“The Sound Pattern of English” by Chomsky & Hall
(Chapter 1 & 2, 1991)

Martina Köhler

6th of December 2016

Eberhard Karls Universität Tübingen,
Department of Computer Linguistics: “Finite State Morphology”
1. Linguistic Universals
   1.1 Motivation for Finite State Automaton
   1.2 What are Linguistic Universals?
   1.3 Grammar of English
   1.4 Theory
   1.5 Surface Structure

2. Transformational Cycle
   2.1 Examples
   2.2 General Principle

3. Transformational Cycle Applied To Finite State Automaton
   3.1 More Examples
   3.2 Transformational Cycle Within the Word
Motivation for Finite State Automaton

- Transformational Cycle is important for Modeling
Motivation for Finite State Automaton

- Transformational Cycle is important for Modeling
- useful for figuring out fitting automaton
Motivation for Finite State Automaton

- Transformational Cycle is important for Modeling
- useful for figuring out fitting automaton
- you need to know the underlying rules
Motivation for Finite State Automaton

- Transformational Cycle is important for Modeling
- useful for figuring out fitting automaton
- you need to know the underlying rules
- so that you can form a finite state automaton & a regular expression
What are Linguistic Universals?

Grammar

- Hypothesis concerning the actual internalized grammar of the speaker-hearer
What are Linguistic Universals?

<table>
<thead>
<tr>
<th>Grammar</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Hypothesis concerning the actual internalized grammar of the speaker-hearer</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Linguistic Universals</th>
</tr>
</thead>
<tbody>
<tr>
<td>• An attempt to develop a theory of natural language that is built on a system of hypotheses</td>
</tr>
<tr>
<td>• These hypotheses are constructed by a list of properties</td>
</tr>
</tbody>
</table>
What are Linguistic Universals?

**Grammar**

- Hypothesis concerning the actual internalized grammar of the speaker-hearer

**Linguistic Universals**

- An attempt to develop a theory of natural language that is built on a system of hypotheses
- These hypotheses are constructed by a list of properties
- Properties are referred to as “linguistic universals”
What are Linguistic Universals?

**Grammar**
- Hypothesis concerning the actual internalized grammar of the speaker-hearer

**Linguistic Universals**
- An attempt to develop a theory of natural language that is built on a system of hypotheses
- These hypotheses are constructed by a list of properties
- Properties are referred to as “linguistic universals”
- Something that all languages have in common
The Grammar of English

- The grammar of a language assigns a structural description to each phonetic representation
The Grammar of English

• The grammar of a language assigns a structural description to each phonetic representation

• Example: “dogs” = # dog # s # = noun + plural
The Grammar of English

- The grammar of a language assigns a structural description to each phonetic representation.
- Example: “dogs” = # dog # s # = noun + plural
- That’s a structural description that specifies the semantic meaning behind each phonetic representation.
The Grammar of English

• The grammar of a language assigns a structural description to each phonetic representation

• **Example:** “dogs” = # dog # s # = noun + plural

• that’s a structural description that specifies the semantic meaning behind each phonetic representation

• **Exception to this rule are “semi-grammatical sentences”**

• interpretable but not well understood
Infinity of Language

- the class of phonetic representations of well-formed sentences in each human language is infinite

\[ A = (Q_{\text{zero.pnum}}, \Sigma, \delta, q_{\text{zero.pnum}}, F) \]

\[ \delta: q_{\text{zero.pnum}} \cdot a \]

What would be the regular expression for this automaton?

\[ \text{reg} \ A = a^* / \text{five.pnum} / \text{four.pnum} / \text{zero.pnum} \]
Infinity of Language

- the class of phonetic representations of well-formed sentences in each human language is infinite
- no human language has a limit on the number of sentences that can be formed
Infinity of Language

- the class of phonetic representations of well-formed sentences in each human language is infinite
- no human language has a limit on the number of sentences that can be formed

\[ A := (\{q_0\}, \{a\}, \delta, q_0, \{q_0\}) \]

What would be the regular expression for this automaton?

\[ \text{reg} \quad A = a^* / \text{five.pnum} / \text{four.pnum} / \text{zero.pnum} \]
Infinity of Language

• the class of phonetic representations of well-formed sentences in each human language is infinite
• no human language has a limit on the number of sentences that can be formed

\[ A := (\{q_0\}, \{a\}, \delta, q_0, \{q_0\}) \]

\[ \delta : \]

What would be the regular expression for this automaton?

\[ \text{reg} A = a^* /five.pnum//four.pnum/zero.pnum \]
Infinity of Language

- the class of phonetic representations of well-formed sentences in each human language is infinite
- no human language has a limit on the number of sentences that can be formed

$A := (\{q_0\}, \{a\}, \delta, q_0, \{q_0\})$

$a$

$q_0$

$\delta :$

- What would be the regular expression for this automaton?
- $\text{reg}_A = a^*$
Regular Expression: $a^*$ is finite, even though the derived language itself is infinite (just like natural language)
• Regular Expression: $a^*$ is finite, even though the derived language itself is infinite (just like natural language)
• Therefore the grammar of a language is finite as well
• Regular Expression: $a^*$ is finite, even though the derived language itself is infinite (just like natural language)
• Therefore the grammar of a language is finite as well
• since there are not infinitely many rules that can be applied to a language
Recursive Language

- Grammar for a regular language can be recursive, just like human language

Example
- John left.
- I was surprised that John left.
- Bill expected me to be surprised that John left.

