Non-Classical Logics for Natural Language: Introduction to Substructural Logics

Raffaella Bernardi

KRDB, Free University of Bozen-Bolzano

e-mail: bernardi@inf.unibz.it

Contents

1. Course Overview

- **Today**:

  1. `(Formal) Linguistic Background`. First of all, we introduce the concept of Formal Grammar for Natural Languages by presenting some linguistic background and the **challenges** such a grammar should face. Furthermore, we motivate the use of a **Logical Grammar** to address such challenges.
  2. `Introduction to Substructural Logics`. First of all, we will introduce **non-classical logics** by underlining the differences with respect to classical logics. Then we move to introduce **substructural logics** and we will briefly look at some well known of them – Relevant Logics, Linear Logic... Then, we will focus attention on a sub-family of Substructural Logics known as **Lambek Calculi** or Logics of Residuation.

- **Tomorrow**:

  1. `Lambek Calculus`. We start presenting both the Model Theoretical and Proof Theoretical aspects of the Lambek Calculi. Then, we look at its application to natural language parsing.
  2. `Syntax-Semantics Interface.` In the second part, we consider the application of the Lambek Calculi and the Lambda calculus to natural language analysis. We start by introducing some background notions of **Formal Semantics** focusing on the set-theoretical perspective first, and then on the corresponding functional perspective. Then we will show how the Lambek Calculi account for the composition of linguistic resources while simultaneously allowing parsing and the construction of meaning.

2. Linguistics Background

Formal Linguistics is the study of natural language. Formal Linguists aim to

- formally define **grammaticality** of sentences
- understand how syntactic structures are built
- formally define the **meaning** of sentences
- understand how semantic structures are built

2. Linguistics Background

Formal Linguistics is the study of natural language. Formal Linguists aim to

- formally define **grammaticality** of sentences
- understand how syntactic structures are built
- formally define the **meaning** of sentences
- understand how semantic structures are built
- model **syntax-semantic** interface

2. Linguistics Background

Formal Linguistics is the study of natural language. Formal Linguists aim to

- formally define **grammaticality** of sentences
- understand how syntactic structures are built
- formally define the **meaning** of sentences
- understand how semantic structures are built
- model **syntax-semantic** interface
- find the universal **core** of all natural languages
- find natural language variations

2.1. Syntax: Preliminary Notions

- **Syntax**: "setting out things together", in our case things are words. The main question addressed here is "*How do words compose together to form a grammatical sentence (s) (or fragments of it)?*"

2.1. Syntax: Preliminary Notions

- **Syntax**: "setting out things together", in our case things are words. The main question addressed here is "*How do words compose together to form a grammatical sentence (s) (or fragments of it)?*"
- **Categories**: words are said to belong to *classes*/categories. The main categories are nouns (n), verbs (v), adjectives (adj), determiners (det) and adverbs (adv).

2.1. Syntax: Preliminary Notions

- **Syntax**: "setting out things together", in our case things are words. The main question addressed here is "*How do words compose together to form a grammatical sentence (s) (or fragments of it)?*"
- **Categories**: words are said to belong to *classes*/categories. The main categories are nouns (*n*), verbs (*v*), adjectives (*adj*), determiners (*det*) and adverbs (*adv*).
- **Constituents**: Groups of categories may form a single *unit or phrase* called constituents. The main phrases are noun phrases (*np*), verb phrases (*vp*), prepositional phrases (*pp*). Noun phrases for instance are: "she"; "Michael"; "Rajeev Goré"; "the house"; "a young two-year child".

2.1. Syntax: Preliminary Notions

- **Syntax**: "setting out things together", in our case things are words. The main question addressed here is "*How do words compose together to form a grammatical sentence (s) (or fragments of it)?*"
- **Categories**: words are said to belong to *classes*/categories. The main categories are nouns (n), verbs (v), adjectives (adj), determiners (det) and adverbs (adv).
- **Constituents**: Groups of categories may form a single *unit or phrase* called constituents. The main phrases are noun phrases (np), verb phrases (vp), prepositional phrases (pp). Noun phrases for instance are: "she"; "Michael"; "Rajeev Goré"; "the house"; "a young two-year child".

