **R**.

Example 3.4.2 Consider the bilanguage $T = \{< i, j > | i > j\}\{(a, b, \{(0, 0)\})\}$ – it contains those bistrings that have i floating features on the top tier, j floating features on the bottom tier, followed by an end marker which is simply a feature a on the top tier associated to a feature b on the bottom tier. Clearly, if $i - j \neq i' - j'$, $< i, j > \not\equiv_T < i', j' >$ so T is not in **R**. However, it is in **CA**, since the following automaton will accept it:

(3.26)

in state	from `x` to `y`	from `x` to `z>y`	from `y` to `w>x`	automaton will
0	absent	absent	absent	stay in 0
0	absent	absent	present	go to 1
1	absent	absent	present	stay in 1
1	present	absent	absent	go to 2
2	any	any	any	go to 3

With 2 as the only accepting state, the machine will accept only those strings whose scan puts the machine in 2, but not further. To get into 2, the last thing the machine must encounter is a single association line (the end marker) in state 1. To get into state one, the machine can make a number of top moves over floating elements (this is the loop over state 1) preceded by a number of full moves over floating elements (this is the loop over state 0). Note that this is not scanning independent: no provision was made for top and bottom moves to replace full moves out of state 0.

What Example 3.4.2 shows is that **CA** is not contained in **R**. It is, of course, contained in **BA**, and the bilanguage B of Example 3.4.1 introduced above shows that the containment is proper. The biautomaton that accepts this bilanguage is trivial: it contains only one state and only full advance is permitted (and that only when no association lines are present). To see that no coder can accept this bilanguage, suppose indirectly that an n-state coder A accepts B. The bistrings $< k, k >$ are all accepted ($k = 1, 2, 3, ..., n+1$), so there is at least one accepting state f which accepts both $< i, i >$ and $< j, j >$, $1 \leq i < j \leq n+1$ by the pigeonhole principle. Let $j - i = p$, and consider the bistring $< j, i >$. In the first i steps, we arrive in f, and in the next p steps we make legal top moves (since we are at the end of the bottom string) which are indistinguishable from legal full moves. But p full moves would take as back to f, which is an accepting state, so p top moves also take us back to f, meaning that $< j, i >$ is accepted by A, contradiction. To complete our "geographic survey", note that **R** is not contained in **CA**: this can be seen e.g. by considering the regular bilanguage $D = \{< 1, j > | j > 0\}$. Collecting these results gives us

Theorem 3.4.4 Both **R** and **CA** are properly contained in **BA**, but neither is contained in the other.

To summarize, **R** is closed under union and intersection as the standard direct product construction shows, and also under complementation, as can be trivially established both from the characterization by automata. Boolean operations thus offer no surprises, but string operations need to be revised. If we use concatenation as the only k-string composition operation, there will be an infinite number of further undecomposable structures, such as the bistrings resulting from the spreading of a single element on the bottom (top) tier. These structures, and many others, have no structural break in them if indeed concatenation was the only possibility: that is why we introduced t-catenation and b-catenation above. Regular expressions also work:

Theorem 3.4.5 Every bilanguage accepted by an autosegmental automaton can be built up from the elementary bistrings $(x, y, \emptyset)$ and $(x, y, \{(0, 0\})$ by union, t-catenation, b-catenation, concatenation, and Kleene-closure.

Once these operations are available for creating larger bistrings from two successive bistrings, **Kleene-closure** will include these as well. This way the oddness of the bilanguage B introduced in Example 3.4.1 above disappears: $B = \{< i, i > | i > 0\}$ is *not* the Kleene $^+$ of $< 1, 1 >$, because the closure means arbitrary many catenation operations *including* t-catenations and b-catenations. The Kleene $^+$ of $< 1, 1 >$ is really the set of all well-formed bistrings (over one-letter tier alphabets) which is of course regular. From the characterization by automata it easily follows that the concatenation, t-catenation, b-catenation, and Kleene-closure of regular bilanguages is again regular. Standard proofs will also generalize for closure under (inverse) homomorphisms and (inverse) transductions.

As we discussed earlier, transductions play a particularly important role in replacing context sensitive phonological rules by finite state devices. For this reason we combine the notions of k-strings and n-place regular relations and state this last result (for the proof, modeled after Salomaa 1973 ch. 4, see Kornai 1991 ch 2.5.2) as a separate

Theorem 3.4.6 If L is a regular bilanguage and $B = (S, I, O, i, F, T)$ a generalized bisequential mapping, the image $(L)B$ of L under B is also a regular bilanguage.

Autosegmental rules can be grouped into two categories: rules affecting the strings stored on the various tiers, and rules affecting the association relations. The two categories are not independent: rules governing the *insertion* and *deletion* of symbols on the various tiers are often conditioned on the association patterns of the affected elements or their neighbors, while rules governing the *association* and *delinking* of elements of different tiers will often depend on the contents of the tiers.

Repeated insertion is sufficient to build any string symbol by symbol – deletion rules are used only because they make the statement of the grammar more compact. Similarly, repeated association would be sufficient for building any association relation line by line, and delinking rules are only used to the extent that they simplify the statement of phonological regularities. A typical example would be a *degemination* rule that will delink one of the two timing slots (a notion that we will develop further in 4.2) associated to a segment *k,* if it was preceded by another segment *s,* and also deletes the freed slot. In pre-autosegmental theory the rule would be something like $k[+long] \rightarrow k[-long]/s\_$ (the features defining k and s are replaced by k and s to simplify matters), and in autosegmental notation we have

```
(3.25)     s k      s k
          / /|  ->  | |
         X X X      X X
```

The regular bi-transducer corresponding to this rule will have an *input bistring,* an *output bistring,* and a *finite state control.* The heads scanning the bistrings and the association relations are read only. If we denote the bistring on the left hand side of the arrow by β and that on the right hand side by β', the bi-transducer accepts those pairs of bistrings that have the form $(\alpha\beta\gamma, \alpha\beta'\gamma)$ and those pairs (δ, δ) that meet the rule vacuously (β is not part of δ).

Exercise 3.4.2 Verify that the regular bi-transducer defined with the finite state control in (3.26) accepts those and only those pairs of bistrings (ρ, σ) in which σ is formed from ρ by applying (3.25).

(3.26)

in state	input bistring	output bistring	heads move	result state
0	s1X	s1X	f/f	1
0	q!=s1X	q	m/m	0
0	q!=s1X	r!=q	m/m	-
1	k1X	k1X	b/0	2
1	q!=k1X	q	m/m	0
1	q!=k1X	r!=q	m/m	-
2	k1X	k1X	m/m	0
2	q!=k1X	r!=q	m/m	-

Chapter 4

Morphology

Morphology, the study of the shape and structure of words, is the historically oldest layer of linguistics: most of the early work on Sanskrit (Pāṇini, ∼520 BCE – ∼460BCE), Greek (Dionysius Thrax, ∼ 166 BCE – ∼ 90 BCE), and Latin (Stilo, ∼152–74 BCE, Varro, 116–27 BCE) concerns morphological questions. It is also a field that brings into sharp relief what are perhaps the most vexing aspects of linguistics from a mathematical perspective: radical typological differences, flexible boundaries, and near-truths.

Mild typological differences are common to most fields of study: for example, the internal organs of different primates are easily distinguished by experts, yet they differ only mildly, so that a person knowledgeable about gorillas in general and human livers in particular can make a reasonable guess about the position, shape, size and functioning of gorilla livers without ever having seen one. *Radical typological differences* are much less common: continuing with the analogy, one knowledgeable about the internal sex organs of males but not of females would have a hard time to guess their position, shape, size or functioning. In morphology, radical typological differences abound: no amount of expert knowledge about Modern English is sufficient to make a reasonable guess e.g. about the case system of Modern Russian, in spite of the fact that the two languages descended from the same IndoEuropean origins. Mathematics, on the whole, is much better suited for studying mild (parametric) typological differences than radical ones. We exemplify the problem and discuss a possible solution in section 4.1, which deals with prosody in general and the typology of stress systems in particular.

In mathematics, *flexible boundaries* are practically unheard of: if in one case some matter depends on arithmetic notions we are unlikely to find other cases where the exact same matter depends on topological notions. It is easier to find examples in computer science, where the same functionality, e.g. version control of files, may be provided as part of the operating system in one case, or as part of the text editor in another. In morphology, flexible boundaries are remarkably common: the exact same function, forming the past tense, may be provided by regular suffixation (*walk* → *walked*), by ablaut (*sing* → *sang*), or by suppletion (*go* → *went*). We exemplify the problem in 4.2, where we introduce the notions of derivation and inflection.

Finally, we call *near-truths* those regularities that come tantalizingly close to being actually true, yet are detectably false with the available measurement techniques. A well known example is the "law" that atomic weights are integer multiples of that of hydrogen: for example the helium/hydrogen atomic weight ratio is 3.971, nitrogen/hydrogen 13.897. Near-truths are so powerful that one is inclined to disregard the discrepancies: from a chemistry-internal perspective everything would be so much simpler if atomic weights were truly subject to the law. In fact, we have to transcend the traditional boundaries of chemistry, and gain a good understanding of isotopes and nuclear physics to see why the law is only nearly true. Unfortunately, there is no good candidate for a deeper theory that can clean up linguistic near-truths: as we shall see repeatedly, all forms of "cognitive" and "functional" explanations systematically fall short. Therefore, in 4.3 we look at a solution, *optimality theory,* which builds near-truths into the very architecture of linguistics.

Since it will be necessary to give examples from languages that are not always familiar to the average undergraduate or graduate student of mathematics, we decided, somewhat arbitrarily, to treat every language that does not have a chapter in Comrie (1990) as requiring some documentation, especially as the names of languages are subject to a great deal of variation, both in in spelling and in that different groups of people may use very different names for one and the same language. Such languages are always identified by their three-letter Ethnologue code (14th Edition, Grimes (ed) 2000)

4.1 The prosodic hierarchy

The marking of certain substructures as belonging to a certain prosodic domain such as the *mora, syllable, foot,* or *prosodic word* is an essential part of phonological representations for two interrelated reasons. First, because a great number of phonological processes or constraints make reference to such domains: for the syllable (in English) see Kahn (1976), for the foot (in Japanese) see Poser (1990). Second, because the domains themselves can carry feature information that can't properly be attributed to any smaller constituent inside the domain: *stress* and *boundary tones* provide widely attested examples, though some readers may also be familiar with *emphasis* (pharyngealization) in Arabic, Aramaic [CLD], and Berber [TZM] (Jakobson 1957, Hoberman 1987, Elmedlaoui 1985).

It should be said at the outset that our understanding of the prosodic hierarchy is not yet sufficient: for example, it is not clear that moras are constituents of the syllables the same way syllables are constituents of feet, whether notions like extrametricality or ambisyllabicity are primitive or derived, whether abstract units of stress can be additively combined (as in the *grid-based* theories starting with Liberman 1975), and so on. Even so, there is no doubt that the prosodic hierarchy plays an absolutely pivotal role in phonology and morphology as the guardian of well-formedness, in a manner broadly analogous to the use of *checksums* in digital signal transmission. Arguably, the main function of phonology is to repair the damage that morphological rules, in particular concatenation, would cause.

4.1.1 Syllables

There are three different approaches one may wish to consider for a more precise definition of the syllable. First, we can introduce a *syllabic alphabet* or *syllabary* that is analogous to the phonemic alphabet introduced in 3.1, but without the requirement that distinct syllables show no similarity (segmental overlap). This approach is the historically oldest, going back at least a thousand years to the syllabic writing systems such as found in Japanese Hiragana, and arguably much earlier with Brahmi (5th century BCE) and the Cypriot syllabary (15th century BCE), though in fact many of the early scripts are closer to being mora-based than syllable-based.

Second, we can introduce explicit boundary markers, such as the - or · used in dictionaries to indicate syllable boundaries, a notation that goes back at least to the 19th century. An interesting twist on the use of boundary markers is to permit *improper* parentheses, e.g. to denote by $[a(b]c)$ the case where b is said to be *ambisyllabic* (belonging to both syllables ab and bc). The notion of ambisyllabicity receives a slightly different formulation in autosegmental theory (see 3.3). Note that improper parentheses could describe cases like $[a(bc]d)$ where more than one element is ambiguously affiliated, while the autosegmental well-formedness conditions would rule this out.