Another Example
- The cat meowed.
- The cat that the dog chased meowed.
- The cat that the dog that the man hit chased meowed.
Recursive Language

- Grammar for a regular language can be recursive, just like human language
- Consider following examples:

<table>
<thead>
<tr>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>John left.</td>
</tr>
<tr>
<td>I was surprised that John left.</td>
</tr>
<tr>
<td>Bill expected me to be surprised that John left.</td>
</tr>
</tbody>
</table>
Recursive Language

- Grammar for a regular language can be recursive, just like human language
- Consider following examples:

**Example**

John left.
I was surprised that John left.
Bill expected me to be surprised that John left.

**Another Example**

The cat meowed.
The cat that the dog chased meowed.
The cat that the dog that the man hit chased meowed.
Recursive Language

How well do you understand these sentences?

Chomsky believes that the limited comprehension is a memory capacity problem, not a limitation of language. However, only the "syntactic components" of a language have this recursive property. The more recursive a statement, the more transformational cycles need to be applied.

-> That empirical question needs to be researched.
Recursive Language

- How well do you understand these sentences?
- Chomsky believes that the limited comprehension is a memory capacity problem, not a limitation of language
Recursive Language

- How well do you understand the these sentences?
- Chomsky believes that the limited comprehension is a memory capacity problem, not a limitation of language
- However, only the “syntactic components” of a language have this recursive property
Recursive Language

- How well do you understand these sentences?
- Chomsky believes that the limited comprehension is a memory capacity problem, not a limitation of language.
- However, only the “syntactic components” of a language have this recursive property.
- The more recursive a statement, the more transformational cycles need to be applied.
Recursive Language

- How well do you understand these sentences?
- Chomsky believes that the limited comprehension is a memory capacity problem, not a limitation of language.
- However, only the "syntactic components" of a language have this recursive property.
- The more recursive a statement, the more transformational cycles need to be applied.
- \(\rightarrow\) That empirical question needs to be researched.
Components of a Grammar: General Theory

Semantic Component

Syntactic Component

Phonological Component

assigns

semantic interpretation

infinite number of syntactic descriptions

assigns

“deep structure”

contains & makes references

partially determines

takes into account

surface structure and structural description
### Structural Descriptions

Structural descriptions are what we use in finite state morphology as the upper “description level”.

Example: ‘boys’ ⇒ noun + plural
Structural Descriptions

Structural descriptions are what we use in finite state morphology as the upper “description level”.
Example: ‘boys’ ⇒ noun + plural

- All structural descriptions contain:
  - full syntactic descriptions
  - semantic representations
  - phonetic representations

- Thus, grammar generates an infinite number of sentences
Structural Descriptions

Structural descriptions are what we use in finite state morphology as the upper “description level”.
Example: ‘boys” ⇒ noun + plural

• All structural descriptions contain:
  - full syntactic descriptions
  - semantic representations
  - phonetic representations

• Thus, grammar generates an infinite number of sentences

• with infinite sound-meaning correspondences
Surface Structure

How can we generate Surface Structure?

Formatives

A string of minimal elements.
Each formative is assigned to various categories that determine its abstract underlying form.
Surface Structure

How can we generate Surface Structure?

Formatives

A string of minimal elements.
Each formative is assigned to various categories that determine its abstract underlying form

Example:
“boy” belongs to the category “noun”, “male”, and “animate”, etc.
This information is presented in a “lexicon”
Surface Structure

How can we generate Surface Structure?

Formatives

A string of minimal elements.
Each formative is assigned to various categories that determine its abstract underlying form

Example:
“boy” belongs to the category “noun”, “male”, and “animate”, etc.
This information is presented in a “lexicon”

Conclusion: Surface structure contains all information of categories (also called “features”)
Surface Structure

How can we represent these formatives?

Analysis of strings into “proper bracketing”

• Let A, B, & C be formatives
Surface Structure

How can we represent these formatives?

Analysis of strings into “proper bracketing”

- Let A, B, & C be formatives
- ABC cannot specify AB as phrase or BC as phrase, because 
  ((AB)C), (A(BC)) and (ABC) are possibilities
Surface Structure

How can we represent these formatives?