  Tests like substitution help decide whether words form constituents.

2.1. Syntax: Preliminary Notions

- **Syntax**: "setting out things together", in our case things are words. The main question addressed here is "*How do words compose together to form a grammatical sentence (s) (or fragments of it)?*"

- **Categories**: words are said to belong to *classes*/categories. The main categories are nouns (*n*), verbs (*v*), adjectives (*adj*), determiners (*det*) and adverbs (*adv*).

- **Constituents**: Groups of categories may form a single *unit or phrase* called constituents. The main phrases are noun phrases (*np*), verb phrases (*vp*), prepositional phrases (*pp*). Noun phrases for instance are: "she"; "Michael"; "Rajeev Goré"; "the house"; "a young two-year child".

  Tests like substitution help decide whether words form constituents.

- **Dependency**: Categories are interdependent, for example

Ryanair services $[\text{Pescara}]_{np}$	Ryanair flies $[\text{to Pescara}]_{pp}$
*Ryanair services $[\text{to Pescara}]_{pp}$	*Ryanair flies $[\text{Pescara}]_{np}$

  the verbs services and flies determine which category can/must be juxtaposed. If their constraints are not satisfied the structure is ungrammatical.

2.2. Long-distance Dependencies

Interdependent constituents need not be juxtaposed, but may form long-distance dependencies, manifested by gaps

- What cities does Ryanair service [. . .]?

2.2. Long-distance Dependencies

Interdependent constituents need not be juxtaposed, but may form long-distance dependencies, manifested by gaps

- What cities does Ryanair service [...]?

The constituent what cities depends on the verb service, but is at the front of the sentence rather than at the object position.

2.2. Long-distance Dependencies

Interdependent constituents need not be juxtaposed, but may form long-distance dependencies, manifested by gaps

- What cities does Ryanair service [. . .]?

The constituent what cities depends on the verb service, but is at the front of the sentence rather than at the object position.

Such distance can be large,

- Which flight do you want me to book [. . .]?

2.2. Long-distance Dependencies

Interdependent constituents need not be juxtaposed, but may form long-distance dependencies, manifested by gaps

- What cities does Ryanair service [. . .]?

The constituent what cities depends on the verb service, but is at the front of the sentence rather than at the object position.

Such distance can be large,

- Which flight do you want me to book [. . .]?
- Which flight do you want me to have the travel agent book [. . .]?

2.2. Long-distance Dependencies

Interdependent constituents need not be juxtaposed, but may form long-distance dependencies, manifested by gaps

- What cities does Ryanair service [...]?

The constituent what cities depends on the verb service, but is at the front of the sentence rather than at the object position.

Such distance can be large,

- Which flight do you want me to book [...]?
- Which flight do you want me to have the travel agent book [...]?
- Which flight do you want me to have the travel agent nearby my office book [...]?

2.3. Relative Pronouns and Coordination

- **Relative Pronoun** (eg. who, which): they function as e.g. the subject or object of the verb embedded in the relative clause (*rc*),
  - [[the [student [who [. . .] knows Sara]$_{rc}$]$_{n}$]$_{np}$ [left]$_{v}$]$_{s}$.

2.3. Relative Pronouns and Coordination

- **Relative Pronoun** (eg. who, which): they function as e.g. the subject or object of the verb embedded in the relative clause (rc),
  - [[the [student [who [...] knows Sara]$_{rc}$]$_{n}$]$_{np}$ [left]$_{v}$]$_{s}$.
  - [[the [book [which Sara wrote [...]]$_{rc}$]$_{n}$]$_{np}$ [is interesting]$_{v}$]$_{s}$.