Finally, we can use tree structure notation, which is more recent than the other two, and has the advantage of being immediately familiar to contemporary mathematicians and computer scientists, but is incapable of expressing some of the subtleties like ambisyllabicity that the earlier notations are better equipped to deal with. One such subtlety, easily expressible with autosegmental notatation or with improper parentheses, is the notion of *extrametricality,* meaning that the parse tree simply fails to extend to some leaves.

The unsettled notation is much more a reflection of the conceptual difficulties keenly felt by the linguist (for an overview see Blevins 1995) than of practical difficulties on the part of the native speaker. In fact, most speakers of most languages have clear intuitions about the syllables in their language, know exactly where one ends and the next one begins, and can, with little or no formal training, draw up an inventory of syllables. It is precisely the confluence of these practical properties that makes the syllable such a natural building block for a script, and when new scripts are invented, such as Chief Sequoyah of the Cherokee [CER] did in 1819, these are often syllabaries. The ease with which native speakers manipulate syllables is all the more remarkable given the near impossibility of detecting syllable boundaries algorithmically in the acoustic data (see Chapter 9).

4.1.2 Moras

Syllables often come in two, and sometimes in three sizes: *light,* syllables are said to contain one mora, *heavy* syllables contain two, and *superheavy* syllables contain three. However, this containment is like a dime containing two nickels: true for the purposes of exchanging equal value, but not in the sense that the nickels could be found inside the dime. To see this, compare the typical light syllable, which has a consonant followed by a short vowel, to the typical heavy syllable, which will have the vowel long or followed by another consonant. If the second mora were contributed by the vowel length or by the coda consonant, we would expect syllables with long vowels and coda to be superheavy, but in fact only long vowels followed by long consonants (or clusters) generally end up trimoraic.

The exchange of equal value is best seen in the operation of various rules of stress and tone assignment. A familiar example involving stress is classical Latin, where the last syllable is extrametrical (ignored by the rule of stress placement) and stress always falls on the penultimate mora before it. For tone, consider the Kikuria [KUJ] example discussed in Odden (1995). In Bantu languages tense/aspect is often marked by assigning high tone to a given position in the verb stem: in Kikuria the high tone's falling on the first, second, third, or fourth mora signifies remote past, recent past, subjunctive, and perfective respectively. To quote Odden,

> Regardless of how one counts, what is counted are vowel moras, not segments not syllables. Stated in terms of mora count, high tone is simply assigned to the fourth mora in the perfective, but there is no consistent locus of tone assignment if one counts either syllables or segments.

An entirely remarkable aspect of the situation is that in some other languages, such as Lardil [LBZ] (see Wilkinson 1988), the whole phenomenon of rules being sensitive to the number of moras is absent: it is the number of syllables, as opposed to the number of moras, that matters. The distinction is known in linguistics as *quantity sensitive* vs. *quantity insensitive* languages, and for the most part it neatly divides languages in two typological bins. But there are nagging problems, such as Spanish, where the verbal system is quantity insensitive, but the nominal system is quantity sensitive. Moras constitute a very active area of research, see e.g. Broselow (1995), Hayes (1995).

4.1.3 Feet

One step up from syllables we find *metrical feet*, groupings that contains one strong and one weak syllable. Such feet account nicely for the long observed phenomenon that syllable stress generally appears as a pulse train, with stressed and unstressed syllables alternating quite predictably. When two stressed syllables meet, one of them generally get destressed: compare Italian *cittá* 'city' and *vécchia* 'old' to the combination *citta vécchia* 'old city' (Stress Retraction, see Nespor and Vogel 1989 – for a symmetrical rule see Exercise 3.4.1. above.) Another way of resolving such *clashes* is by the insertion of unstressed material such as a pause (Selkirk, 1984).

Other feet constructions, such as *unbounded feet,* a flat structure incorporating an arbitrary number of syllables, *degenerate feet,* containing just one syllable, and even *ternary feet.* But there is no dount that in the vast majority of cases, feet are binary (contain exactly two syllables), Kiribati [GLB] being the best, perhaps the only, counterexample that resists reanalysis in terms of binary feet (Blevins and Harrison, 1999).

4.1.4 Words and stress typology

At the top of the hierarchy we find the *prosodic word,* composed of feet, and carrying exactly one primary stress. Locating the syllable with the main stress, as well as those syllables that carry lesser (secondary, tertiary etc) stress is a primary concern of the *metrical theory* of phonology.[1] When the problem is solvable, the location of the primary stress is rule-governed, linguists speak of *fixed* stress. When the location of stress can't be predicted either on the basis of the phonological composition of the word (e.g. number and weight of syllables), nor on the basis of morphological composition, linguists speak of *free* stress, and make recourse to the purely descriptive method of marking in the lexicon where the stress should fall.

While this last recourse may strike the mathematician as pathetically inept, bordering on the ridiculous, it is nothing to be sneered at. First, languages provide many examples where a feature is genuinely unpredictable: as any foreign learner of German will know from bitter experience, the gender of German nouns *must be* memorized, as any heuristic appeal to "natural gender" will leave a large number of exceptions in its wake. Second, the procedure is completely legitimate even in cases where rules are available: tabulating a finite function can define it more compactly than a very complex formula would. The goal is to minimize the information that needs to be tabulated: we will begin to develop the tools to address this issue in Chapter 7. The reader who feels invincible should try to to tackle the following

Exercise* 4.1.1 Develop rules describing the placement of accent in Sanskrit.

For the purposes of the typology, the interesting cases are the ones where stress is fixed: for example, in Hungarian primary stress is always on the first syllable; in French it is on the last syllable; in Araucanian [ARU] it is on the second syllable; in Warao [WBA] it is on the next to last (penultimate) syllable; and in Macedonian [MKJ] it is on the antepenultimate syllable. Interestingly, no example is known where stress would always fall on the third syllable from the left, or the fourth from the right.

[1]Though in many ways related to the generative theory of *metrics,* metrical theory is an endeavor with a completely different focus: metrics is a branch of *poetics*, concerned with poetic meter (see e.g. Halle and Keyser 1971), while metrical theory is a branch of linguistics proper.

Quantity-sensitive languages offer a much larger variety: for example, in Eastern Cheremis [MAL] stress falls on the rightmost heavy syllable, but if all syllables are light, stress is on the leftmost syllable (Sebeok and Ingemann 1961). In Cairene Arabic [ARZ], stress is on the final syllable if it is superheavy, or on the penultimate syllable if heavy, otherwise on the rightmost non-final odd-numbered light syllable counting from the nearest preceding heavy syllable or from the beginning of the word (McCarthy, 1979). The reader interested in the full variety of possible stress rules should consult the StressTyp database (R.W.N. Goedemans and Visch, 1996).

The modern theory of stress typology begins with Hayes (1980), who tried to account for the observed variety of stress patterns in terms of a few simple operations, such as building binary trees over the string of syllables left to right or right to left – for a current proposal along the same lines see Hayes (1995). Here we will discuss another theory, which, in spite of its narrower scope (it applies to quantity-insensitive systems only) and sometimes clearly mistaken predictions, has taken the important step of divesting linguistic typology from much of its post hoc character. Though its originators, Goldsmith and Larson (1990), call their model dynamic, and use a spreading activation metaphor to describe its workings, it can be construed as static, involving nothing more than the solution of a system of linear equations.

Suppose we have n syllables arranged as a sequence of nodes, each characterized at time t by two real parameters, its current activation level $a_k(T)$ and the bias b_k that is applied to it independent of time T. The two parameters of the network are the leftward and rightward feeding factors α and β. The model is updated in discrete time: for $1 < k < n$ we have $a_k(T+1) = \alpha a_{k+1}(T) + \beta a_{k-1}(T) + b_k$. At the edges we have $a_1(T+1) = \alpha a_2(T) + b_1$ and $a_n(T+1) = \beta a_{n-1}(T) + b_n$. Denoting the matrix that has ones directly above (below) the diagonal and zeros elsewhere by $\mathbf{U}$ ($\mathbf{L}$) and collecting the activation levels in a vector $\mathbf{a}$, the biases in $\mathbf{b}$, we thus have

$$\mathbf{a}(T+1) = (\alpha\mathbf{U} + \beta\mathbf{L})\mathbf{a}(T) + \mathbf{b} \tag{4.1}$$

4.2 Affixation

4.3 Optimality

Chapter 5

Syntax

Chapter 6

Semantics

Chapter 7

Complexity

Grammars are imperfect models of linguistic behavior. To the extent we are more interested in competence than in performance (see 3.1), this is actually desirable, but more typically discrepancies between the predictions of the model and the observables represent serious over- or undergeneration (see 2.2). There is, moreover, an important range of models and phenomena where it is not quite obvious which of the above cases obtain. Suppose the task is to predict the rest of the series $2, 3, 5, \ldots$ A number of attractive hypotheses present themselves: the prime numbers, the Fibonacci numbers, square-free numbers, the sequence $2, 3, 5, 2, 3, 5, 2, 3, 5, \ldots$ and so on. The empirically-minded reader may object that the situation will be greatly simplified if we obtain a few more datapoints, but this is quite often impossible: the set of actual human languages can not be extended at will.

Therefore, it would be desirable to have an external measure of simplicity, so that we can select the best (most simple) hypothesis compatible with a given range of facts. Staring with Pāṇini, linguists tend to equate simplicity with shortness, and they have devoted a great deal of energy to devising notational conventions that will make the the linguistically attractive rules and generalizations compactly expressible. The central idea here is that such notational conventions get amortized over many rules, so in fact we can introduce them without seriously impacting the overall simplicity of the system.

In this chapter we begin to develop such a theory of simplicity based on the ideas of Kolmogorov complexity originally put forth in Solomonoff (1964). In 7.1 we introduce the basic notions of *information* and *entropy.* In 7.2 we

7.1 Information

In the classical model of information theory the *message* to be transmitted is completely devoid of internal structure: we are given a (finite or infinite) set $A = \{a_1, a_2, \ldots\}$ of elementary messages, and a probability distribution P over A. More complex messages are formed by concatenating elementary messages, and it is assumed that this is a Bernoulli experiment, so that the choice of the next elementary message is independent of what went on before. To transmit one of the a_i we need to encode it as a bitstring $C(a_i)$. We need to make sure that C is invertible, and that $C^+ : A^+ \rightarrow \{0,1\}^+$, defined by lifting C to be a (concatenation-preserving) homomorphism, will also be invertible. This is trivially satisfied as long as no codeword is a prefix of another codeword, i.e. the code is **prefix-free**, since in that case we know exactly where each codeword ends (and the next one begins) in the stream of bits transmitted.[1]

Exercise 7.1.1 Construct a code C such that C is not prefix-free but C^+ is nevertheless invertible.

An important subclass of prefix-free codes is fixed-length codes, where the length n_i of $C(a_i)$ is constant (usually a multiple of eight). However, in the cases that are of the greatest interest for the theory, the number of elementary messages is infinite, so no fixed length code will do. Prefix-free variable-length binary codes will satisfy the following

Theorem 7.1.1 Kraft Inequality. If c_i are codewords of length n_i such that $c_i\alpha = c_j$ implies $\alpha = \lambda$,

$$\sum_i 2^{-n_i} \leq 1 \tag{7.1}$$

Proof A prefix-free set of codewords can be depicted as a binary tree, where each sequence of zeros and ones corresponds to a unique path from the root to a leaf node, zero (one) meaning turn left (right). For the tree with two nodes, (7.1) is trivially satisfied (as equality). Since all prefix codes can be obtained by extending a leaf one or both ways, the result follows by induction for finite codes, and by standard limit arguments, for infinite codes as well.

Given a probability distribution P, *Shannon-Fano codes* are computed by an algorithm that constructs this tree top down, by dividing the total probability mass in two parts recursively, until each node has only one elementary message. Shannon-Fano codes are of historical/theoretical interest only: in practical applications the number of elementary messages is finite, and the more efficient *Huffman codes,* where the tree is created from the bottom up starting with the least probable message, have replaced Shannon-Fano codes entirely.