Analysis of strings into “proper bracketing”

• Let $A$, $B$, & $C$ be formatives

• $ABC$ cannot specify $AB$ as phrase or $BC$ as phrase, because
  $((AB)C)$, $(A(BC))$ and $(ABC)$ are possibilities

• Phrases can have only one phrase specification at a time
Surface Structure

Phrases are Assigned to Different Categories

*we established telegraphic communication*

- Complete string is assigned to the category “sentence” (S)
Surface Structure

Phrases are Assigned to Different Categories

- Complete string is assigned to the category “sentence” (S)
- Other categories are proved by general linguistic theory:

\[ \text{we established telegraphic communication} \]
Surface Structure

Phrases are Assigned to Different Categories

*we established telegraphic communication*

- Complete string is assigned to the category “sentence” (S)
- Other categories are proved by general linguistic theory:
  - “noun phrase” (NP): phrase which has a noun as its head word
    (e.g.: Books are good)
Phrases are Assigned to Different Categories

**we established telegraphic communication**

- Complete string is assigned to the category “sentence” (S)
- Other categories are proved by general linguistic theory:
  - “noun phrase” (NP): phrase which has a noun as its head word (e.g.: Books are good)
  - “verb phrase” (VP): phrase consisting of at least one verb and its dependence with an object (e.g.: Michel gave Mike a book)
Surface Structure

<table>
<thead>
<tr>
<th>Phrases are Assigned to Different Categories</th>
</tr>
</thead>
<tbody>
<tr>
<td>we established telegraphic communication</td>
</tr>
</tbody>
</table>

- Complete string is assigned to the category “sentence” (S)
- Other categories are proved by general linguistic theory:
  - “noun phrase” (NP): phrase which has a noun as its head word (e.g.: Books are good)
  - “verb phrase” (VP): phrase consisting of at least one verb and its dependence with an object (e.g.: Michel gave Mike a book)

Goal: Finding such general rules that can be applied to any language
Surface Structure

Applying General Rules to Our Example
Surface Structure

Applying General Rules to Our Example

Diagram: A graph illustrating the surface structure of a sentence with labeled parts of speech.
Surface Structure

Equivalent Forms

- Figure beforehand is equivalent to following form:
Surface Structure

Equivalent Forms

- Figure beforehand is equivalent to following form:

\[
\begin{align*}
[S & [NP & [N + we + ]N ]NP [VP [V \langle + establish + \rangle_\text{v} + past + ]_\text{v} \langle NP [A [N + tele + \\
& [STEM + graph + \rangle_\text{stem} ]N + ic + ]_\text{a} \langle [V + communicate + ]_\text{v} + ion + ]_\text{n} ]NP ]VP ]_s
\end{align*}
\]
Surface Structure

Equivalent Forms

- Figure beforehand is equivalent to following form:

  \[
  \]

- Formative *we* is both a **N** and **NP**
Surface Structure

Equivalent Forms

- Figure beforehand is equivalent to following form:

\[
\begin{align*}
\text{S} & \quad \text{NP} \quad \text{[N + we + ]N} \quad \text{NP} \quad \text{VP} \quad \text{[V \ [V + establish + ]V + past + ]V} \quad \text{NP} \quad \text{[A [N + tele + ]N} \quad \text{NP} \quad \text{VP} \quad \text{S} \\
\text{[STEM + graph + ]STEM} & \quad \text{N} \quad \text{ic + ]A} \quad \text{[N [V + communicate + ]V + ion + ]N} \quad \text{NP} \quad \text{VP} \quad \text{S}
\end{align*}
\]

- Formative we is both a N and NP

- “+” symbol represents formative boundaries
Surface Structure

Equivalent Forms

- Figure beforehand is equivalent to following form:

  \[
  S [N_{NP} [N + we + ]_N]_{NP} [V_P [V + establish + ]_V + past + ]_V [NP [A [N + tele + ]_N]_{NP} [V_P [STEM + graph + ]_STEM]_N + ic + ]_A [N [V + communicate + ]_V + ion + ]_N]_{NP} [V_P ]_S
  \]

- Formative we is both a N and NP
- “+” symbol represents formative boundaries
- This kind of bracketing of the surface structure is very important for the transformational cycle
Surface Structure

Equivalent Forms

- Figure beforehand is equivalent to following form:

\[
[S \left[ \text{NP \ N\overline{n}} \right]_{NP} \left[ \text{VP \ V\overline{s}} \right]_{VP} \left[ V \overline{v} \right]_{V} + \text{establish} + ]_{V} + \text{past} + ]_{V} \left[ \text{NP \ A} \right] \left[ \text{N\overline{t}} \overline{e} \overline{l} \right]_{A} + ]_{N} \left[ \text{STEM + graph} + ]_{STEM} \right)_{N} + \text{ic} + ]_{A} \left[ \text{V\overline{v}} + \text{communicate} + ]_{V} + \text{ion} + ]_{N} \left[ \text{NP} \right]_{NP} \left[ \text{VP} \right]_{VP} \]