2.3. Relative Pronouns and Coordination

- **Relative Pronoun** (eg. who, which): they function as e.g. the subject or object of the verb embedded in the relative clause (rc),
  - [[the [student [who [...] knows Sara]$_{rc}$]$_{n}$]$_{np}$ [left]$_{v}$]$_{s}$.
  - [[the [book [which Sara wrote [...]]$_{rc}$]$_{n}$]$_{np}$ [is interesting]$_{v}$]$_{s}$.
- **Coordination**: Expressions of the same syntactic category can be coordinated via "and", "or", "but" to form more complex phrases of the same category. For instance, a coordinated verb phrase can consist of two other verb phrases separated by a conjunction:

2.3. Relative Pronouns and Coordination

- **Relative Pronoun** (eg. who, which): they function as e.g. the subject or object of the verb embedded in the relative clause (rc),
  - [[the [student [who [. . .] knows Sara]$_{rc}$]$_{n}$]$_{np}$ [left]$_{v}$]$_{s}$.
  - [[the [book [which Sara wrote [. . .]]$_{rc}$]$_{n}$]$_{np}$ [is interesting]$_{v}$]$_{s}$.
- **Coordination**: Expressions of the same syntactic category can be coordinated via "and", "or", "but" to form more complex phrases of the same category. For instance, a coordinated verb phrase can consist of two other verb phrases separated by a conjunction:
  - There are no flights [[leaving Denver]$_{vp}$ and [arriving in San Francisco]$_{vp}$]$_{vp}$

2.3. Relative Pronouns and Coordination

- **Relative Pronoun** (eg. who, which): they function as e.g. the subject or object of the verb embedded in the relative clause (rc),
  - [[the [student [who [...] knows Sara]$_{rc}$]$_{n}$]$_{np}$ [left]$_{v}$]$_{s}$.
  - [[the [book [which Sara wrote [...]]$_{rc}$]$_{n}$]$_{np}$ [is interesting]$_{v}$]$_{s}$.
- **Coordination**: Expressions of the same syntactic category can be coordinated via "and", "or", "but" to form more complex phrases of the same category. For instance, a coordinated verb phrase can consist of two other verb phrases separated by a conjunction:
  - There are no flights [[leaving Denver]$_{vp}$ and [arriving in San Francisco]$_{vp}$]$_{vp}$

  The conjuncted expressions belong to traditional constituent classes, vp. However, we could also have

2.3. Relative Pronouns and Coordination

- **Relative Pronoun** (eg. who, which): they function as e.g. the subject or object of the verb embedded in the relative clause (rc),
  - [[the [student [who [. . .] knows Sara]$_{rc}$]$_{n}$]$_{np}$ [left]$_{v}$]$_{s}$.
  - [[the [book [which Sara wrote [. . .]]$_{rc}$]$_{n}$]$_{np}$ [is interesting]$_{v}$]$_{s}$.

- **Coordination**: Expressions of the same syntactic category can be coordinated via "and", "or", "but" to form more complex phrases of the same category. For instance, a coordinated verb phrase can consist of two other verb phrases separated by a conjunction:
  - There are no flights [[leaving Denver]$_{vp}$ and [arriving in San Francisco]$_{vp}$]$_{vp}$

  The conjuncted expressions belong to traditional constituent classes, vp. However, we could also have

  - I [[[want to try to write [. . .]] and [hope to see produced [. . .]]] [the movie]$_{np}$]$_{vp}$"

  Again, the interdependent constituents are disconnected from each other.

2.4. Ambiguity

- **Lexical Ambiguity**: a single word can have more than one syntactic category; for example, “smoke” can be a noun or a verb, “her” can be a pronoun or a possessive determiner.

2.4. Ambiguity

- **Lexical Ambiguity**: a single word can have more than one syntactic category; for example, “smoke” can be a noun or a verb, “her” can be a pronoun or a possessive determiner.
- **Structural Ambiguity**: there are a few valid tree forms for a single sequence of words; for example, which are the possible structures for “old men and women”?

2.4. Ambiguity

- **Lexical Ambiguity**: a single word can have more than one syntactic category; for example, "smoke" can be a noun or a verb, "her" can be a pronoun or a possessive determiner.
- **Structural Ambiguity**: there are a few valid tree forms for a single sequence of words; for example, which are the possible structures for "old men and women"?