Exercise 7.1.2 Specify the top down algorithm in more detail, using raw (cumulative) probabilities. Compare your algorithms to the actual Fano (Shannon) algorithms. Specify the bottom-up procedure, and compare it to the Huffman algorithm. Do any of these procedures fit the notion of a *greedy* algorithm?

Now we introduce the quantity $H(P) = -\sum_i p \log_2 p$, known as the **entropy** of the distribution P. While the definition of H may at first blush look rather arbitrary, it is, up to a constant multiplier (which can be absorbed in the base of the logarithm chosen) uniquely defined as *the* function that enjoys some simple

[1]Somewhat confusingly, codes enjoying the prefix-free property are also called *prefix codes.*

and natural properties we expect any reasonably numerical characterization of the intuitive notion of *information* to have.

Theorem 7.1.2 Uniqueness of Entropy. We investigate nonnegative, continuous, symmetrical functions $I(p_1, p_2, \dots, p_k)$ defined for discrete probability distributions $P = \{p_1, p_2, \dots, p_k\} (\sum p_i = 1)$. (i) If P' is formed from P by adding another outcome a_{k+1} with probability $p_{k+1} = 0$, we require $I(P) = I(P')$ (ii) We require I to take maximum value in the equiprobable case $p_i = 1/k$. (iii) For P, Q independent we require $I(PQ) = I(P) + I(Q)$ and in general we require $I(PQ) = I(P) + I(Q|P)$. The only functions I satisfying the above conditions are $-c \sum p_i \log(p_i) = cH$ for arbitrary nonnegative constant c.

Proof Let us denote the maximum value $I(P_k)$ taken in the equiprobable case $P_k = \{1/k, 1/k, \dots, 1/k\}$ by $l(k)$. By (i) we have $l(k) \leq l(k+1)$. By property (iii), taking r independent distributions P_k we have $l(k^r) = l(P_k^r) = l(P_{k^r}) = rl(k)$. Letting $k = e^z$ and $l(e^x) = B(x)$ this functional equation becomes $B(zr) = rB(z)$ which, by a well-known theorem of Cauchy, can only be satisfied by $B(z) = cz$ for some constant c, so $l(e^z) = cz$, and thus $l(k) = c \log(k)$ (and by the monotone growth of l established earlier, $c > 0$). Turning to the non-equiprobable case given by rational probabilities $p_i = n_i/n$, we define Q to contain n different events, divided into k groups of size n_i, such that the conditional probability of an event in the ith group is $1/n_i$ if a_i was observed in P, zero otherwise. This way, we can refer to the equiprobable case for which I is already known, and compute $I(Q|P) = c \sum_{i=1}^{k} p_i \log(n_i) = c \sum_{i=1}^{k} p_i \log(p_i) + c \log(n)$. In the joint distribution PQ, each event has the same probability $1/n$, so that $I(PQ) = c \log(n)$. Given our condition (iii), we established $I(P) = -c \sum_{i=1}^{k} p_i \log(p_i)$ for p_i rational, and thus by continuity for any p_i. That H indeed enjoys properties (i)-(iii) is easily seen: (i) is satisfied because we use the convention $0 \log(0) = 0$ (justified by the limit properties of $x \log(x)$), (ii) follows by Jensen's Inequality, and finally (iii), and the more general chain rule $H(P_1 P_2 \dots P_k) = H(P_1) + H(P_2|P_1) + H(P_3|P_1 P_2) + \dots + H(P_k|P_1 P_2 \dots P_{k-1})$ follows by simply rearranging the terms in the definition.

Exercise 7.1.3 Obtain a sample of English and count the frequency of each character, including whitespace. What is the *grapheme entropy* of the sample? How much is the result changed by using e.g. Perl or C program texts rather than ordinary (newspaper) text in the sample?

Exercise 7.1.4 Write a program that parses English into syllables, and count each syllable type in a larger sample. What is the *syllable entropy* of the text?

While it is easiest to consider phonemes, graphemes, syllables, and other finite sets of elementary messages, the definition of entropy is equally meaningful for infinite sets, and in fact extends naturally to continuous distributions with density $f(x)$ by taking $H(f(x)) = -\int_{-\infty}^{\infty} f(x) \log_2(f(x)) dx$. Here we will consider English as being composed of words as elementary messages and estimate its *word entropy*. The task is made somewhat harder by the fact that words, being generated by productive morphological processes such as compounding (see Chapter 4), form an infinite set. Though the probabilities of *the, of, to, a,* *and, in, for, that* and other frequent words can be estimated quite reliably from counting them in samples (corpora) of medium size, say a few million words, it is clear that no finite sample will provide a reliable estimate for all the (infinitely many) words that we would need to cover.

Therefore we start with a well-known regularity of large corpora, **Zipf's Law,** which states that the rth word in the corpus will have relative frequency proportional to $1/r^B$, where B, the *Zipf constant,* is a fixed number slightly above 1. To establish the constant of proportionality C_k, note that a corpus of size N will have about $N^{1/B}$ different words. Let us denote the cumulative probability of the most frequent k words by P_k and assume Zipf's Law holds in the tail, so that we have

$$1 - P_k = C_k \sum_{r=k+1}^{N^{1/B}} r^{-B} \approx C_k \int_k^{N^{1/B}} x^{-B} dx = \frac{C_k}{(1-B)}[N^{\frac{1-B}{B}} - k^{1-B}] \quad (7.2)$$

For large N the first bracketed term can be neglected, and therefore we obtain $C_k \approx (1-P_k)(B-1)/k^{B-1}$. The first k words, for relatively small fixed k, already cover a significant part of the corpus: for example, the standard list in Vol 3 of (Knuth, 1971) contains 31 words said to cover 36% of English text, the 130-150 most frequent collected in Unix `eign` covers approximately 40% of newspaper text, and to reach 50% coverage we need less than 256 words. To estimate the entropy we take

$$H = -\sum_{r=1}^{k} p_r \log_2(p_r) - \sum_{r=k+1}^{N^{1/B}} p_r \log_2(p_r) \quad (7.3)$$

The first sum, denoted H_k, can be reliably estimated from frequency counts, and of course can never exceed the maximum (equiprobable) value of $\log_2(k)$. In the second sum, we first use the Zipf estimate only for $\log_2(p_r) \approx \log_2(C_k) - B\log_2(r)$. Since the first term is independent of r, we have $-\log_2(C_k)\sum_{r=k+1}^{N^{1/B}} p_r$ and of course the sum of these probabilities is $1 - P_k$. Finally, the remaining $C_k B \sum_{r=k+1}^{N^{1/B}} \log_2(r) r^{-B}$ can again be approximated by the integral

$$C_k B \int_{k+1}^{N^{1/B}} \log_2(x) x^{-B} dx \quad (7.4)$$

and again the value at the upper limit can be neglected for large N, so we get

$$C_k B \frac{(B-1)\log_2(k) + 1}{(B-1)^2 k^{B-1}} \quad (7.5)$$

Collecting the terms and substituting our estimate of C_k we obtain

$$H \approx H_k + \frac{1-P_k}{k^{2B-2}}(B\log_2(k) - (1-P_k)(B-1)k^{B-1} + \frac{B}{B-1}) \quad (7.6)$$

H_{256} can be estimated from medium or larger corpora to be about 3.9 bits. P_{256} is about 0.52, and B for English is about 1.05, so the estimate yields 12.67 bits, quite close to the $\sim$ 12 bits that can be directly estimated based on large corpora (over a billion words). In other languages, the critical parameters may take different values: for example, in Hungarian it requires the first 4096 words to cover about 50% of the data, and the entropy contributed by H_{4096} is closer to 4.3 bits, so we obtain $H \leq 11.6$ bits. (7.6) is not very sensitive to the choice of k, but is very sensitive to B. Fortunately, on larger corpora B is better separated from 1 than Zipf originally thought. To quote Mandelbrot (1961:196):

> Zipf's values for B are grossly underestimated, as compared with values obtained when the first few most frequent words are disregarded. As a result, Zipf finds that the observed values of B are close to 1 or even less than 1, while we find that the values of B are not less than 1.

As we shall see in the following Theorem, for prefix codes the entropy appears as a sharp lower bound on the expected code length: no code can provide better "on the wire" compression (smaller average number of bits). Our estimate therefore means that English words require about 12 bits on average to transmit or to store – this compares very favorably to using 7-bit ascii which would require about 35 bits (frequency weighted average word length in English is about 5 characters).

Theorem 7.1.3 (Shannon 1948) Let a_i be arbitrary messages with probability p_i and encoding $C(a_i) = c_i$ of length n_i such that $\sum_i p_i = 1$ and the set of codewords is prefix-free,

$$L(P) = \sum_i p_i n_i \geq H(P) \tag{7.7}$$

with equality iff the codewords are all exactly of length $\log_2(1/p_i)$.

Proof

$$H(P) - L(p) = \sum_i p_i \log_2 \frac{2^{-n_i}}{p_i} = \log_2 e \sum_i p_i \ln \frac{2^{-n_i}}{p_i} \tag{7.8}$$

The right-hand side can be bound from above using $\ln x \leq x - 1$, and by the Kraft Inequality we get

$$H(P) - L(p) \leq \log_2 e \sum_i p_i (\frac{2^{-n_i}}{p_i} - 1) \leq 0 \tag{7.9}$$

When the inverse probabilities are very far from powers of two, direct encoding may entail considerable loss compared to the entropy ideal. For example, if $p_1 = .9, p_2 = .1$ the entropy is 0.469 bits, while encoding the two cases as 0 vs 1 would require a full bit. In such cases, it may make sense to consider blocks of messages. For example, three Bernoulli trials would lead 111 with probability .729; 110, 101, or 011 with probability .081; 001, 010, or 100 with probability .009; and 000 with probability .001. By using 0 for the most frequent case, 110, 100, and 101 for the next three, and finally 11100, 11101, 11110, and 11111 for the remaining four, we can encode the average block of three messages in 1.598 bits, so the average elementary message will only require $1.598/3 = 0.533$ bits.

Exercise 7.1.5 Prove that no block coding scheme can go below the entropy limit, but that with sufficiently large block size average code length can approximate the entropy within any $\epsilon > 0$.

Exercise 7.1.6 Prove that a regular language is prefix-free iff it is accepted by a DFSA with no transitions out of accepting states. Is a prefix-free language context-free iff it is accepted by a DPDA with the same restriction on its control?

In real-life communication, prefix-free codes are less important than the foregoing theorems would suggest, not because real channels are inherently noisy (the standard error-correcting techniques would be just as applicable), but because of the peculiar notion of *synchrony* that they assume. On the one hand, prefix-freeness eliminates the need for transmitting an explicit concatenation symbol, but on the other, it makes no provision for `BEGIN` or `END` symbols: the only way the channel can operate is is by keeping the sender and the receiver in perfect synchrony.

Exercise 7.1.7 Research the role of the ascii codes 0x02 (STX), 0x03 (ETX), and 0x16 (SYN).

Variable-length codes (typicaly, Huffman encoding) therefore tends to be utilized only as a subsidiary encoding, internal to some larger coding scheme which has the resources for synchronization. We will discuss an example, the G3 standard of fax transmission, in Chapter 10.

Exercise 7.1.8 Take the elementary messages to be integers i drawn from the geometrical distribution ($p_i = 1/2^i$). We define a complex message as a sequence of n such integers, and assume that n itself is geometrically distributed. How many bits will the average complex message require if you restrict yourself to prefix-free codes (no blocking)? With blocking, what is the optimal block size? What is the optimum average message length if the restriction on prefix-freeness is removed?

Human language, when viewed as a sequence of phonemes, shows very strong evidence of *phonotactic regularities* i.e. dependence between the elementary messages. If we chose syllables as elementary, the dependence is weaker, but still considerable. If we use morphemes as our elementary concatenative units, the dependence is very strong, and if we use words, it is again weaker, but far from negligible. Besides its uses in the study of coding and compression, the importance of entropy comes from the fact that it enables us to quantify such dependencies. For independent variables we have $H(PQ) = H(P) + H(Q)$ (see condition (iii) in Theorem 7.1.2 above), so we define the **mutual information** between P and Q as $H(P)+H(Q)-H(PQ)$. Mutual information will always be a nonnegative quantitive, equal to zero iff the variables P and Q are independent.