- Formative \text{we} is both a \text{N} and \text{NP}

- “+” symbol represents formative boundaries

- This kind of bracketing of the surface structure is very important for the transformational cycle

- But first, we’re going to talk about something else important
Surface Structure

Readjustment Rule

Example

\[ \Sigma' := \{established\} \text{ (phonological representation)} \]

Readjustment Rule:

\[ \Sigma := [ V [ V + establish^+ ]_V + past^+ ]_V \]

- Syntactic Component of the grammar generates a surface structure \( \Sigma \)
Surface Structure

Readjustment Rule

Example

\[ \Sigma' := \{established\} \] (phonological representation)

Readjustment Rule:

\[ \Sigma := [V[V + establish+]_V + past+]_V \]

- Syntactic Component of the grammar generates a surface structure \( \Sigma \)
- Perceptual Processing recovers \( \Sigma' \) in order to get to the deep structure, namely \( \Sigma \)
Surface Structure

Readjustment Rule

• What would a readjustment rule need to do in order to replace the following appropriately?

• \([v [v \text{sing}]_v \text{past}]_v \& [v [v \text{mend}]_v \text{past}]_v\)
Surface Structure

Readjustment Rule

• What would a readjustment rule need to do in order to replace the following appropriately?

• \[ v \langle v_{s}ng \rangle_{v} past \rangle_{v} \& \langle v \langle v_{mend} \rangle_{v} past \rangle_{v} \]

• Solution:

• \[ \langle v_{s} * ng \rangle_{v} \& \langle v \langle v_{mend} \rangle_{v} d \rangle_{v} \]
Surface Structure

Readjustment Rule

• What would a readjustment rule need to do in order to replace the following appropriately?

• \([V_{sing}]_V past\) & \([V_{mend}]_V past\)_V

• Solution:

• \([Vs * ng]_V \& [V_{mend}]_V d\)_V

• We’ve applied the readjustment rule on the innermost bracket and removed the innermost bracket afterwards
Surface Structure

Readjustment Rule

• What would a readjustment rule need to do in order to replace the following appropriately?

• \[ [V [Vsing]_V past]_V \ & \ [V [Vmend]_V past]_V \]

• Solution:

• \[ [Vs * ng]_V \ & \ [V [Vmend]_V d]_V \]

• We’ve applied the readjustment rule on the innermost bracket and removed the innermost bracket afterwards

• That’s the basic principle of the transformational cycle
Surface Structure

Relation to Finite State Morphology

- Surface structure (description / upper level):
  \[
  [V [V^{\text{sing}}] V^{\text{past}}]_V
  \]
Surface Structure

Relation to Finite State Morphology

- Surface structure (description / upper level):
  \[ [v [v \text{sing}] v \text{past}] v \]

- Phonological representation (lower level): \[ [v s * ng] v \]
Surface Structure

Relevant Problem: Proper Word Separation
1. Linguistic Universals
   1.1 Motivation for Finite State Automaton
   1.2 What are Linguistic Universals?
   1.3 Grammar of English
   1.4 Theory
   1.5 Surface Structure

2. Transformational Cycle
   2.1 Examples
   2.2 General Principle

3. Transformational Cycle Applied To Finite State Automaton
   3.1 More Examples
   3.2 Transformational Cycle Within the Word
Transformational Cycle

Example

- There are at least two processes of stress assignment in English:
Transformational Cycle

Example

- There are at least two processes of stress assignment in English:
  - $/one.pnum/three.pnum$ & $/two.pnum/one.pnum$ have different surface structures:
  - *blackboard* (contour 13) & *black* *board* (contour 21) have different surface structures:
There are at least two processes of stress assignment in English:

- **blackboard** (contour 13) & **black board** (contour 21) have different surface structures:

```
  1  3
  \__________\     \__________\
  |          |     |          |
  \__________\     \__________\
  |          |     |          |
  \__________\     \__________\
  |          |     |          |
  \__________\     \__________\
  |          |     |          |
  \__________\     \__________\
  |          |     |          |
  \__________\     \__________\
  |          |     |          |
  \__________\     \__________\
  |          |     |          |
  \__________\     \__________\
  |          |     |          |
  \__________\     \__________\
  |          |     |          |
  \__________\     \__________\
  |          |     |          |
  \__________\     \__________\
  |          |     |          |
  \__________\     \__________\
  |          |     |          |
  \__________\     \__________\
```
There are at least two processes of stress assignment in English:

- blackboard (contour 13) & black board (contour 21) have different surface structures:

How were the stress contours applied?
Transformational Cycle

Example

- First look at lowest level (A, N)
Transformational Cycle

Example

- First look at lowest level (A, N)
- In monosyllables, the vowel receives primary stress (1)
Transformational Cycle