  (a) [[old men]and women] or

2.4. Ambiguity

- **Lexical Ambiguity**: a single word can have more than one syntactic category; for example, "smoke" can be a noun or a verb, "her" can be a pronoun or a possessive determiner.
- **Structural Ambiguity**: there are a few valid tree forms for a single sequence of words; for example, which are the possible structures for "old men and women"?

  (a) [[old men]and women] or

  (b) [old[men and women]].

2.4. Ambiguity

- **Lexical Ambiguity**: a single word can have more than one syntactic category; for example, "smoke" can be a noun or a verb, "her" can be a pronoun or a possessive determiner.
- **Structural Ambiguity**: there are a few valid tree forms for a single sequence of words; for example, which are the possible structures for "old men and women"?

  (a) [[old men]and women] or

  (b) [old[men and women]].

- **Mismatch between syntax and semantics** (QPs: non local scope construals): [Alice [thinks [someone left]$_s$]$_{vp}$]$_s$

2.4. Ambiguity

- **Lexical Ambiguity**: a single word can have more than one syntactic category; for example, "smoke" can be a noun or a verb, "her" can be a pronoun or a possessive determiner.

- **Structural Ambiguity**: there are a few valid tree forms for a single sequence of words; for example, which are the possible structures for "old men and women"?

  (a) [[old men]and women] or

  (b) [old[men and women]].

- **Mismatch between syntax and semantics** (QPs: non local scope construals): [Alice [thinks [someone left]$_s$]$_{vp}$]$_s$

(a1) $\texttt{Think}(\texttt{alice}, \exists x(\texttt{left}(x)))$

2.4. Ambiguity

- **Lexical Ambiguity**: a single word can have more than one syntactic category; for example, "smoke" can be a noun or a verb, "her" can be a pronoun or a possessive determiner.
- **Structural Ambiguity**: there are a few valid tree forms for a single sequence of words; for example, which are the possible structures for "old men and women"?

  (a) [[old men]and women] or

  (b) [old[men and women]].

- **Mismatch between syntax and semantics** (QPs: non local scope construals): [Alice [thinks [someone left]$_s$]$_{vp}$]$_s$

(a1) $\texttt{Think}(\texttt{alice}, \exists x(\texttt{left}(x)))$

(a2) $\exists x(\texttt{Think}(\texttt{alice}, \texttt{left}(x)))$

3. Formal Linguistics

Given a linguistic input, we want to use a formal device to:

3. Formal Linguistics

Given a linguistic input, we want to use a formal device to:

- recognize whether it is grammatical.

3. Formal Linguistics

Given a linguistic input, we want to use a formal device to:

- recognize whether it is grammatical.
- give its syntactic structure.

3. Formal Linguistics

Given a linguistic input, we want to use a formal device to:

- recognize whether it is grammatical.
- give its syntactic structure.
- build its meaning representation.

3. Formal Linguistics

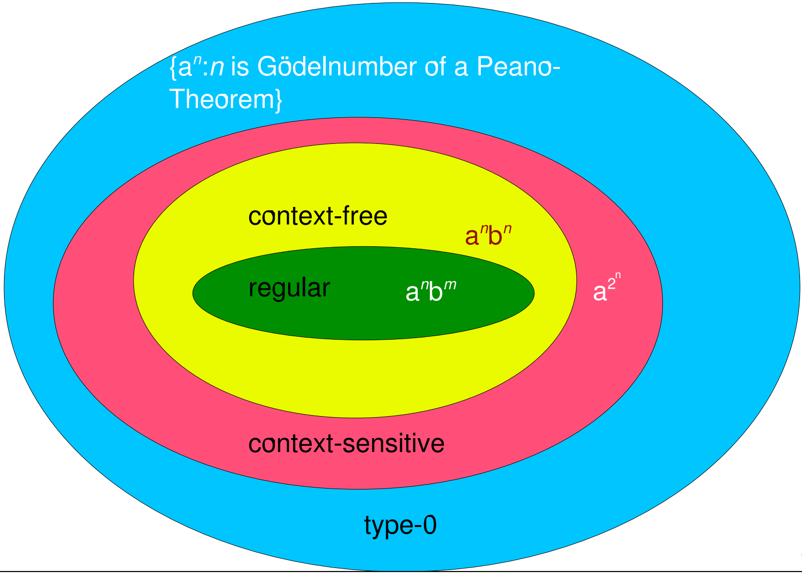
Given a linguistic input, we want to use a formal device to:

- recognize whether it is grammatical.
- give its syntactic structure.
- build its meaning representation.