All forms of communication that are parasitic on spoken language, such as writing, or exercise the same fundamental cognitive capabilities, such as sign language, are strongly non-Bernoullian. Other significant sources of messages, such as music or pictures, also tend to exhibit a high degree of temporal/spatial redundancy, as we shall discuss in Chapters 8-10.

7.2 Kolmogorov complexity

The model described above is well suited only for the transmission of elementary messages that are truly independent of one another. If this assumption fails, redundancy between successive symbols can be squeezed out to obtain further compression. Consider, for example, the sequence of bits 010100101011... that is obtained by taking the fractional part of $n\sqrt{2}$ and emitting 1 if greater than .5, 0 otherwise. It follows from Weyl's theorem of equidistribution that P(0)=P(1)=.5. The entropy will be exactly 1 bit, suggesting that the best we could do was to transmit the sequence bit by bit: transmitting the first n elementary messages would require n bits. But this is clearly *not* the best that we can do: to generate the message at the other side of the channel requires only the transmission of the basic algorithm, which takes a constant number of bits, plus the fact that it needs to be run n times, which takes $\log_2 n$ bits.

We have not, of course, transcended the Shannon limit, but simply put it in sharp relief that Shannon entropy limits compressability *relative* to a particular method of transmission, namely prefix-free codes. The faster method of transmitting 010100101011... requires a great deal of shared knowledge between sender and recipient: they both need to know what $\sqrt{}$ is and how you compute it, they need to agree on `for`-loops, `if`-statements, a compare operator, and so on. Kolmogorov complexity is based on the idea that all this shared background boils down to the knowledge required to program Turing machines in general, or just one particular (universal) Turing machine.

Turing's original machines were largely hardwired, with what we would nowadays call the program burned into the finite state control (firmware) of the machine. For our purposes, it will be more convenient to think of Turing machines as universal machines that can be programmed by finite binary strings. For convenience, we will retain the requirement of prefix-freenes as it applies to such programs: we say that a partial recursive function $F(p, x)$ is **self-delimiting** if for any prefix q of the program p $F(q, x)$ is undefined. This way, a sequence of programs can be transmitted as a concatenation of program strings. The second variable of F, which we think of as the input to the machine programmed by p, is a string of natural numbers, or rather, a single (e.g. Gödel) number that is used to encode strings of natural numbers.

3 **Definition 7.2.1** The **conditional complexity** $H_F(x|y)$ of x given y is the length of the smallest program p such that $x = F(p, y)$, or ∞ if no such program exists.

To remove the conditional aspects of the definition, we will need two steps, one entirely trivial, substituting $y = 0$, and one very much in need of justification, replacing all Fs by a single universal Turing machine U.

Definition 7.2.2 The **complexity** $H_F(x)$ of x relative to F is the length of the smallest program p such that $x = F(p, \lambda)$, or ∞ if no such program exists.

Theorem 7.2.1 (Solomonoff 1960, Kolmogorov 1965) There is a partially recursive function $U(p, y)$ such that for any partially recursive $F(p, y)$ there exists a constant C_F satisfying

$$H_U(x|y) \leq H_F(x|y) + C_F \tag{7.10}$$

Proof We construct U by means of a universal Turing machnine $V(a, p, x)$ which can be programmed by the appropriate choice of a to emulate any $F(p, x)$. To force the prefix-free property, for any string $d = d_1 d_2 \dots d_n$ we form $d^0 = d_1 0 d_2 0 \dots 0 d_n 1$ by inserting 0s as concatenation markers and 1 as an end marker. Since any binary string p can be uniquely decomposed as an initial segment a^0 and a trailer b, we can define $U(p, x)$ by $V(a, b, x)$. In particular, for $F(p, x)$ there is some f such that for all p, x $F(p, x) = V(f, p, x)$. In case x can be computed from y by some shortest program p running on F, we have $U(f^0 p, y) = V(f, p, y) = F(p, y) = x$, so that $H_U(x, y) \leq 2|f| + H_F(x|y)$.

There are many ways to enumerate the partial recursive functions, and many choices of V (and therefore U) could be made. What Theorem 7.2.2 means is that the choice between any two will only affect $H_U(x|y)$ up to an additive constant, and thus we can suppress U and write simply $H(x|y)$, keeping in mind that it is defined only up to a $O(1)$ term. The claim is often made (see e.g. Chaitin 1982) that $H(x)$ measures the complexity of an individual object x, as opposed to entropy, which very much presumed that the objects of study are drawn from a probability distribution. However, this claim is somewhat misleading, since the focus of the theory is really the asymptotic complexity of a sequence of objects, such as initial substrings of some infinite string, where the $O(1)$ term can be really and truly neglected. For a single object, one could always find an U that will make $K_U(x)$ zero, just as if our only interest was in compressing a singly file, we could compress it down to 1 bit, with an uncompress function that prints out the object in question if the bit was set, and does nothing otherwise.

To take advantage of asymptotic methods, one typically needs to endow familiar unordered objects with some kind of order. For example, formal languages are inherently unordered, but it is no great stretch to order them lexicographically. Once this is done, we can talk about the n-th string, and replace sets by their characteristic function written as a (possibly infinite) bitstring whose n-th bit is 1 or 0 depending on whether the n-th string y_n enjoyed some property or not. In order to discuss regular languages, we need to capture the structure of the state machine in bitstrings. Given some language $L \subset \Sigma^*$ and any string x we define $\chi = \chi_1 \chi_2 \dots$ to be 1 on the nth bit χ_n iff the string xy_i is in L. Clearly, two strings x and x' have the same χ iff they are right congruent, so that different χ correspond to different states in the DFSM. For L regular, the first n bits of any χ can be transmitted by transmitting the DFSA in O(1) bits, transmitting which state (out of finitely many) the χ in question corresponds to (again O(1) bits), and transmitting the value of n, which requires no more than $\log_2(n)$ bits.

Theorem 7.2.2 (Li and Vitanyi 1995) $L \subset \Sigma^*$ is regular iff there exists a constant C_L depending on L but not on n such that for all $x \in \Sigma^* K(\chi : n) \leq log_2(n) + C_L$.

Proof We have already seen the only if part – to prove the converse we need to establish that for any constant C_L there will be only finitely many *infinitely long* bitstrings that have no more than $\log_2(n) + C_L$ asymptotic complexity. Once this is demonstrated, the rest follows by the usual construction of FSA from right congruence classes.

Add

Ming Li and Paul Vitanyi (1995) A New Approach to Formal Language Theory by Kolmogorov Complexity. *SIAM Journal on Computing* Volume 24, Number 2 pp. 398-410

Chapter 8

Linguistic pattern recognition

In general, the pattern recognition task is defined as one where an infinite, continuous set of inputs is associated with a finite variety of outputs. A typical example is *face recognition*, where the goal is to identify the face as belonging to the same person in spite of changes in viewing angle, distance, light, makeup and hairdo, facial expression, etc. We speak of *linguistic* pattern recognition when the set of outputs is structured linguistically. This means both that in the output units of linguistic significance follow each other in discrete time, (e.g. a temporal succession of letters, words, or sentences) and that these units themselves come from a finite (or finitely generated) set. We could stretch the definition to include data that lacks temporal organization: for example, the recognition of isolated characters is considered by many to be a linguistic pattern recognition task, especially in the case of Han and Hangul characters, which can be decomposed spatially though not necessarily temporally (see Chapter 10). However, no amount of stretching the definition will allow for face or fingerprint recognition, as the output in these domains can be made finite only by imposing some artificial cutoff or limitation on the system.

Both linguistic and nonlinguistic pattern recognition involve an early stage of signal processing, often referred to as the *front end*. In speech recognition, the front end is generally based on acoustic principles, while in optical character recognition image processing techniques are used. In Chapter 7 we looked at codes that were fully invertible: here we will distinguish *low-* and *high-level* signal processing based on whether the input signal is largely recoverable from the output or not. In low-level signal processing, we encode the input in a largely invertible (lossless) manner, while in high-level signal processing, so much of the information that was present in the input gets discarded, that the output can no longer serve as a basis for reconstructing the input. It should be clear from the foregoing that the distinction between low- and high-level signal processing is largely a matter of degree, especially as many signal processing algorithms have tuneable parameters that control the degree of information loss. Nevertheless, there are some general guidelines that can, with proper care, be used to distinguish the two kinds of signal processing in nearly all cases of practical interest.

On one side, any transformation that limits losses to such a degree that they are below the threshold of human perception is definitely low-level. A typical example would be the algorithm used in "ripping" music CDs: here the input is 2*16 bits sampled at 44.1 kHz, for a total of 1411 kbps, and the output is MP3, requiring only 128 kbps (stereo) or 64kbps (mono). On the other side, any *feature extraction* algorithm that produces categorial (especially 0-1) output from continuous input is considered high-level. A typical example would be an algorithm that decides whether a stretch of speech is voiced or unvoiced (see Chapter 9). Here the key issue is not whether the output is categorial, but rather the number of categories considered. Two categories (one bit) is definitely high-level, and 64k categories (16 bits) are typical of low-level signal processing. We discuss this issue, *quantization,* in 8.1.

Because of their special importance in linguistic pattern recognition, we introduce the basic notions of Markov processes (chains) and Hidden Markov Models (HMMs) in 8.2 both for the discrete (quantized) and the continuous case.

Another criterion for distinguishing low- and high-level processing is whether the output of the front end is still suitable for human pattern recognition. For example, agressive signal processing techniques can reduce the number of bits per second required for transmitting speech signals from 64kbps (MP3 mono) to 2kbps (MELP) with considerable degradation in subjective signal quality, but little or no loss of intelligibility. Surprisingly, there exist pattern recognition methods and techniques that work best when the signal is degraded beyond the point of human recognizability: we discuss such cases, and the reasons for their existence, in 8.3.

In 8.4 we turn to the issue of *multimodal* recognition, when the signal arrives in separate channels, e.g. one for video (lip reading) and one for audio. Finally, in 8.5 we discuss *topic detection* and the general problem of classifying documents.

8.1 Quantization

Historically, quantization techniques grew out of analog/digital signal conversion, and to some extent they still carry the historical baggage associated with considerations of keeping the A/D circuitry simple. We begin with air pressure $s(t)$ viewed as a continuous function of time, and we assume the existence of an amplitude bound A such that $-A \leq s(t) \leq A$ for all t considered.

Exercise 8.1.1 Research the empirical amplitude distribution of speech over long stretches of time. If this has variance σ, how do you need to set k in $A = k\sigma$ such that less than one in a hundred samples will have $|s| > A$? How do you need to set k so that on the average no more than one in a thousand samples gets clipped?

Our goal is to quantize both the time axis and the amplitude axis, so that $s(t)$ is replaced by a stepwise constant function $r(t_i)$ where r can take on only discrete values $r_0, r_1, \ldots, r_N$. In the simplest case, both the r_i and the t_j are uniformly spaced. The reciprocal of $t_{i+1} - t_i$ is called the **sampling frequency** – in the case of speech it is typically in the 6-40 kilohertz range. For simplicity, N is usually chosen as a power of 2, so that instead of N-way sampling we speak of b-bit sampling, $N = 2^b$. In the electrical engineering literature, this is known as **pulse code modulation** or PCM. In the case of speech, b is typically between 1 and 24: at high sampling rates (20 kHz or above), one bit is already sufficient for low quality, but generally intelligible, speech coding.

Uniform spacing means that the quantization levels are the centers of the 2^b intervals of size $\Delta = 2A/2^b$. The squared error of the stepwise approximation is called the *quantization noise*, and we can estimate it for any elementary interval of time by taking b sufficiently large so that the error becomes a random variable with zero mean that is uniformly distributed over the range $-\Delta/2, \Delta/2$. Since the squared error of such a variable is obviously $\Delta^2/12$, the more commonly used **Signal to Quantization Noise Ratio** or SQNR is $3P_x 2^{2b}/A^2$ where P_x is the **signal energy,** defined as the area under the curve $s^2(t)$. Typically, speech engineers express such energy ratios using the **decibel scale,** 10 times their base 2 logarithm, so we obtain the conclusion that adding an extra bit improves SQNR by about 6.02 decibels for PCM.