Example

- First look at lowest level (A, N)
- In monosyllables, the vowel receives primary stress (1)
- Afterwards, the innermost bracket is removed:
Transformational Cycle

Example

- First look at lowest level (A, N)
- In monosyllables, the vowel receives primary stress (1)
- Afterwards, the innermost bracket is removed:
Transformational Cycle

Example

- When primary stress is placed in a certain position, then all other stresses in the string are automatically weakened by one
Transformational Cycle

Example

- When primary stress is placed in a certain position, then all other stresses in the string are automatically weakened by one.
- Compound Rule: Assign primary stress to a primary-stressed vowel in the context

\[\overline{\ldots \hat{V} \ldots} \]

 dash indicates the position of the segment where the rule is applied.
Transformational Cycle

Example

- When primary stress is placed in a certain position, then all other stresses in the string are automatically weakened by one.

- **Compound Rule:** Assign primary stress to a primary-stressed vowel in the context.

- **Nuclear Stress Rule:** Assign primary stress to a primary-stressed vowel in the context.
Transformational Cycle

Example

- When primary stress is placed in a certain position, then all other stresses in the string are automatically weakened by one.
- **Compound Rule:** Assign primary stress to a primary-stressed vowel in the context.
- **Nuclear Stress Rule:** Assign primary stress to a primary-stressed vowel in the context.
- $\overline{V}_{\text{primary}}$: stands for vowel with primary stress.
Transformational Cycle

*Example*

- When primary stress is placed in a certain position, then all other stresses in the string are automatically weakened by one.
- **Compound Rule:** Assign primary stress to a primary-stressed vowel in the context.
- **Nuclear Stress Rule:** Assign primary stress to a primary-stressed vowel in the context.
- \( V \): stands for vowel with primary stress.
- Dash indicates the position of the segment where the rule is applied.
Transformational Cycle

Example

- What rule do we need to apply?
Transformational Cycle
Example

- What rule do we need to apply?
- Applying Compound Rule:

\[ [N \# \# \text{black} \# \# \text{board} \# \#]_N \]
Transformational Cycle

Example

• What rule do we need to apply?
• Applying Compound Rule:
Transformational Cycle

Example

- What rule do we need to apply?
- Applying Compound Rule:
  
  $$[N\#\#black\#\#board\#\#]_N$$
  
  $$\ldots\ldots\ldots\sqrt{\ldots}_N$$
  
  $$\#\#black\#\#board\#\#$$
Transformational Cycle

Example

- However, we are not yet finished, since the desired outcome is \( \text{contour} = 13 \), thus we need to apply another rule:
Transformational Cycle

Example

- However, we are not yet finished, since the desired outcome is contour = 13, thus we need to apply another rule:

  $$\cdots \# \# C_0 \overset{2}{\nabla} C_0 \# \downarrow N$$
Transformational Cycle

Example

• However, we are not yet finished, since the desired outcome is contour = 13, thus we need to apply another rule:

  \[ \cdots \# \# C_0 \check{\gamma} C_0 \# \]_N

• Where $C_0$ is a string of zero or more consonants
Transformational Cycle

Example

- However, we are not yet finished, since the desired outcome is contour = 13, thus we need to apply another rule:

  \[ \cdots \# \# C_0 \bar{V} C_0 \# \]_N

- Where \( C_0 \) is a string of zero or more consonants

- Thus we get the desired outcome: contour = 13
Transformational Cycle

Example

- What rule do we need to apply?
Transformational Cycle

Example

- What rule do we need to apply?
- Nuclear Stress Rule:
Transformational Cycle

Example

- What rule do we need to apply?
- Nuclear Stress Rule:
Transformational Cycle

Example

- What rule do we need to apply?
- Nuclear Stress Rule:
  
  \[
  \left[ \text{NP} \right. \, # \, # \text{black} \, # \, # \text{board} \, # \, # \left. \text{NP} \right]
  \]
  
  \[
  ^1 \text{V} \ldots \text{---} \ldots ^1 \text{NP}
  \]
  
  \[
  # \, # \text{black} \, # \, # \text{board} \, # \, #
  \]
Transformation Cycle

- Compound Rule & Nuclear Stress Rule can be more generalized
Transformation Cycle

- Compound Rule & Nuclear Stress Rule can be more generalized
- Compound Rule cannot only be used for “blackboard” (compound noun), but also for “heart-broken” (compound adjective), and “air-condition” (compound verb)
Transformation Cycle

- Compound Rule & Nuclear Stress Rule can be more generalized
- Compound Rule cannot only be used for “blackboard” (compound noun), but also for “heart-broken” (compound adjective), and “air-condition” (compound verb)
- Nuclear Stress Rule doesn’t only apply to “black board” (noun phrase), but also to “read the book” (verb phrases), “eager to please” (adjective phrase) and to whole sentences: “John left”
Transformation Cycle

