Morphology

Morphology is the area of linguistics which studies the structure of words.

Almost all natural language applications require some processing of words: lexicon lookup, morphological analysis and generation, part-of-speech determination etc.

In order to implement such functions, it is necessary to understand which morphological processes take place in a variety of languages.

Why look at many languages?

Example

hem dibbru koll ha-layla

Observations:

- dibbru is third person, plural, past form of the verb *dibber*
- this form is obtained by concatenating the suffix [u] to the base [dibber]
- in the inflected form *dibbru*, the vowel [e] of the base [dibber] is reduced to a schwa. This reduction is mandatory, as [dibberu] is ungrammatical.

Example

These simple observations shed light on a variety of issues:

- What information is encoded by morphology?

  In the example, morphology encodes details such as person, number and tense.

- How does morphology encode information?

  In the example, the final form is obtained by concatenating an affix (which is not a word) to the end of a base (which might be a word).

- Interaction of morphology and phonology

  In the example, the vowel [e] is shortened to a schwa.

Structure of this part of the course

- Typology of languages
- Inflection and derivation
- What information is encoded by morphology
- How morphology encodes information
  - concatenation, infixation, circumfixation, root and pattern, reduplication
- Interaction of morphology and phonology

Typology of languages

Isolating : no bound forms. Example: Mandarin Chinese

Agglutinative : bound forms occur and are arranged in the word like beads on a string. Example: Turkish

Polysynthetic : elements that often occur as separate words in other languages (such as arguments of the verb) are expressed morphologically. Example: Yupik (central Alaska)

Inflectional : distinct features are merged into a single bound form. Example: Latin

Isolating languages

No bound forms. Example: Mandarin Chinese

gǒu bú ài chī qīngcài
dog not like eat vegetable

Can mean any of the following (inter alia):

- the dog doesn't like to eat vegetables
- the dog didn't like to eat vegetables
- the dogs don't like to eat vegetables
- the dogs didn't like to eat vegetables
- dogs don't like to eat vegetables

Agglutinative languages

Beads on a string. Example: Turkish

çöplüklerimizdekiledenmiydi

çöp	lük	ler	imiz	de	ki	ler	den	mi	y	di	
garbage	*Aff*	*Pl*	*1p/Pl*	*Loc*	*Rel*	*Pl*	*Abl*	*Int*	*Aux*	*Past*	

"was it from those that were in our garbage cans?"

"h-mi-$e-b-paxeinu?"

Polysynthetic languages

Morphology encodes units that are usually considered syntactic (as in noun incorporation). Example: Yupik

qayá:liy'u:l'u:n'i

qayá: li y'u: l'u: n'i
kayaks make excellent he Past
"he was excellent at making kayaks"

"The grammar is in the morphology"

Inflectional languages

Portmanteau morphemes: a single morpheme can encode various bits of information. Example: Latin

amó

am ó
love 1p/Sg/Pres/Indicative/Active

Inflections and derivations

Inflectional morphology takes as input a word and outputs a form of the same word appropriate to a particular context.

Example: [dibber] $\Rightarrow$ [dibbru]

The output is appropriate to a context in which the subject is third person plural and the tense is past.

Hence: words have *paradigms*, defining all possible inflected forms of a word. Words which belong to the same paradigm are all *inflected forms* of a single *lexeme*.

Inflections and derivations

Derivational morphology takes as input a word and outputs a different word that is derived from the input. This is also called *word formation*.

Example: establish+ment+ary+an+ism

Example: *hexlit* $\longrightarrow$ *haxlata* $\longrightarrow$ *hexleti* $\longrightarrow$ *hexletiyut*

Inflections and derivations - distinctive criteria

- Inflection does not change the part-of-speech, derivation might.

  haxlata–haxlatot; *haxlata–hexleti*

- Inflection is sometimes required by the syntax, derivation never is.

- If a language marks an inflectional category, it marks it on all appropriate words. In other words, the relation denoted by inflectional morphology is *productive*.

haxlata	–	*haxlatot*		*haxlata*	–	*hexleti*
hapgana	–	*hapganot*		*hapgana*	–	$*$*hepgeni*

Verbal morphology

Verbs specify the number (and type) of arguments they may take. In many languages, morphological devices modify these lexically specified markings.