There is, of course, no reason to assume that the uniform quantization scheme is in any way optimal. There are many methods to decrease the number of bits per second required to transmit the signal or, equivalenty, to increase signal quality for the same number of bits, and here we provide only a thumbnail schetch. In one family of methods, the original signal gets warped by the application of a nonlinear function W so that we uniformly quantize $W(s(t))$ instead of $s(t)$. If W is chosen linear for $|s| < A/87.56$ and logarithmic for larger s, we speak of **A-law** PCM, commonly used in digital telephony in Europe since the seventies. If $W = \text{sign}(s) A \log(1 + 255|s|/A)/3$, we speak of **$\mu$-law** PCM, commonly used for digital telephony in the US and Japan. These methods, predating MP3 by over three decades, provide high "toll" quality speech at the same 64kbps rate.

Exercise 8.1.2 What must be the amplitude distribution of speech over long stretches of time for A-law (μ-law) to provide the optimal warping function? Compare these to the result of 8.1.1.

Beyond A-law or μ-law, in practical terms very little can be gained by matching the warping function more closely to the long-term characteristics of speech. The *adaptive* family of methods exploits redundancies in the short-term characteristics of speech – these will be discussed in more detail in Chapter 9.

8.2 Markov processes, Hidden Markov Models

Here we discuss the simple case when the message to be coded exhibits 1st order Markovian dependence. Consider a finite set of elementary messages (symbols) $a_i (1 \leq i \leq k)$ with probabilities p_i. If in all complex messages $P(a_{i_n}|a_{i_1}a_{i_2}\ldots a_{i_{n-1}}) = P(a_{i_n}|a_{i_{n-1}})$ holds, i.e. a_{i_n} is predictable on the basis of the immediately preceeding symbol just as well as it is predictable on the basis of all preceeding symbols, we say that the messages are generated by a **first order Markov process** with transition probabilities $t_{ij} = P(a_j|a_i)$.

In general, a **signal process** is an infinite sequence of random variables $X_t, t = 1, 2, 3, \ldots$ whose values are the elementary messages collected in a set A. For convenience, two-way infinite sequences including variables for $X_0, X_{-1}, X_{-2}, \ldots$ are often used, since this makes the **shift** operator S that assigns to every sequence of values $a(t_i)$ the sequence $a(t_{i-1})$ invertible. A process is called *stationary* if the shift operator (and therefore every positive or negative power of it) is measure-preserving.

Definition 8.2.1 A stochastic process is a probability measure μ defined on $A^{\mathbb{Z}}$. If for every measurable subset U we have $\mu(S(U)) = \mu(U)$ the process is **stationary**. If for every function f of n variables $f(X_i, X_{i+1}, \ldots, X_{i+n-1})$ converges with probability 1 to the expected value $E(f(X_1, X_2, \ldots X_n)$ whenever the latter is finite, the process is **ergodic**.

Exercise 8.2.1 Can a non-stationary process be ergodic? Can a non-ergodic process be stationary?

In general, the **entropy of a signal process** is defined as

$$\lim_{n\to\infty} H(X_1, X_2, \ldots, X_N)/N$$

if this limit exists: in the case of word-unigram based models it is often referred to as the *per-word entropy* of the process. For the Bernoulli case studied in 7.1, the random variables X_i are independently identically distributed, so this definition reduces to $H(X_1) = H(X)$. In the non-Bernoulli case, by the chain rule we have $\frac{1}{N}(H(X_1)+H(X_2|X_1)+H(X_3|X_1X_2)+\ldots+H(X_N|X_1X_2\ldots X_{N-1}))$, and if the process is Markovian this reduces to $\frac{1}{N}(H(X_1)+H(X_2|X_1)+H(X_3|X_2)+\ldots+H(X_N|X_{N-1}))$. If the process is ergodic, all terms except the first one are $H(X_2|X_1)$, and these dominate the sum as $N \to \infty$. Therefore we obtain the result that

Theorem 8.2.1 The word entropy of a (stationary, ergodic) Markov process is $H(X_2|X_1)$.

To fully define one-sided first order Markov chains we only need a set of *initial probabilities* I_i and the transition probabilities t_{ij}. In two-sided chains, initial probabilities are replaced by the probabilities T_i that the chain is in state i: these obviously satisfy $T_j = \sum_{i=1}^{n} T_i t_{ij}$ and thus can be found as the eigenvector corresponding to the eigenvalue 1 (which will be dominant if all t_{ij} are strictly positive). By a classical theorem of A.A. Markov, if the process is **transitive** in the sense that every state can be reached from every state in finitely many steps (i.e. if the transition matrix is irreducible), the state occupancy probabilities T_i satisfy the law of large numbers:

Theorem 8.2.2 (Markov 1912) For any ϵ, δ arbitrarily small positive numbers there exists a length N such that if m_i denotes the absolute frequency of the process being in state a_i during a trial of length N, we have

$$P(|m_i/N - T_i| > \delta) < \epsilon \tag{8.1}$$

The current state of the chain can be identified with the last elementary message emitted, since future behavior of the chain can be predicted just as well on the basis of this knowledge as on the basis of knowing all its past history. From 8.2.1 the word entropy of such a chain can be computed easily: if $X_1 = a_i$ is given, $H(X_2)$ is $-\sum_j t_{ij} \log_2(t_{ij})$, so we have

$$H(X_2|X_1) = -\sum_i T_i \sum_j t_{ij} \log_2(t_{ij}) \tag{8.2}$$

What makes this entropy formula particularly important is that for longer sequences average log probability is concentrated on this value. By definition, the probability $P(a)$ of any sequence $a = a_{i_1} a_{i_2} \dots a_{i_N}$ of elementary messages is $T_{i_1} \prod_{k=1}^{N-1} t_{i_k i_{k+1}}$, and the probability of a set C containing messages of length N is simply the sum of the probabilities of the individual sequences.

Theorem 8.2.3 (Shannon 1948) For arbitrary small $\epsilon \geq 0, \eta \geq 0$ there is a set $C_{\eta,\epsilon}$ of messages of length N for sufficiently large N such that $P(a \notin C) < \epsilon$ and if $a \in C$ then $|\log_2(1/P(a))/N - H| < \eta$

Proof. Let us collect the t_{ij}: if m_{ij} counts the number of times t_{ij} occurred in the product $T_{i_1} \prod_{k=1}^{N-1} t_{i_k i_{k+1}}$ we have

$$P(a) = T_{i_1} \prod_{i,j} t_{ij}^{m_{ij}} \tag{8.3}$$

.

We define C as containing those and only those sequences a that have positive probability (include no $t_{ij} = 0$) and satisfy $|m_{ij} - NT_i t_{ij}| < N\epsilon$ for all i, j For these, the product in (8.3) can be rewritten with exponents $NT_i t_{ij} + N\epsilon\Theta_{i,j}$ with $|\Theta_{ij}| < 1$. Therefore,

$$\log_2(1/P(a)) = -\log_2(T_{i_1}) - N \sum_{t_{ij} \neq 0} T_i t_{ij} \log_2(t_{ij}) - N\epsilon \sum_{t_{ij} \neq 0} \Theta_{ij} \log_2(t_{ij}) \tag{8.4}$$

Since the second term is just $-NH$ we have

$$|\log_2(1/P(a))/N - H| < -\log_2(T_{i_1})/N + \epsilon \sum_{t_{ij} \neq 0} \log_2(t_{ij})$$

The first term tends to 0 as $N \to \infty$ and the second term can be made less than an arbitrary small η with the appropriate choce of ϵ. What remains to be seen is that sequences $a \notin C$ with nonzero probability have overall measure $< \epsilon$. For a nonzero probability a not to belong in C it is sufficient for $|m_{ij} - NT_i t_{ij}| \geq N\epsilon$ to hold for at least one i, j. Thus we need to calculate $\sum_{t_{ij} \neq 0} P(|t_{ij} - NT_i t_{ij}| \geq N\epsilon)$. Since this is a finite sum (maximum n^2 terms altogether) take any $t_{ij} \neq 0$ and apply Theorem 8.1.2 to find N large enough for $P(|m_i - NT_i| < N\delta/2) > 1 - \epsilon$ to hold. By restricting our attention to state a_i we have a pure Bernoulli experiment whose outcomes (moving to state a_j) satisfy the weak law of large numbers, and thus for any ϵ and $\delta/2$ we can make $P(|m_{ij}/m_i - t_{ij}| < \delta/2|) > 1 - \epsilon$. Combining these two we obtain

$$P(|m_i - NT_i| < N\delta/2)P(|m_{ij} - m_i t_{ij}| < m_i\delta/2) \geq (1-\epsilon)(1-\epsilon) \geq 1 - 2\epsilon$$

By the triangle inequality $P(|m_{ij} - NT_i t_{ij}| < N\delta) \geq 1 - 2\epsilon$ so $P(|m_{ij} - NT_i t_{ij}| \geq N\delta) < 2\epsilon$, and by summing over all i, j we obtain $P(\overline{C}) < 2n^2\epsilon$, which can be made as small as desired.

In **Hidden Markov Models** or HMMs the association of states and elementary messages is weakened by means of assigning an *output distribution* O_i to each state i. It is assumed that the O_i are independent of time, and independent of each other (though possibly identically or very similarly distributed). In the special case $O_i(a_i) = 1, O_i(a_j) = 0 (j \neq i)$ we regain Markov chains, but in the typical case the state can not be deterministically recovered from the elementary message but remains to some extent hidden, hence the name. Each state i of an HMM can be conceived as a signal process on its own right, with entropy $H(O_i) = H_i$. We do not require the O_i to be discrete, in fact *continuous density* HMMs play an important role in speech recognition, as we shall see in Chapter 9. Thought it would be possible to generalize the definition to situations where the underlying Markov chain is also replaced by a continuous process, this makes little sense for our purposes, since our goal is to identify the underlying states with linguistic units, which are, by their very nature, discrete (see 3.1).

The entropy of the whole process can be computed as a weighted mixture of the output entropies only if each state is final (diagonal transition matrix). In the general case, we have to resort to the original definition $\frac{1}{N}(H(X_1) + H(X_2|X_1) + H(X_3|X_1X_2) + \ldots + H(X_N|X_1X_2 \ldots X_{N-1}))$ and we see that a term like $H(X_3|X_1X_2)$ can no longer be equated to $H(X_3|X_2)$ since it is the previous *state* of the model not the previous *output* that contributes to the current state and thus indirectly to the current output. Introducing a Markov chain of random *state variables* S_i we have $P(X_i = a_j) = \sum_{k=1}^{n} S_i(k)O_k(a_j)$ (where the sum ranges over the states). By definition, word entropy will be the limit of

$$\frac{1}{N}H(X_1, \ldots, X_N) = \frac{1}{N}(H(S_1, \ldots, S_N) + H(X_1, \ldots, X_N|S_1, \ldots, S_N) \\ -H(S_1, \ldots, S_N|X_1, \ldots, X_N)) \quad (8.5)$$

and we already computed the first term as $H(S_2|S_1)$. The second term is $\frac{1}{N}NH(X|S) = H(X_1|S_1)$, which is generally easy to compute from the underlying Markov process and the output probability distributions, and it is only the last term, $\lim \frac{1}{N}H(S_1, \ldots, S_N|X_1, \ldots, X_N)$ that causes difficulties, inasmuch as computing the states from the outputs is a nontrivial task. Under certain circumstances (see Chapter 10) we may be satisfied with pointwise maximum likelihood estimates.

8.3 High-level signal processing

Historically, linguistic pattern recognition systems were heavily slanted towards symbol-manipulation techniques, with the critical pattern recognition step often entirely obscured by the high-level preprocessing techniques that are referred to as *feature detection.* To the extent we can decompose the atomic concatenative units as bundles of distinctive features (see 3.2), the simultaneous detection of all features amounts to recognizing the units themselves. In many settings, both linguistic and non-linguistic, this makes excellent sense, since the actual number of distinct units N is considerably larger than the number of binary features used for their feature decomposition (on the order of $\log_2(N)$). Further, some of the well-established features, such as *voicing* in speech and *position* in handwriting recognition are relatively easy to detect, which gave rise to high hopes that the detection of other features will prove just as unproblematic – in speech recognition, this is known as the Stevens-Blumstein (1981) program.