- Compound Rule & Nuclear Stress Rule can be more generalized.
- Compound Rule cannot only be used for “blackboard” (compound noun), but also for “heart-broken” (compound adjective), and “air-condition” (compound verb).
- Nuclear Stress Rule doesn’t only apply to “black board” (noun phrase), but also to “read the book” (verb phrases), “eager to please” (adjective phrase) and to whole sentences: “John left.”
- We can state following new rules:
Assign primary stress to a primary-stressed vowel in the context

\[ \text{--- \ldots } \hat{V} \ldots \text{]}_{\text{NAV}} \]

Assign primary stress to a primary-stressed vowel in the context

\[ \hat{V} \ldots \text{--- \ldots } \]

Where NAV stands for a bracket with all labels like N, A, or V and \([ \ ]_\alpha\) stands for anything else.
Assign primary stress to a primary-stressed vowel in the context

\[ \ldots \hat{V} \ldots \]_{\text{NAV}}

Assign primary stress to a primary-stressed vowel in the context

\[ \hat{V} \ldots \ldots \ldots \]_{\alpha}

Where NAV stands for a bracket with all labels like N, A, or V and \([ ]_{\alpha}\) stands for anything else.

Another notation may be:

\[
\begin{pmatrix}
1 \text{ stress} \\
V
\end{pmatrix} \rightarrow [1 \text{ stress}] \div \left( \frac{\hat{V} \ldots \ldots \ldots}{\hat{V} \ldots \ldots \ldots} \right)_{\text{NAV}}
\]

(a) COMPOUND RULE

(b) NUCLEAR STRESS RULE
Transformational Cycle

General Principle

- First of all, all brackets and formants need to be created as shown before within the introduction
Transformational Cycle

General Principle

- First of all, all brackets and formants need to be created as shown before within the introduction
- Starting from the innermost bracket, apply rules and erase brackets
Transformational Cycle

General Principle

- First of all, all brackets and formants need to be created as shown before within the introduction.
- Starting from the innermost bracket, apply rules and erase brackets.
- Move recursively forward until there are no more brackets left.
1. Linguistic Universals
   1.1 Motivation for Finite State Automaton
   1.2 What are Linguistic Universals?
   1.3 Grammar of English
   1.4 Theory
   1.5 Surface Structure

2. Transformational Cycle
   2.1 Examples
   2.2 General Principle

3. Transformational Cycle Applied To Finite State Automaton
   3.1 More Examples
   3.2 Transformational Cycle Within the Word
### Transformational Cycle

#### More Complex Example

<table>
<thead>
<tr>
<th>Description</th>
<th>Contours</th>
</tr>
</thead>
<tbody>
<tr>
<td>black board-eraser</td>
<td>213</td>
</tr>
<tr>
<td>“board eraser that is black”</td>
<td></td>
</tr>
<tr>
<td>blackboard eraser</td>
<td>132</td>
</tr>
<tr>
<td>“eraser for a blackboard”</td>
<td></td>
</tr>
<tr>
<td>black board eraser</td>
<td>312</td>
</tr>
<tr>
<td>“eraser of a black board”</td>
<td></td>
</tr>
</tbody>
</table>
Transformational Cycle

More Complex Example

<table>
<thead>
<tr>
<th>Contours</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>black board-eraser</td>
<td>213</td>
</tr>
<tr>
<td>“board eraser that is black”</td>
<td></td>
</tr>
<tr>
<td>blackboard eraser</td>
<td>132</td>
</tr>
<tr>
<td>“eraser for a blackbaord”</td>
<td></td>
</tr>
<tr>
<td>black board eraser</td>
<td>312</td>
</tr>
<tr>
<td>“eraser of a black board”</td>
<td></td>
</tr>
</tbody>
</table>

How do we receive the appropriate contours by using the transformational cycle?
Transformational Cycle

More Complex Example

1.)

\[
\begin{array}{c}
\left[ \begin{array}{c}
1 \text{ stress} \\
V
\end{array} \right] \rightarrow \left[ \begin{array}{c}
1 \text{ stress} \\
V
\end{array} \right] / \left\{ \frac{1}{V} \ldots \right\}_{NAV}
\end{array}
\]

(a) Compound Rule

(b) Nuclear Stress Rule

\[
[_{NP} [_{A} \text{black}]_A [_{N} \text{board}]_N [_{N} \text{eraser}]_N ]_N ]_{NP}
\]
Transformational Cycle

More Complex Example

1.)

\[
\begin{array}{c}
\left[ \frac{1 \text{ stress}}{V} \right] \rightarrow \left[ \frac{1 \text{ stress}}{V} \right] \bigg/ \left( \frac{1}{V} \ldots \frac{1}{V} \ldots \right)_{\text{NAV}} \bigg) \\
\end{array}
\]