Example: passivization (Latin)

puer Cicerōnem laudat
*boy Cicero praise/3/Sg/Pres/Ind/***Act**
"the boy praises Cicero"

Cicerōnem laudātur
*Cicero praise/3/Sg/Pres/Ind/***Pass**
"Cicero is praised"

Example: causativization

napal $\rightarrow$ *hippil*; *nasa&* $\rightarrow$ *hissi&*

Verbal morphology

Verbs are commonly marked with indications of the time at which the situations denoted by them occurred, or the state of completion of the situation. Such markers encode *tense* and *aspect*, respectively.

Example: Latin

vir Cicerōnem laudābō
*man Cicero praise/3/Sg/***Future***/Ind*
"the man will praise Cicero"

vir Cicerōnem laudāvit
*man Cicero praise/3/Sg/***Perf***/Ind*
"the man has praised Cicero"

Verbal morphology

In many languages the verb must *agree* on person, number, gender or other features with one or more of its arguments.

Example:

The princess kisses the frog
**The princess kiss the frog*

hem dibbru koll ha-layla
**hem dibbra koll ha-layla*

In some languages (e.g., Georgian and Chicheŵa) verbs agree not only with their subjects but also with their objects.

Nominal morphology

Inflectional categories for nouns (and adjectives) include

- number (singular, plural, dual)
- case (marking various kinds of semantic function)
- gender (feminine, masculine, neuter)

Latin has five cases: nominative, genitive, dative, accusative, ablative.

Finnish has fourteen different cases!

Example: the inflection paradigm of the noun *magnus* (big) in Latin.

The inflection paradigm of Latin magnus

		masculine	feminine	neuter
sing.	nom	magn+**us**	magn+**a**	magn+**um**
	gen	magn+**ī**	magn+**ae**	magn+**ī**
	dat	magn+**ō**	magn+**ae**	magn+**ō**
	acc	magn+**um**	magn+**am**	magn+**um**
	abl	magn+**ō**	magn+**ā**	magn+**ō**
plur.	nom	magn+**ī**	magn+**ae**	magn+**a**
	gen	magn+**ōrum**	magn+**ārum**	magn+**ōrum**
	dat	magn+**īs**	magn+**īs**	magn+**īs**
	acc	magn+**ōs**	magn+**ās**	magn+**a**
	abl	magn+**īs**	magn+**īs**	magn+**īs**

Nominal morphology

Many languages distinguish between two or three grammatical genders: feminine, masculine and neuter.

In some languages, such as the Bantu languages, more detailed gender classes exist.

Example: Swahili has inflection affixes for humans, thin objects, paired things, instruments and extended body parts, inter alia.

Adjectival morphology

Many languages express comparison of adjectives morphologically.

Example: Welsh

gwyn	gwynn+**ed**	gwynn+**ach**	gwynn+**af**
white	as white	whiter	whitest
teg	tec+**ed**	tec+**ach**	tec+**af**
fair	as fair	fairer	fairest

Derivational morphology

In general, derivational morphology is not as productive as inflectional morphology.

Nominalization: destroy → destruction; *$amar* → *$mira*; *pittex* → *pittux*; *hiskim* → *heskem*

Deverbal adjectives: drink → drinkable; *nazal* → *nazil*

Denominalized adjectives: *$ulxan* → *$ulxani*

Adjective nominalization: grammatical → grammaticality; *nadir* → *ndirut*

Negation: able → unable; *xuti* → *'alxuti*

Compounding

In contrast to derivations and inflections, where affixes are attached to a stem, in compounding two or more lexemes' stems are joint together, forming another lexeme.

Example: policeman; newspaper; *beit seper*; *ypat &einaym*

Both lexemes might undergo modification in the process.

In German, the concatenation is expressed in the orthography:

lebensversicherungsgesellschaftsangestellter

leben	s	versicherung	s	gesellschaft	s	angestellter
life		*insurance*		*company*		*employee*

What are morphemes?

In order to know what morphemes are, it is useful to check in what ways they are expressed.