Here we begin with a drastically simplified example, that of recognizing the (printed) characters *c, d, e* and *f.* Only two of these, c and e, are positioned between the normal "n" lines of writing, the other two, d and f, have *ascenders* that extend above the normal top line. And only two, d and e, have loops that completely sorround a white area, c and f leave the plane as a single connected component. Therefore, to recognize these four characters it is sufficient to detect the features [$\pm$ ascender] and [$\pm$ loop], a seemingly trivial task.

Exercise 8.2.1 Consider the difficulties of extracting either geometrical features like position or topological features like connectedness from a greyscale image. Write a list for later comparison with the eventual set of problems that will be discussed in Chapter 10.

First we apply a low-level step of *pixelization,* dividing the line containing the string of characters into a sufficient number of squares, say 30 by 40 for the average character, so that a line with 80 characters is placed into a 2400 by 40 array so that the bottom of this array coincides with the baseline of the print, and the top coincides with the horizontal line drawn through the highest points of the ascenders.

Using 8 bit greylevel to describe the amount of black ink found in a pixel, we have devoted some 96 kilobytes to encode what is the visual representation of 20 bytes of information. To recover the two bits per character that we are actually interested in, we first need to *segment* the line image into 80 rougly 30 by 40 rectangles, so that each of these contains exactly one character. We do this by considering the width one columns of pixels one by one, adding up the grey values in each to form a *blackness profile* (ref). Those columns where the result is small (zero) are considered dividers, and those where the values are higher than a threshold are considered parts of characters. We obtain an alternating string of divider and character zones, and we consider the segmentation well established if we have the correct number of character zones (80) and these all have approximately the same width.

Once these preliminaries are out of the way, the ascender feature can be simply detected by looking at the top 5 rows of pixels in each *character bounding box:* if these are all white (the sum of greyness is below a low threshold) the character in question has no ascender, otherwise it does. Detecting loops is a more complex computation, since we need to find local minima of the greyness function, which involves computing the gradient and determining that the Hessian is positive definite. To compute the gradient it is actually useful to blur the image, e.g. by averaging greyness over larger neighborhoods (e.g. over a disk of 5-10 pixel radius). Otherwise, completely flat white and black regions would both give zero gradient, and the gradient values near the edges would be numerically unstable.

Visually, such transformations would make the image less legible, as would the ascender transform which deletes everything but the top five rows of pixels. What this simplified example shows is that the transformations that enhance automatic feature detection may be very different from the ones that enhance human perception – a conclusion that will hold true as long as we focus on the goal of pattern recognition without any attempt to mimic the human perception/recognition process.

8.4 The information content of the signal

Human speech is intrinsically multimodal: in the typical case, we don't just hear the speakers, but also see their mouth, hand gestures, etc. There is evidence that the visual cues will significantly interact with the audio cues: here we describe the celebrated *McGurk effect*. . ...

Therefore, the first order of business is to determine the relative contributions of the different chanels, a problem that is made very difficult by the fact that the linguistic signal is highly redundant. In telephony, a one second stretch is typically encoded in 8 kilobytes, while in imaging, a one square cm area takes about 56 kilobytes at the 600dpi resolution common to most printers and scanners. On the average, one second of speech will contain about 10-15 phonemes, each containing no more than 6 bits of information, so the redundancy is about a thousandfold. For the Latin alphabet, one square cm will contain anywhere from 3 handwritten characters, say 18 bits, to 60 small printed characters (45 bytes), so the redundancy factor is between 1200 and 24000. In fact, these estimates are on the conservative side, since they do not take into account the redundancy between adjacent phonemes or characters.

8.5 Document Classification

As the number of machine-readable documents grows, finding the ones relevant to a particular query becomes an increasingly important problem. In the ideal case, we would like to have a *question answering* algorithm that would provide the best answer, relative to any collection of documents D, for any (not necessarily well-formed English) query q such as *Who is the CEO of General Motors?* Since D may contain contradictory answers (some of which may have been true at different times, others just plain wrong) it is clear that in general this is not a solvable problem. Question answering, together with unrestricted *machine learning, machine translation, pattern recognition, commonsense reasoning* etc. belong in the informal class of *AI-complete* problems: a solution to any of them could be leveraged to solve all problems of Artificial Intelligence.

Exercise 8.4.1 Define the above problems more formally, and develop Karp-style reduction proofs to show their equivalence.

As Matijasevič (1981) emphasizes, the proof of the unsolvability of a problem is never the final point in our investigations, but rather the starting point for tackling more subtle problems. AI-complete problems are so important that finding less general but better solvable formulations is still an important practical goal. Instead of unrestricted machine translation, which would encompass the issue of translating into arbitrary systems of logical calculus, and thus would subsume the whole AI-complete problem of *knowledge representation,* we may restrict attention to translation between natural languages, or even to a given pair of natural languages. Instead of the full question answering problem, we may restrict attention to the more narrow issue of *information extraction:* given a document d, e.g. a news article, and a relation R, e.g *Is-CEO-Of,* find all pairs of values $\{(x, y)|x \in \mathit{Person},\ y \in \mathit{Company},\ (x, y) \in \mathit{Is\text{-}CEO\text{-}Of}\}$ supported by d. This is still rather ambitious, as it requires a more robust approach to parsing the document than we are currently capable of (see Chapter 5), but at least it does away with many thorny problems of knowledge representation and natural language semantics (see Chapter 6). Once the problem is restricted this way, there is no particular reason to believe that it is algorithmically unsolvable, and in fact practical algorithms that run linear in the size of d are available in the public domain (Kornai 1999), though these do not claim to extract all instances, just a large enough percentage to be useful.

Even with linear parsing techniques, syntactic preprocessing of a large collection of documents (such as the web, currently containing about 10^9 documents) remains impractical, and the problem is typically attacked by means of introducing a crude intermediate classification system T composed of a few hundred to a few thousand *topics.* We assume that T partitions D into largely disjoint subsets $D_t \subset D(t \in T)$ and that queries themselves can be classified for topic(s). The idea is that questions about e.g. current research in computer science are unlikely to be answered by documents discussing the economic conditions prevailing in 19th century Congo, and conversely, questions about slavery, colonial exploitation, or African history are unlikely to be answered by computer science research papers.

Therefore we have two closely related problems: in *query parsing* we try to determine the set of topics relevant to the query, and in *topic detection* we try to determine which topics are discussed in a document. In many practical systems, the two problems are conflated into one, by treating queries as (very short) documents on their own right. We thus have the problem of *document classification:* given some documents D, topics T and some sample of (d, t) pairs from the relation *Is-About* $\subset D \times T$, find the values of *Is-About* for new documents. Since the space of topics is not really structured linguistically (if anything, it is structured by some conceptual organization imposed on encyclopedic knowledge), strictly speaking this is not a linguistic pattern recognition problem, but we discuss it in some detail since the mathematical techniques used are quite relevant to mathematical linguistics as a whole. First, some terminology and notation.

We assume a finite set of words $w_1, w_2, \ldots, w_N$ arranged on order of decreasing frequency. N is generally in the range $10^5 - 10^6$ – for words not in this set we introduce a catchall *unknown word* w_0. By *general English* we mean a probability distribution G_E that assigns the appropriate frequencies to the w_i either in some large collection of topicless texts or in a corpus that is appropriately representative of all topics. By the (word unigram) probability model of a **topic** t we mean a probability distribution G_t that assigns the appropriate frequencies to the w_i in a large collection of documents about t. Given a collection C we call the number of documents that contain w the **document frequency** of the word, denoted $DF(w, C)$, and we call the total number of w tokens its **term frequency** in C, denoted $TF(w, C)$.

Within a given topic t, only a few dozen, or perhaps a few hundred, words are truly characteristic (have $G_t(w)$ that differs significantly from the background probability $G_E(w)$) and our first goal is to find these. To this end, we need to estimate G_E: the trivial method is to use the *uncorrected observed frequency* $p(w) = TF(w, C)/L(C)$ where $L(C)$ is the **length** of the corpus C (total number of word tokens in it). While this is obviously very attractive, the numerical values so obtained tend to be highly unstable. For example, the word *with* makes up about 4.44% of a 55m word sample of the *Wall Street Journal* but 5.00% of a 46m word sample of the *San Jose Mercury News.* For medium frequency words, the effect is even more marked: for example *uniform* appears 7.65 times per million word in the WSJ and 18.7 times per million in the Merc sample. And for low frequency words, the straightforward estimate very often comes out as 0, which tends to introduce singularities in models based on the estimates.

Assume a set of topics $T = \{t_1, t_k, \ldots, t_k\}$ arranged in order of decreasing probability $Q(T) = q_1, q_2, \ldots, q_k$. Let $\sum_{i=1}^{k} q_i = T \leq 1$, so that a document is topicless with probability $1 - T$. The general English probability of a word w can therefore be computed on topicless documents to be $p_w = G_E(w)$ or as $\sum_{i=1}^{k} q_i G_i(w)$. In practice, it is next to impossible to collect a large set of truly topicless documents, so we estimate p_w based on a collection D that is representative of the distribution Q of topics. Ideally, every document would belong only to one topic, but in practice we have to assume that each document is the weighted mixture of general English and a few topics. Since the probabilities of words can differ by many orders of magnitude (both for general English and for the sublanguage defined by any particular topic) we separate the discussion of the high, mid, and low frequency cases.

If a word has approximately constant probability $G_t(w)$ across topics t we say it is a *function word* of English. For such words, the estimate $p_w = (1-T)G_t(w)$ or even simply $G_t(w)$ is reasonable. If a document d has length $l(d) >> 1/p_w$ we expect the word to appear in d at least once. Let us denote the **size** (number of documents) of a collection C by $S(C)$. If D_l contains only those documents in D that are longer than l, we expect $DF(w, D_l) = S(D_l)$. We can turn this around and use this as a method of discovering function words: a reasonable choice of threshold frequency would be 10^{-4} and we can say that the function words of English will be those words that appear in all (or a very large proportion of) those documents that have length $\geq 10^5$. We emphasize that not all words with high observed frequency will meet the test: for example the word *Journal* has about twice the frequency of *when* in the widely used WSJ corpora, but it will fail the $DF(w, D_l) = S(D_l)$ test in any other collection, while *when* will pass. The extreme high end of the distribution, words having 0.2% or greater probability, are generally function words, and the first few hundred function words (which go down to the mid-range) collectively account for about half of any corpus.

Function words are of course not the only words of general English: in the mid-range and below we will make a distinction between *specific* and *non-specific* content words. Informally, a content word is non-specific if it provides little information about the identity of the topic(s) that it appears in. For example, words like *see* or *book* could not be called function words even under the most liberal definition of the term (and there will be many long documents that fail to contain them) but their content is not specific enough: for any topic t, $P(t|w)$ is about the same as the general probability of the topic q_t, or, what is the same by Bayes' rule, $G_t(w)/p_w$ is close to 1.

Exercise 8.4.2 Assume a large collection of topic-classified data. Define an overall measure of 'closeness to 1' that is independent of the distribution Q of topics (does not require the collection to be representative of this distribution).

In practice we rarely have access to a large enough collection of topic-classified data (see `http://about.reuters.com/researchandstandards/corpus` for a collection that covers a narrow range of topics very well), and we have to look at the converse task: what words, if any, are specific to a few topics in the sense that $P(d \in D_t | w \in d) >> P(d \in D_t)$. This is well measured by the number of documents containing the word: for example *Fourier* appears in only about 200k documents in a large collection containing over 200m English documents (see `www.northernlight.com`), while *see* occurs in 42m and *book* in 29m. However in a collection of 13k documents about digital signal processing *Fourier* appears 1100 times, so $P(d \in D_t)$ is about $6.5 \cdot 10^{-5}$ while $P(d \in D_t|w)$ is about $5.5 \cdot 10^{-3}$, two orders of magnitude better. In general, words with low DF values, or what is the same, high 1/DF=IDF **inverse document frequency** values are good candidates for being specific content words. Again, the criterion has to be used with care: it is quite possible that a word has high IDF because of deficiencies in the corpus, not because it is inherently very specific. For example, the word *alternately* has even higher IDF than *Fourier,* yet it is hard to imagine any topic that would call for its use more often than others.