(a) COMPOUND RULE

(b) NUCLEAR STRESS RULE

\[
\begin{array}{c}
\begin{array}{c}
[\text{NP} \left[ \frac{\text{black}}{A} \right] \text{A} \left[ \frac{\text{board}}{N} \right] \text{N} \left[ \frac{\text{eraser}}{N} \right] \text{N} ] \text{N} ] \text{NP} \\
\end{array}
\end{array}
\]

1 1

1 1

1 2

2 1 3
Transformational Cycle

More Complex Example

2.)

\[
\begin{bmatrix}
1 \text{ stress} \\
V
\end{bmatrix} \rightarrow \left[ \text{1 stress} \right] \div \left( \frac{1}{V} \ldots \frac{1}{V} \ldots \right)_{NAV}
\]

(a) Compound Rule
(b) Nuclear Stress Rule

\[
\left[ N \left[ N \left[ \text{black} \right]_{A} \left[ \text{board} \right]_{N} \right]_{N} \left[ \text{eraser} \right]_{N} \right]_{N}
\]
Transformational Cycle

More Complex Example

2.

\[
\begin{bmatrix}
1 \\
V
\end{bmatrix}
\rightarrow
\begin{bmatrix}
1 \\
\text{stress}
\end{bmatrix}
\text{/}
\left\{\frac{1}{V} \ldots V \ldots \right\}_{NAV}
\]

(a) COMPOUND RULE
(b) NUCLEAR STRESS RULE
Transformational Cycle

More Complex Example

3.)

\[
\begin{align*}
\left[ 1 \text{ stress} \right] & \rightarrow \left[ 1 \text{ stress} \right] / \left\{ \frac{1}{\text{NAV}} \cdots \frac{1}{\text{NAV}} \cdots \right\} \\
& (a) \text{ Compound Rule} \\
& (b) \text{ Nuclear Stress Rule}
\end{align*}
\]

\[
[ N \left[ NP \left[ A \text{black} \right] A \left[ N \text{board} \right] N \right] NP \left[ N \text{eraser} \right] N ] N
\]
Transformational Cycle

More Complex Example

3.)

\[
\begin{bmatrix}
1 \text{ stress} \\
\downarrow
\end{bmatrix} \rightarrow \begin{bmatrix}
1 \text{ stress}
\end{bmatrix} \div \left\{ \begin{array}{c}
\frac{1}{V} \\
\cdots
\frac{1}{V} \\
\cdots
\end{array} \right\}_{\text{NAV}}
\]

(a) COMPOUND RULE
(b) NUCLEAR STRESS RULE
Transformational Cycle

More Complex Example

\[
\begin{bmatrix}
\text{1 stress} \\
V
\end{bmatrix} \rightarrow [1 \text{ stress}] / \left\{ \left( \frac{1}{V} \cdots \frac{1}{\cdots} \right)_{\text{NAV}} \right\}
\]

(a) COMPOUND RULE

(b) NUCLEAR STRESS RULE

Please, do this on your own:

\[
[\text{NP} \ [dJohn's]_d \ [N \ [a black]_a \ [N board]_N \ [N eraser]_N \ ]_N \ ]_N \ ]_NP
\]
Transformational Cycle

More Complex Example

Please, do this on your own:
Transformational Cycle

More Complex Example

- However, the Transformational Cycle doesn’t seem to work indefinitely, especially for very long examples:
Transformational Cycle

More Complex Example

- However, the Transformational Cycle doesn’t seem to work indefinitely, especially for very long examples:
- my friend can’t help being shocked at anyone who would fail to consider his sad plight
Transformational Cycle

More Complex Example

- However, the Transformational Cycle doesn’t seem to work indefinitely, especially for very long examples:
- my friend can’t help being shocked at anyone who would fail to consider his sad plight
- Numerously applied Nuclear Stress Rules will return: ...
  sad, plight, which isn’t correct
Transformational Cycle

More Complex Example

• However, the Transformational Cycle doesn’t seem to work indefinitely, especially for very long examples:

• my friend can’t help being shocked at anyone who would fail to consider his sad plight

• Numerously applied Nuclear Stress Rules will return: ...

• Restrictions or readjustment rules will need to made in order to receive correct output
Realization

However, the **Compound Rule** as well as the **Nuclear Stress Rule** still show, that you can generate an amazing number of correct output with just two simple rules.
Transformational Cycle

More Complex Example

Realization

However, the Compound Rule as well as the Nuclear Stress Rule still show, that you can generate an amazing number of correct output with just two simple rules.