The simplest model of morphology is the situation where a morphologically complex word can be analyzed as a series of morphemes concatenated together.

An example: Turkish. Not only is Turkish morphology exclusively concatenative; in addition, all affixes are suffixes. Turkish words are of the form *stem suffix**.

çöplüklerimizdekiledenmiydi

çöp	lük	ler	imiz	de	ki	ler	den	mi	y	di	
garbage	*Aff*	*Pl*	*1p/Pl*	*Loc*	*Rel*	*Pl*	*Abl*	*Int*	*Aux*	*Past*	

What are morphemes?

Linear concatenation is not the only way in which languages put morphemes together. Affixes may also attach as *infixes* inside words.

Example: Bontoc (Philippines)

fikas	→	f-**um**+ikas
strong		*be strong*

kilad	→	k-**um**+ilad
red		*be red*

fusul	→	f-**um**+usul
enemy		*be an enemy*

What are morphemes?

In the Bontoc case the infix must be placed after the first consonant of the word to which it attaches.

In general, the placement of infixes is governed by prosodic principles.

Example: Ulwa (Nicaragua)

suu+**ki**-lu	my dog
suu+**ma**-lu	your (Sg) dog
suu+**ka**-lu	his/her/its dog
suu+**ni**-lu	our (inclusive) dog
suu+**ki+na**-lu	our (exclusive) dog
suu+**ma+na**-lu	your (Pl) dog
suu+**ka+na**-lu	their dog

What are morphemes?

Some languages exhibit *circumfixes*, affixes which attach discontinuously around a stem.

Example: German participles

säuseln	**ge**+säusel+**t**
brüsten	**ge**+brüst+**et**
täuschen	**ge**+täusch+**t**

What are morphemes?

In contrast to processes of attaching an affix to a stem, there exist also nonsegmental morphological processes. A typical example is the Semitic *root and pattern* morphology.

Example: Hebrew *binyanim*

_a_a_, ni__a_, _i__el, _u__a_, hi__i_, hu__a_, hit_a__e_.

What are morphemes?

Another nonsegmental process is *reduplication*.

Example: Indonesian

orang → orang+orang
man *men*

Sometimes only part of the word is duplicated, as in Yidin (Australia) plural:

mulari → mula+mulari
man *men*

gindalba → gindal+gindalba
lizard *lizards*

So, what are morphemes?

In its most general definition, a morpheme is an ordered pair $\langle \text{cat}, \text{phon} \rangle$, where cat is the morphological category expressed by the morpheme (for example, its syntactic and semantic features), and phon represents its phonological form, including the ways in which it is attached to its stem.

Example:

$\langle (Adj \rightarrow N, \text{“state of”}), (\text{[ut], suffix}) \rangle$ $\quad$ $nadir \rightarrow ndirut$

$\langle (root \rightarrow V, causative), (\_i\_\_e\_) \rangle$ $\quad$ $g.d.l \rightarrow$ giddel

What are words, then?

A morpheme is a pairing of syntactic/semantic information with phonological information. In the same way, it is useful to assume that words have dual structures: phonological and morphological. The two structures are not always isomorphic.

It is a fairly traditional observation in morphology that there are really two kinds of words from a structural point of view: phonological words and syntactic words. These two notions specify overlapping but not identical sets of entities. furthermore, the orthographic word might not correspond to any of these.

What information should a morphological analyzer produce?

The answer depends on the application:

Sometimes it is sufficient to know that *dibbru* is an inflected form of *dibber*; sometimes morphological information is needed, either as a list of features (*dibbru* is third person, plural, past form of the verb *dibber*) or as a structure tree; sometimes it is better to produce a list of phonemes without determining word boundaries. For some applications, the root *d.b.r* might be needed.

Morphotactics

Morphotactics investigates the constraints imposed on the order in which morphemes are combined.

Various kinds of such constraints are known.