At the low end of the distribution, we find a large number of *hapax legomena* (words that appear only once in the corpus) and *dis legomena* (words that appear only twice). For large corpora, typically about 40% to 60% of all word types appears only once, and another 10% to 15% only twice. This observation provides strong empirical evidence that the vocabulary of any language can not be considered finite: for if it was finite, there would be a smallest probability p among the probabilities of the words, and in any random collection of documents with length $>> 1/p$ we would expect to find no hapaxes at all. Obviously, for hapaxes TF=DF=1, so to the extent every document has a topic this could be established deterministically from the hapax in question. In machine learning terms, this amounts to memorizing the training data, and the general experience is that such methods fail to work well for new data. Overall, we need to balance the TF and IDF factors, and the simplest way of doing this is by the classical TF·IDF formula that looks at the product of these two numbers.

Chapter 9

Speech

Conceptually the techniques of linguistic pattern recognition are largely independent of the medium. But overall performance is influenced by the preprocessing to such an extent that until a few years ago the pattern recognition step was generally viewed as a small appendix to the main body of signal processing knowledge. To this day, it remains impossible to build a serious system without paying close attention to preprocessing, and deep algorithmic work on the recognizer will often yield smaller gains than seemingly more superficial changes to the front end. In 9.1 we introduce a speech coding method, linear prediction, which has played an important role in practical application since the seventies, and in 9.2 we extend the discussion of quantization started in 8.1 from scalars to vectors. Fourier transform-based (homomorphic) techniques are discussed in 9.3. As we shall see, even the most naive kbps scale provides a surprisingly good measure of our understanding the nature of speech: the more we know, the better we can compress the signal.

9.1 Linear prediction

In 8.1 we discussed A-law and μ-law speech coding which, by means of fitting the quantization steps to the long-term amplitude distribution of speech, achieves good quality transmittion at 64kbps. To go beyond this limit, we need to exploit short-term regularities. Perhaps the simplest idea is to use **delta coding,** transmitting the difference between to adjacent samples, rather than the samples themselves. As an example, let us consider the function $\sin(\alpha x)$ on the interval $[0, 1]$ sampled n times. To be sure, if we knew at the receiving end that the function to be transmitted was of this format, we could simply transmit α, so to make this example more realistic we would need to demonstrate that from the space of functions F that provide the input for speech coding $\sin(\alpha x)$ could not be selected at low cost.

The signal power $\int_0^1 \sin^2(\alpha x)$ can be computed exactly as $1/2 - \sin(2\alpha)/4\alpha$ which is very close to $1/2$ for α in the range of interest: in speech, the lower and upper limits of audible frequencies, about 20 and 20,000 Hz, provide reasonable cutoff points with $125 < \alpha < 125,000$. The noise can be estimated as $1/4^b$, yielding an SNR of $\log(4)b$ (ignoring additive constants). With delta coding, the values to be coded are between $-\alpha/n$ and α/n, so the SNR improves by $2\log(n) - 2\log(\alpha)$, which could be quite noticeable for $n >> \alpha$. However, actual speech is a mixture of low and high frequency components, and the gain on lower frequencies is largely offset by the loss at higher frequencies, so the overall impact of this method is small (about one bit per sample can be saved for equivalent speech quality, see Jayant and Noll 1984, Digital Coding of Waveforms. Prentice Hall, Englewood Cliffs, NJ).

Delta coding amounts to an assumption of constancy: we estimate the next sample based on the previous sample, transmitting only an additive correction. This would really work best for a constant function, where there would be nothing to transmit after the first sample. By the same token, if we permit both an additive and a multiplicative correction factor, functions of the form $ae^{bx} + c$ would be the easiest to transmit. Because extreme amplitudes never persist, we can restrict attention to decaying exponentials ($b < 0$), and because the long-term average of speech waveforms is zero, we can assume c=0. This suggests modelling speech by a succession of exponentially decaying impulses. Though the formulas associated with such coding are quite complex, historically this was a very attractive model since exponential decay is easy to model in analog circuitry.

Linear predictive coding (LPC) extends the basic idea of delta coding by considering not only the previous sample but the previous p samples to be available for predicting the next sample. Formally, we define

Chapter 10

Handwriting and machine print

The recognition of handwritten or printed text by computer is referred to as Optical Character Recognition (OCR). When the input device is a digitizer tablet that transmits the signal in real time (as in pen-based computers and personal digital assistants) or includes timing information together with pen position (as in signature capture) we speak of *dynamic* recognition. When the input device is a still camera or a scanner, which captures the position of digital ink on the page but not the order in which it was laid down, we speak of *static* or *image-based* OCR.

The difficulties encountered in dynamic OCR are largely similar to those found in speech recognition: the stream of position/pen pressure values output by the digitizer tablet is analogous to the stream of speech signal vectors output by the audio processing front end, and the same kinds of low-level signal processing and pattern recognition techniques (see Chapters 7-9) are widely employed for both. Here we deal primarily with static OCR, both because such algorithms are critical for transmitting our cultural heritage to the digital age, and because static OCR encompasses a range of problems that have no counterpart in the recognition of spoken or signed language.

In 10.1 we begin with *page decomposition*, which includes both the separation of linguistic material from photos, line drawings, and other non-linguistic information, establishing the local horizontal and vertical directions (deskewing), and the appropriate grouping of titles, headers, footers, and other material set in a font different from the main body of the text. In 10.2 we survey the main approaches to high-level signal processing, and in 10.3 we discuss various the main architectures. For clearly segmented printed materials, the standard (sequential) architecture offers virtually error-free OCR for the most important alphabetic systems including variants of the Latin, Greek, Cyrillic, and Hebrew alphabets. However, when the number of symbols is large, as in the Chinese or Korean writing systems, or the symbols are not separated from one another, as in Arabic or Devanagari print, OCR systems are still far from the error rates of human readers, and the gap between the two is also evident when the quality of the image is compromised e.g. by fax transmission.

10.1 Page decomposition

10.2 Feature extraction

In the general case, OCR begins with a scanned image, often very high resolution (600 dpi or more). Such images are generally *downsampled,* typically to *fax resolution* (200 dpi horizontal, 100 dpi vertical) and *binarized* i.e. greyscale pixel values are replaced by binary (black and white) values. There are many competing algorithms to perform these low-level processing steps: here we will discuss

Since fax images are predominantly stored and transmitted in **G3** (CCITT Group 3) format, we describe this standard in some detail here, because it illustrates not only the basic ideas of Huffman coding (see 7.1) but also the distance between theoretical constructs and practical standards. . Each line is defined as having 1728 (binary) pixels (thus somewhat exceeding the 8.5 inch page width used in the US at 200 dpi) that are *run-length encoded:* instead of transmitting the 0s and 1s, the length of the alternating runs of 0s and 1s get transmitted. Length itself is viewed as a base 64 number: for shorter runs, only the last digit (called *terminating length* in this system) gets transmitted, but for longer runs the preceding digit (called the *make up*) is also used. Since the length distribution of white and black runs differs considerably, two separate codebooks are used. Terminating length and make up codes jointly have the prefix-free property both for white and black, but not for the union of the two codebooks. The following terminating length and make up codes are used:

term. length	white code	black code	term. length	white code	black code
0	00110101	0000110111	32	00011011	000001101010
1	000111	010	33	00010010	000001101011
2	0111	11	34	00010011	000011010010
3	1000	10	35	00010100	000011010011
4	1011	011	36	00010101	000011010100
5	1100	0011	37	00010110	000011010101
6	1110	0010	38	00010111	000011010110
7	1111	00011	39	00101000	000011010111
8	10011	000101	40	00101001	000001101100
9	10100	000100	41	00101010	000001101101
10	00111	0000100	42	00101011	000011011010
11	01000	0000101	43	00101100	000011011011
12	001000	0000111	44	00101101	000001010100
13	000011	00000100	45	00000100	000001010101
14	110100	00000111	46	00000101	000001010110
15	110101	000011000	47	00001010	000001010111
16	101010	0000010111	48	00001011	000001100100
17	101011	0000011000	49	01010010	000001100101
18	0100111	0000001000	50	01010011	000001010010
19	0001100	00001100111	51	01010100	000001010011
20	0001000	00001101000	52	01010101	000000100100
21	0010111	00001101100	53	00100100	000000110111
22	0000011	00000110111	54	00100101	000000111000
23	0000100	00000101000	55	01011000	000000100111
24	0101000	00000010111	56	01011001	000000101000
25	0101011	00000011000	57	01011010	000001011000
26	0010011	000011001010	58	01011011	000001011001
27	0100100	000011001011	59	01001010	000000101011
28	0011000	000011001100	60	01001011	000000101100
29	00000010	000011001101	61	00110010	000001011010
30	00000011	000001101000	62	00110011	000001100110
31	00011010	000001101001	63	00110100	000001100111

make up	white	black
64	11011	0000001111
128	10010	000011001000
192	010111	000011001001
256	0110111	000001011011
320	00110110	000000110011
384	00110111	000000110100
448	01100100	000000110101
512	01100101	0000001101100
576	01101000	0000001101101
640	01100111	0000001001010
704	011001100	0000001001011
768	011001101	0000001001100
832	011010010	0000001001101
896	011010011	0000001110010
960	011010100	0000001110011
1024	011010101	0000001110100
1088	011010110	0000001110101
1152	011010111	0000001110110
1216	011011000	0000001110111
1280	011011001	0000001010010
1344	011011010	0000001010011
1408	011011011	0000001010100
1472	010011000	0000001010101
1536	010011001	0000001011010
1600	010011010	0000001011011
1664	011000	0000001100100
1728	010011011	0000001100101

In order to insure that the receiver maintains color synchronization, all lines must begin with a white run length code word (if the actual scanning line begins with a black run, a white run length of zero will be sent). For each page, first end of line (EOL, 000000000001) is sent, followed by variable-length line codes, each terminated by EOL, with six EOLs at the end of the page.

Exercise 10.2.1 Research the CCITT Group 4 (G4) standard, which also exploits some of the redundancy between successive lines of the scanned image and thereby improves compression by a factor of 2. Research JBIG and JBIG2, which generally improve G4 compression by another factor of 4.

10.3 System architecture

While the early experimental OCR systems were often rule-based, by the eighties these have been completely replaced by systems based on statistical techniques. In the recognition of handprint, algorithms with succesive segmentation, classification, and identification (language modeling) stages are still in the lead. For cursive handwriting, HMMs that make the segmentation, classification, and identification decisions in parallel have proven superior, but performance still leaves much to be desired, both because the spatial and the temporal aspects of the writen signal are not necessarily in lockstep (discontinuous constituents arising e.g. at the crossing of t-s and dotting of i-s) and because the inherent variability of handwriting is far greater than that of speech, to the extent that we often see illegible handwriting but rarerly hear unintelligeble speech.

For cursive machine-print see e.g. Bazzi et al 1999. The state of the art in handwriting recognition is closely tracked by the International Workshop on Frontiers of Handwriting Recognition (IWFHR). For language modeling in OCR see Kornai 1994. A good general introduction to the problems of page decomposition is O'Gorman and Kasturi (1995), and to OCR in general Bunke and Wang (1997).

An OCR-specific problem is that we often find different scripts, such as Kanji and Kana, or Cyrillic and Latin, in the same running text.

Chapter 11

Simplicity

References

Kazimierz Ajdukiewicz. 1935. Die syntaktische konnexität. *Studia Philosophica*, 1:1–27.

Stephen Anderson. 1992. *A-Morphous Morphology.* Cambridge University Press.

Yehoshua Bar-Hillel. 1953. A quasi-arithmetical notation for syntactic description. *Language*, 29:47–58.

Roger L. Berger. 1966. *The undecidability of the domino problem*, volume 66. Memoires of the American Methematical Society.

George Berkeley. 1734. *The Analyst, or a Discourse Addressed to an Infidel Mathematician.*

Juliette Blevins and Sheldon P. Harrison. 1999. Trimoraic feet in gilbertese. *Oceanic Linguistics*, 38:203–230.