We look at natural language as a formal language and use formal grammars to achieve these goals.

3.1. Chomsky Hierarchy of Languages

The Chomsky Hierarchy



3.2. Where do Natural Languages fit?

The crucial information to answer this question is which kind of dependencies are found in NLs.

3.2. Where do Natural Languages fit?

The crucial information to answer this question is which kind of dependencies are found in NLs.

- Chomsky (1956, 1957) showed that NLs are not Regular Languages (examples).

3.2. Where do Natural Languages fit?

The crucial information to answer this question is which kind of dependencies are found in NLs.

- Chomsky (1956, 1957) showed that NLs are not Regular Languages (examples).
- Are NLs CFL?

3.2. Where do Natural Languages fit?

The crucial information to answer this question is which kind of dependencies are found in NLs.

- Chomsky (1956, 1957) showed that NLs are not Regular Languages (examples).
- Are NLs CFL?

    1. Chomsky 1957: conjecture that natural languages are not CF

3.2. Where do Natural Languages fit?

The crucial information to answer this question is which kind of dependencies are found in NLs.

- Chomsky (1956, 1957) showed that NLs are not Regular Languages (examples).
- Are NLs CFL?

  1. Chomsky 1957: conjecture that natural languages are not CF
  2. sixties, seventies: many attempts to prove this conjecture

3.2. Where do Natural Languages fit?

The crucial information to answer this question is which kind of dependencies are found in NLs.

- Chomsky (1956, 1957) showed that NLs are not Regular Languages (examples).
- Are NLs CFL?

  1. Chomsky 1957: conjecture that natural languages are not CF
  2. sixties, seventies: many attempts to prove this conjecture
  3. Pullum and Gazdar 1982:

3.2. Where do Natural Languages fit?

The crucial information to answer this question is which kind of dependencies are found in NLs.

- Chomsky (1956, 1957) showed that NLs are not Regular Languages (examples).
- Are NLs CFL?
  1. Chomsky 1957: conjecture that natural languages are not CF
  2. sixties, seventies: many attempts to prove this conjecture
  3. Pullum and Gazdar 1982:
     - all these attempts have failed

3.2. Where do Natural Languages fit?

The crucial information to answer this question is which kind of dependencies are found in NLs.

- Chomsky (1956, 1957) showed that NLs are not Regular Languages (examples).
- Are NLs CFL?
  1. Chomsky 1957: conjecture that natural languages are not CF
  2. sixties, seventies: many attempts to prove this conjecture
  3. Pullum and Gazdar 1982:
     - all these attempts have failed
     - for all we know, natural languages (conceived as string sets) might be context-free

3.2. Where do Natural Languages fit?

The crucial information to answer this question is which kind of dependencies are found in NLs.

- Chomsky (1956, 1957) showed that NLs are not Regular Languages (examples).
- Are NLs CFL?
  1. Chomsky 1957: conjecture that natural languages are not CF
  2. sixties, seventies: many attempts to prove this conjecture
  3. Pullum and Gazdar 1982:
     - all these attempts have failed
     - for all we know, natural languages (conceived as string sets) might be context-free
  4. Huybregts 1984, Shieber 1985: proof that Swiss German is not context-free

3.2. Where do Natural Languages fit?

The crucial information to answer this question is which kind of dependencies are found in NLs.

- Chomsky (1956, 1957) showed that NLs are not Regular Languages (examples).
- Are NLs CFL?
  1. Chomsky 1957: conjecture that natural languages are not CF
  2. sixties, seventies: many attempts to prove this conjecture
  3. Pullum and Gazdar 1982:
     - all these attempts have failed
     - for all we know, natural languages (conceived as string sets) might be context-free
  4. Huybregts 1984, Shieber 1985: proof that Swiss German is not context-free
  5. Joshi (1985) NLs are Mildly Context-sensitive Languages.