Example:

teva& $\rightarrow$ *tiv&i* $\rightarrow$ *tiv&iyut* $\rightarrow$ *&al-tiv&iyut* but
$*$*tiv&iyut-&al*; $*$*&al-tiv&uti*

Morphotactics

Types of constraints:

- Constraints on the type of the affix: *&al* is a prefix, *ut* is a suffix
- Syntactic constraints: [i] converts a noun to an adjective; [ut] converts an adjective to a noun
- Other constraints: in English, "Latin" affixes are attached before "native" ones:

non+im+partial	non+il+legible
*in+non+partial	*in+non+legible

Phonology

Ideally, the task of a morphological analysis system would be to break the word down to its component morphemes and determine the meaning of the resulting decomposition.

Things are not that simple because of the often quite drastic effects of phonological rules. A great deal of the effort in constructing computational models of morphology is spent on developing techniques for dealing with phonological rules.

Since most computational analyses of morphology assume *written* input, phonological rules are often confused with orthographic ones.

Phonology

Orthographic rules often do not correspond to phonological rules.

An orthographic rule that does not correspond to any phonological rule:

city+s $\rightarrow$ cities (and not $*$citys)

bake+ing $\rightarrow$ baking (and not $*$bakeing)

Phonology

A phonological rule (changing [a^j] to [i]) is not reflected in the orthography:

divine+ity $\longrightarrow$ divinity

A phonological rule (stress shift) is not reflected in the orthography:

grammátical $\longrightarrow$ grammaticálity

Phonology

Examples of phonological rules

English: [n] changes to [m] before a labial consonant:

impossible; **im**pose; **im**mortal

Finnish: vowel harmony

NOM	PART	gloss
taivas	taivas+**ta**	sky
puhelin	puheli+**ta**	telephone
lakeus	lakeus+**ta**	plain
syy	syy+**tä**	reason
lyhyt	lyhyt+**tä**	short
ystävällinen	ystävällinen+**tä**	friendly

Implementing morphology and phonology

We begin with a simple problem: a lexicon of some natural language is given as a list of words. Suggest a data structure that will provide insertion and retrieval of data. As a first solution, we are looking for time efficiency rather than space efficiency.

The solution: *trie* (word tree).

Access time: $O(|w|)$. Space requirement: $O(\sum_w |w|)$.

A trie can be augmented to store also a morphological dictionary specifying concatenative affixes, especially suffixes. In this case it is better to turn the tree into a graph.

The obtained model is that of *finite-state automata*.

Finite-state technology

Finite-state automata are not only a good model for representing the lexicon, they are also perfectly adequate for representing dictionaries (lexicons+additional information), describing morphological processes that involve concatenation etc.

A natural extension of finite-state automata – finite-state transducers – is a perfect model for most processes known in morphology and phonology, including non-segmental ones.

Formal language theory – definitions

Formal languages are defined with respect to a given *alphabet*, which is a finite set of symbols, each of which is called a *letter*.

A finite sequence of letters is called a *string*.

Example: Strings
Let $\Sigma = \{0, 1\}$ be an alphabet. Then all binary numbers are strings over Σ.

If $\Sigma = \{a, b, c, d, \ldots, y, z\}$ is an alphabet then *cat*, *incredulous* and *supercalifragilisticexpialidocious* are strings, as are *tac*, *qqq* and *kjshdflkwjehr*.

Formal language theory – definitions

The *length* of a string w, denoted $|w|$, is the number of letters in w. The unique string of length 0 is called the *empty string* and is denoted ϵ.

If $w_1 = \langle x_1, \ldots, x_n \rangle$ and $w_2 = \langle y_1, \ldots, y_m \rangle$, the *concatenation* of w_1 and w_2, denoted $w_1 \cdot w_2$, is the string $\langle x_1, \ldots, x_n, y_1, \ldots, y_m \rangle$. $|w_1 \cdot w_2| = |w_1| + |w_2|$.

For every string w, $w \cdot \epsilon = \epsilon \cdot w = w$.

Formal language theory – definitions

Example: Concatenation
Let $\Sigma = \{a, b, c, d, \ldots, y, z\}$ be an alphabet. Then *master* · *mind* = *mastermind*, *mind* · *master* = *mindmaster* and *master* · *master* = *mastermaster*. Similarly, *learn* · *s* = *learns*, *learn* · *ed* = *learned* and *learn* · *ing* = *learning*.