Juliette Blevins. 1995. The syllable in phonological theory. In John Goldsmith, editor, *Handbook of Phonology*, pages 206–244. Basil Blackwell.

Leonard Bloomfield. 1926. A set of postulates for the science of language. *Language*, 2:153–164.

Ellen Broselow. 1995. The skeletal tier and moras. In John Goldsmith, editor, *Handbook of Phonology*, pages 175–205. Basil Blackwell.

George Cardona. 1965. On pāṇini's morphophonemic principles. *Language*, 41:225–237.

Yuen Ren Chao. 1961. Graphic and phonetic aspects of linguistic and mathematical symbols. In Roman Jakobson, editor, *Structure of language and its mathematical aspects*, pages 69–82. American Mathematical Society.

Colin Cherry, Morris Halle, and Roman Jakobson. 1953. Toward the logical description of languages in their phonemic aspect. *Language*, 29:34–46.

Colin Cherry. 1956. Roman jakobson's distinctive features as the normal coordinates of a language. In Morris Halle, editor, *For Roman Jakobson.* Mouton, The Hague.

Noam Chomsky and Morris Halle. 1968. *The Sound Pattern of English.* Harper & Row, New York.

Noam Chomsky. 1956. Three models for the description of language. *I. R. E. Transactions on Information Theory*, IT-2.

Noam Chomsky. 1961. On the notion 'rule of grammar'. In Roman Jakobson, editor, *Structure of language and its mathematical aspects*, pages 6–24. American Mathematical Society.

Noam Chomsky. 1965. *Aspects of the theory of syntax.* MIT Press, Cambridge MA.

George N. Clements and Kevin C. Ford. 1979. Kikuyu tone shift and its synchronic consequences. *Linguistic Inquiry*, 10:179–210.

George N. Clements. 1976. Vowel harmony in nonlinear generative phonology. In Wolfgang U. Dressler and Oskar E. Pfeiffer, editors, *Phonologica.* Innsbruck.

George N. Clements. 1985. The geometry of phonological features. *Phonology Yearbook*, 2:225–252.

N. E. Collinge. 1985. *The laws of Indo-European.* Benjamins, Amsterdam.

Bernard Comrie, editor. 1990. *The world's major languages.* Oxford University Press.

Confucius. *The Analects.* Penguin.

Haskell B. Curry. 1961. Some logical aspects of grammatical structure. In Roman Jakobson, editor, *Structure of language and its mathematical aspects*, pages 56–68. American Mathematical Society.

Ferdinand de Saussure. 1879. *Mémoire sur le systeme primitif des voyelles dans les langues indo-européennes.* Teubner, Leipzig.

Murray Eden. 1961. On the formalization of handwriting. In Roman Jakobson, editor, *Structure of language and its mathematical aspects*, pages 83–88. American Mathematical Society.

Andzrej Ehrenfeucht. *personal communication.*

Samuel Eilenberg. 1974. *Automata, languages, and machines.* Academic Press, New York.

H.A. Gleason. 1961. Genetic relationship amond languages. In Roman Jakobson, editor, *Structure of language and its mathematical aspects*, pages 179–189. American Mathematical Society.

John Goldsmith and Gary Larson. 1990. Local modeling and syllabification. In Manuela Noske Karen Deaton and Michael Ziolkowski, editors, *Papers from the 26th Annual Regional Meeting of the Chicago Linguistic Society: Parasession on the Syllable in Phonetics and Phonology.*

John A. Goldsmith. 1976. An overview of autosegmental phonology. *Linguistic Analysis*, 2:23–68.

Barbara Grimes, editor. 2000. *The Ethnologue.* Summer Institute of Linguistics.

Morris Halle and Samuel Jay Keyser. 1971. *English Stress: Its Growth and Its Role in Verse.* Harper and Row, New York.

Morris Halle. 1961. On the role of simplicity in linguistic descriptions. In Roman Jakobson, editor, *Structure of language and its mathematical aspects*, pages 89–94. American Mathematical Society.

Morris Halle. 1964. On the bases of phonology. In Jerry A. Fodor and J. Katz, editors, *The structure of language*, pages 324–333. Prentice-Hall, Englewood Cliffs.

Zellig Harris. 1951. *Methods in structural linguistics.* University of Chicago Press, Chicago.

Michael A. Harrison. 1978. *Introduction to Formal Language Theory.* Addison Wesley, Reading, MA.

Bruce Hayes. 1980. *A metrical theory of stress rules.* Phd thesis, MIT, Cambridge MA.

Bruce Hayes. 1995. *Metrical Stress Theory.* University of Chicago Press, Chicago.

Susan R. Hertz. 1982. From text-to-speech with srs. *Journal of the Acoustical Society of America*, 72:1155–1170.

Henry Hiż. 1961. Congrammaticality, batteries of transformations and grammatical categories. In Roman Jakobson, editor, *Structure of language and its mathematical aspects*, pages 43–50. American Mathematical Society.

Robert D. Hoberman, 1987. *Emphasis (pharyngealization) as an autosegmental harmony feature*, volume 23, pages 167–181.

M. Sharon Hunnicutt. 1976. Phonological rules for a text to speech system. *American Journal of Computational Linguistics*, Microfiche 57:1–72.

Larry M. Hyman. 1982. The representation of nasality in gokana. In Harry van der Hulst and Norval Smith, editors, *The Structure of Phonological Representation*. Foris, Dordrecht.

Neil Immerman. 1988. Nondeterministic space is closed under complementation. *SIAM Journal of Computing*, 17:935–938.

Roman Jakobson, Gunnar Fant, and Morris Halle. 1952. *Preliminaries to speech analysis: the distinctive features and their correlates*. MIT Press, Cambridge MA.

Roman Jakobson. 1957. Mufaxxama - the emphatic phonemes in Arabic. In E. Pulgrim, editor, *Studies presented to Joshua Whatmough*. Mouton, The Hague.

Roman Jakobson, editor. 1961. *Structure of language and its mathematical aspects*. American Mathematical Society.

Ch. Douglas Johnson. 1970. *Formal aspects of phonological representation*. Phd dissertation, UC Berkeley.

Daniel Kahn. 1976. *Syllable-based Generalizations in English Phonology*. Indiana University Linguistics Club, Bloomington. New York: Garland Press, 1980.

Ronald M. Kaplan and Martin Kay. 1994. Regular models of phonological rule systems. *Computational Linguistics*, 20:331–378.

Richard M. Karp. 1972. Reducibility among combinatorial problems. In R. Mille and J. Thatcher, editors, *Complexity of Computer Computations*, pages 85–104. Plenum Press, New York.

Donald E. Knuth. 1971. *The Art of Computer Programming*. Addison-Wesley.

András Kornai. 1993. The generative power of feature geometry. *Annals of Mathematics and Artificial Intelligence*, 8:37–46.

András Kornai. 1999. *Extended finite state models of language*. Cambridge University Press, Cambridge UK.

Kimmo Koskenniemi. 1983. Two-level model for morphological analysis. In *Proceedings of IJCAI-83*, pages 683–685.

Joachim Lambek. 1958. The mathematics of sentence structure. *American Mathematical Monthly*, 65:154–170.

Joachim Lambek. 1961. On the calculus of syntactic types. In Roman Jakobson, editor, *Structure of language and its mathematical aspects*, pages 166–178. American Mathematical Society.

William Leben. 1973. *Suprasegmental phonology.* Phd thesis, MIT, Cambridge MA.

Ilse Lehiste. 1970. *Suprasegmentals.* MIT Press, Cambridge MA.

Juliette Levin. 1985. *A metrical theory of syllabicity.* Phd dissertation, MIT.

Mark Y. Liberman. 1975. *The Intonational System of English.* Phd thesis, MIT.

Scott K. Liddell and Robert E. Johnson. 1989. American sign language: The phonological base. *Sign Language Studies*, 64:195–277.

D. Terence Lnagendoen. 1984. *The Vastness of Natural Languages.* Basil Blackwell.

John Lyons. 1968. *Introduction to theoretical linguistics.* Cambridge University Press, London and New York.

Benoit Mandelbrot. 1961. On the thory of word frequencies and on related markovian models of discourse. In Roman Jakobson, editor, *Structure of language and its mathematical aspects*, pages 190–219. American Mathematical Society.

Christopher D. Manning. 1999. *Foundations of Statistical Natural Language Processing.* MIT Press.

John J. McCarthy. 1979. *Formal Problems in Semitic Phonology and Morphology.* Phd dissertation, MIT.

John J. McCarthy. 1988. Feature geometry and dependency: a review. *Phonetica*, 45(2-4):84–108.

James McCawley. 1968. Concerning the base component of a transformational grammar. *Foundations of Language*, 4:243–269.

Marina Nespor and Irene Vogel. 1989. On clashes and lapses. *Phonology*, 7:69–116.

David Odden. 1995. Tone: African languages. In John Goldsmith, editor, *Handbook of Phonology*, pages 445–475. Basil Blackwell.

Anthony G. Oettinger. 1961. Automatic syntactic analysis and the pushdown store. In Roman Jakobson, editor, *Structure of language and its mathematical aspects*, pages 104–129. American Mathematical Society.

Gordon E. Peterson and Frank Harary. 1961. Foundations in phonemic theory. In Roman Jakobson, editor, *Structure of language and its mathematical aspects*, pages 139–165. American Mathematical Society.

William J. Poser. 1990. Evidence for foot structure in japanese. *Language*, 66:78–105.

Hilary Putnam. 1961. Some issues in the theory of grammar. In Roman Jakobson, editor, *Structure of language and its mathematical aspects*, pages 25–42. American Mathematical Society.

Willard V. Quine. 1961. Logic as a source of syntactical insights. In Roman Jakobson, editor, *Structure of language and its mathematical aspects*, pages 1–5. American Mathematical Society.

Abraham Robinson. 1966. *Non-Standard Analysis.* North Holland.

H.G. van der Hulst R.W.N. Goedemans and E.A.M. Visch. 1996. *Stress Pat-*

terns of the world. Part 1: Background. Holland Academic Graphics, The Hague.

Arto Salomaa. 1973. *Formal Languages*. Academic Press, New York.

Wendy Sandler. 1989. *Phonological Representation of the Sign: Linearity and Nonlinearity in American Sign Language*. Foris Publications, Dordrecht.

Elisabeth O. Selkirk. 1984. *Phonology and Syntax: The Relation Between Sound and Structure*. MIT Press, Cambridge MA.

D. D. Sleator and D. Temperley. 1993. Parsing English with a link grammar. In *Third International Workshop on Parsing Technologies*.

J. F Staal. 1962. A method of linguistic description: the order of consonants according to pāṇini. *Language*, 38:1–10.

Finite state devices for natural language processing. 1997. *Emanuel Roche and Yves Schabes*. MIT Press, Cambridge MA.

Kenneth N. Stevens and Sheila E. Blumstein. 1981. The search for invariant acoustic correlates of phonetic features. In P. Eimas and J. Miller, editors, *Perspectives on the study of speech*. Lawrence Erlebaum Associates, Hillsdale, NJ.

Róbert Szelepcsényi. 1987. The method of forcing for nondeterministic automata. *Bulletin of the European Association for Theoretical Computer Science*, 33:96–100.

Höskuldur Thráinsson. 1978. On the phonology of icelandic preaspiration. *Nordic Journal of Linguistics*, 1:3–54.

Nikolai Sergeevich Trubetskoi. 1939. *Grundzüge der Phonologie*. Vandenhoeck & Ruprecht, Göttingen. Transl: *Principles of Phonology* (Berkeley, University of California Press, 1969).

Karl von Frisch. 1967. *Dance language and orientation of bees*. Belknap Press, Cambridge, MA.

John Wilkins. 1668. *Essay toward a real character, and a philosophical language*.

Karina Wilkinson. 1988. Prosodic structure and lardil phonology. *Linguistic Inquiry*, 19:325–334.

Edwin Williams. 1976. Underlying tone in Margi and Igbo. *Linguistic Inquiry*, 7:463–84.

Victor H. Yngve. 1961. The depth hypothesis. In Roman Jakobson, editor, *Structure of language and its mathematical aspects*, pages 130–138. American Mathematical Society.

Index