How can we relate this to Finite State Morphology?
Transformational Cycle Within the Word

Morphology

• In our previous examples we ignored that non-monosyllabic words also existed within the example sentences
Transformational Cycle Within the Word

Morphology

- In our previous examples we ignored that non-monosyllabic words also existed within the example sentences
- Let’s look at the word “eraser” and its new structure:
Transformational Cycle Within the Word

*Morphology*

- In our previous examples we ignored that non-monosyllabic words also existed within the example sentences
- Let’s look at the word “eraser” and its new structure:

\[
\text{[N} \# \text{[v} \# \text{erase} \#] \text{v} \ r \ # \text{]}_N
\]
Transformational Cycle Within the Word

Morphology

- In our previous examples we ignored that non-monosyllabic words also existed within the example sentences.
- Let's look at the word “eraser” and its new structure:
  \[ N \# [v \# erase \#]_v r \# \]_N
- The word itself will also go through transformational cycles.
In our previous examples we ignored that non-monosyllabic words also existed within the example sentences.

Let’s look at the word “eraser” and its new structure:

\[ [N \# [V \# erase \#]_V r \# ]_N \]

The word itself will also go through transformational cycles.

What would happen, if we would proceed as beforehand?
Transformational Cycle Within the Word

Example

- Stress would be placed on the second syllable
Transformational Cycle Within the Word

*Example*

- Stress would be placed on the second syllabys
- Thus, a *new rule* needs to be formulated:
Transformational Cycle Within the Word

Example

- Stress would be placed on the second syllabys
- Thus, a **new rule** needs to be formulated:

\[
V \rightarrow \text{[1 stress]} \quad \text{/} \quad X \quad \text{---} \quad C_0]_{NAV} \quad (25)
\]
Transformational Cycle Within the Word

Example

- Stress would be placed on the second syllabys

- Thus, a **new rule** needs to be formulated:

\[
V \rightarrow [\text{1 stress}] \quad / \quad X \quad \quad C_{0} \quad \text{NAV}_{\text{AV}}
\] (25)

- That rule wouldn’t work on words like: “blackboard”, “blackboard eraser”, etc.
Transformational Cycle Within the Word

Example

- Stress would be placed on the second syllabys
- Thus, a new rule needs to be formulated:
  \[
  V \rightarrow [1 \text{ stress}] / X \quad C_0 \]_{NAV}\]
  \[ (25) \]
- That rule wouldn’t work on words like: “blackboard”, “blackboard eraser”, etc.
- How could we restrain the general rule?
Transformational Cycle Within the Word

Example

- Stress would be placed on the second syllabys
- Thus, a new rule needs to be formulated:
  \[ V \rightarrow [1 \text{ stress}] / X \longrightarrow C_0 ]_{NAV} \] (25)
- That rule wouldn’t work on words like: “blackboard”, “blackboard eraser”, etc.
- How could we restrain the general rule?
  \[ X \text{ contains no internal occurrence of } \# . \]
Transformational Cycle Within the Word

Example

- Stress would be placed on the second syllabys
- Thus, a **new rule** needs to be formulated:
  \[ V \to [1 \text{ stress}] / X \xrightarrow{C_0} ]_{NAV} \]  
  (25)
- That rule wouldn’t work on words like: “blackboard”, “blackboard eraser”, etc.
- How could we restrain the general rule?
  \[ X \text{ contains no internal occurrence of } \#. \]
- Rule (25) can be dismissed in favor for the new, more general rules:
Transformational Cycle Within the Word

(a) \[ V \rightarrow [1 \text{ stress}] / X \rightarrow C_0(W) + \text{affix} \]
(b) \[ V \rightarrow [1 \text{ stress}] / X \rightarrow C_0(W) \]

Clearly there is a generalization being missed by the formulation (29), for the obvious similarity between the two cases is not expressed. To permit us to capture generalizations of this sort, we extend our notations to permit rules such as (30):

\[ (30) \quad X \rightarrow Y / Z \rightarrow R / P \rightarrow Q \]

In general, a rule of the form (31) can be regarded as an abbreviation for the rule (32), where \( Z \) and \( R \) are strings:

\[ (31) \quad X \rightarrow Y / Z \rightarrow R \]

\[ (32) \quad ZXR \rightarrow ZYR \]

Following this convention, we interpret (30) as an abbreviation for (33), where \( Z \) and \( R \) are strings:

\[ (33) \quad ZXR \rightarrow ZYR / P \rightarrow Q \]

This is now a rule of a familiar form. Reapplying the convention that defines (31) in terms of (32), we interpret (33) as an abbreviation for (34):

\[ (34) \quad PZXRQ \rightarrow PZYRQ \]
Questions

1.) Do you think that the transformational cycle could be applied to Finite State Morphology?
Transformational Cycle Within the Word

Questions

• 1.) Do you think that the transformational cycle could be applied to Finite State Morphology?
• 2.) Can we build an automaton that can apply the Compound Rule and the Nuclear Stress Rule?
Thank you for your attention!

)