Formal language theory – definitions

An *exponent* operator over strings is defined in the following way: for every string w, $w^0 = \epsilon$. Then, for $n > 0$, $w^n = w^{n-1} \cdot w$.

Example: Exponent
If $w = go$, then $w^0 = \epsilon$, $w^1 = w = go$, $w^2 = w^1 \cdot w = w \cdot w = gogo$, $w^3 = gogogo$ and so on.

Formal language theory – definitions

The *reversal* of a string w is denoted w^R and is obtained by writing w in the reverse order. Thus, if $w = \langle x_1, x_2, \ldots, x_n \rangle$, $w^R = \langle x_n, x_{n-1}, \ldots, x_1 \rangle$.

Given a string w, a *substring* of w is a sequence formed by taking contiguous symbols of w in the order in which they occur in w. If $w = \langle x_1, \ldots, x_n \rangle$ then for any i, j such that $1 \leq i \leq j \leq n$, $\langle x_i, \ldots x_j \rangle$ is a substring of w.

Two special cases of substrings are *prefix* and *suffix*: if $w = w_l \cdot w_c \cdot w_r$ then w_l is a prefix of w and w_r is a suffix of w.

Formal language theory – definitions

Example: Substrings
Let $\Sigma = \{a, b, c, d, \ldots, y, z\}$ be an alphabet and $w =$ *indistinguishable* a string over Σ. Then ϵ, *in*, *indis*, *indistinguish* and *indistinguishable* are prefixes of w, while ϵ, *e*, *able*, *distinguishable* and *indistinguishable* are suffixes of w. Substrings that are neither prefixes nor suffixes include *distinguish*, *gui* and *is*.

Formal language theory – definitions

Given an alphabet Σ, the set of all strings over Σ is denoted by Σ^*.

A *formal language* over an alphabet Σ is a subset of Σ^*.

Formal language theory – definitions

Example: Languages
Let $\Sigma = \{a, b, c, \ldots, y, z\}$. Then Σ^* is the set of all strings over the Latin alphabet. Any subset of this set is a language. In particular, the following are formal languages:

Formal language theory – definitions

- Σ^*;
- the set of strings consisting of consonants only;
- the set of strings consisting of vowels only;
- the set of strings each of which contains at least one vowel and at least one consonant;
- the set of palindromes;
- the set of strings whose length is less than 17 letters;
- the set of single-letter strings;
- the set {*i, you, he, she, it, we, they*};
- the set of words occurring in Joyce's Ulysses;
- the empty set;

Note that the first five languages are infinite while the last five are finite.

Formal language theory – definitions

The string operations can be lifted to languages.

If L is a language then the *reversal* of L, denoted L^R, is the language $\{w \mid w^R \in L\}$.

If L_1 and L_2 are languages, then
$L_1 \cdot L_2 = \{w_1 \cdot w_2 \mid w_1 \in L_1 \text{ and } w_2 \in L_2\}$.

Example: Language operations
$L_1 = \{i, you, he, she, it, we, they\}$, $L_2 = \{smile, sleep\}$.

Then ${L_1}^R = \{i, uoy, eh, ehs, ti, ew, yeht\}$ and $L_1 \cdot L_2 = \{ismile, yousmile, hesmile, shesmile, itsmile, wesmile, theysmile, isleep, yousleep, hesleep, shesleep, itsleep, wesleep, theysleep\}$.

Formal language theory – definitions

If L is a language then $L^0 = \{\epsilon\}$.
Then, for $i > 0$, $L^i = L \cdot L^{i-1}$.

Example: Language exponentiation
Let L be the set of words $\{$*bau*, *haus*, *hof*, *frau*$\}$. Then $L^0 = \{\epsilon\}$, $L^1 = L$ and $L^2 = \{$*baubau*, *bauhaus*, *bauhof*, *baufrau*, *hausbau*, *haushaus*, *haushof*, *hausfrau*, *hofbau*, *hofhaus*, *hofhof*, *hoffrau*, *fraubau*, *frauhaus*, *frauhof*, *fraufrau*$\}$.

Formal language theory – definitions

The *Kleene closure* of L and is denoted L^* and is defined as $\bigcup_{i=0}^{\infty} L^i$.
$L^+ = \bigcup_{i=1}^{\infty} L^i$.

Example: Kleene closure
Let $L = \{dog, cat\}$. Observe that $L^0 = \{\epsilon\}$, $L^1 = \{dog, cat\}$, $L^2 = \{catcat, catdog, dogcat, dogdog\}$, etc. Thus L^* contains, among its infinite set of strings, the strings ϵ, *cat*, *dog*, *catcat*, *catdog*, *dogcat*, *dogdog*, *catcatcat*, *catdogcat*, *dogcatcat*, *dogdogcat*, etc.

The notation for Σ^* should now become clear: it is simply a special case of L^*, where $L = \Sigma$.

Regular expressions

Regular expressions are a formalism for defining (formal) languages. Their "syntax" is formally defined and is relatively simple. Their "semantics" is sets of strings: the denotation of a regular expression is a set of strings in some formal language.

Regular expressions

Regular expressions are defined recursively as follows:

- $\emptyset$ is a regular expression
- ϵ is a regular expression
- if $a \in \Sigma$ is a letter then a is a regular expression
- if r_1 and r_2 are regular expressions then so are $(r_1 + r_2)$ and $(r_1 \cdot r_2)$
- if r is a regular expression then so is $(r)^*$
- nothing else is a regular expression over Σ.

Regular expressions

Example: Regular expressions
Let Σ be the alphabet $\{a, b, c, \ldots, y, z\}$. Some regular expressions over this alphabet are:

- $\emptyset$
- a
- $((c \cdot a) \cdot t)$
- $(((m \cdot e) \cdot (o)^*) \cdot w)$
- $(a + (e + (i + (o + u))))$
- $((a + (e + (i + (o + u)))))^*$

Regular expressions

For every regular expression r its denotation, $[\![r]\!]$, is a set of strings defined as follows:

- $[\![\emptyset]\!] = \emptyset$
- $[\![\epsilon]\!] = \{\epsilon\}$
- if $a \in \Sigma$ is a letter then $[\![a]\!] = \{a\}$
- if r_1 and r_2 are regular expressions whose denotations are $[\![r_1]\!]$ and $[\![r_2]\!]$, respectively, then $[\![(r_1 + r_2)]\!] = [\![r_1]\!] \cup [\![r_2]\!]$, $[\![(r_1 \cdot r_2)]\!] = [\![r_1]\!] \cdot [\![r_2]\!]$ and $[\![(r_1)^*]\!] = [\![r_1]\!]^*$

Regular expressions

Example: Regular expressions and their denotations

$\emptyset$	$\emptyset$
a	$\{a\}$
$((c \cdot a) \cdot t)$	$\{c \cdot a \cdot t\}$
$(((m \cdot e) \cdot (o)^*) \cdot w)$	{*mew, meow, meoow, meooow, meooo*
$(a + (e + (i + (o + u))))$	$\{a, e, i, o, u\}$
$((a + (e + (i + (o + u)))))^*$	*all strings of 0 or more vowels*

Regular expressions

Example: Regular expressions
Given the alphabet of all English letters, $\Sigma = \{a, b, c, \ldots, y, z\}$, the language Σ^* is denoted by the regular expression Σ^*.

The set of all strings which contain a vowel is denoted by $\Sigma^* \cdot (a + e + i + o + u) \cdot \Sigma^*$.

The set of all strings that begin in "*un*" is denoted by $(un)\Sigma^*$.

The set of strings that end in either "*tion*" or "*sion*" is denoted by $\Sigma^* \cdot (s + t) \cdot (ion)$.

Note that all these languages are infinite.

Properties of regular languages

Closure properties:

A class of languages $\mathcal{L}$ is said to be closed under some operation '$\bullet$' if and only if whenever two languages L_1, L_2 are in the class ($L_1, L_2 \in \mathcal{L}$), also the result of performing the operation on the two languages is in this class: $L_1 \bullet L_2 \in \mathcal{L}$.

Properties of regular languages

Regular languages are closed under:

- Union
- Intersection
- Complementation
- Difference
- Concatenation
- Kleene-star
- Substitution and homomorphism