3.3. FG for Natural Languages

Now we will move to see how CFG have been applied to natural language. To this end, it is convenient to distinguish rules from non-terminal to terminal symbols which define the lexical entries (or lexicon).

3.3. FG for Natural Languages

Now we will move to see how CFG have been applied to natural language. To this end, it is convenient to distinguish rules from non-terminal to terminal symbols which define the lexical entries (or lexicon).

- Terminal: The terminal symbols are **words** (e.g. sara, dress ...).

3.3. FG for Natural Languages

Now we will move to see how CFG have been applied to natural language. To this end, it is convenient to distinguish rules from non-terminal to terminal symbols which define the lexical entries (or lexicon).

- Terminal: The terminal symbols are **words** (e.g. sara, dress ...).
- Non-terminal: The non-terminal symbols are syntactic **categories** (CAT) (e.g. n, det, np, vp, ...).

3.3. FG for Natural Languages

Now we will move to see how CFG have been applied to natural language. To this end, it is convenient to distinguish rules from non-terminal to terminal symbols which define the lexical entries (or lexicon).

- Terminal: The terminal symbols are **words** (e.g. sara, dress ...).
- Non-terminal: The non-terminal symbols are syntactic **categories** (CAT) (e.g. n, det, np, vp, ...).
- Start symbol: The start symbol is the s and stands for **sentence**.

3.3. FG for Natural Languages

Now we will move to see how CFG have been applied to natural language. To this end, it is convenient to distinguish rules from non-terminal to terminal symbols which define the lexical entries (or lexicon).

- Terminal: The terminal symbols are **words** (e.g. sara, dress . . .).
- Non-terminal: The non-terminal symbols are syntactic **categories** (CAT) (e.g. n, det, np, vp, . . .).
- Start symbol: The start symbol is the s and stands for **sentence**.

The production rules are divided into:

- Lexicon: e.g. $np \rightarrow$ sara. They form the set LEX

3.3. FG for Natural Languages

Now we will move to see how CFG have been applied to natural language. To this end, it is convenient to distinguish rules from non-terminal to terminal symbols which define the lexical entries (or lexicon).

- Terminal: The terminal symbols are **words** (e.g. sara, dress ...).
- Non-terminal: The non-terminal symbols are syntactic **categories** (CAT) (e.g. n, det, np, vp, ...).
- Start symbol: The start symbol is the s and stands for **sentence**.

The production rules are divided into:

- Lexicon: e.g. $np \rightarrow$ sara. They form the set LEX
- Grammatical Rules: They are of the type $s \rightarrow np\ vp$.

3.3. FG for Natural Languages

Now we will move to see how CFG have been applied to natural language. To this end, it is convenient to distinguish rules from non-terminal to terminal symbols which define the lexical entries (or lexicon).

- Terminal: The terminal symbols are **words** (e.g. sara, dress ...).
- Non-terminal: The non-terminal symbols are syntactic **categories** (CAT) (e.g. n, det, np, vp, ...).
- Start symbol: The start symbol is the s and stands for **sentence**.

The production rules are divided into:

- Lexicon: e.g. $np \rightarrow$ sara. They form the set LEX
- Grammatical Rules: They are of the type $s \rightarrow np\ vp$.

Well known formal grammars are Phrase Structure Grammars (PSG).

3.3.1. PSG: English Toy Fragment

We consider a small fragment of English defined by the following **grammar** $G = \langle \mathsf{LEX}, \mathsf{Rules} \rangle$, with vocabulary Σ and categories CAT.

3.3.1. PSG: English Toy Fragment

We consider a small fragment of English defined by the following **grammar** $G = \langle \mathsf{LEX}, \mathsf{Rules} \rangle$, with vocabulary Σ and categories CAT.

- $\mathsf{LEX} = \Sigma \times \mathsf{CAT}$

3.3.1. PSG: English Toy Fragment

We consider a small fragment of English defined by the following **grammar** $G = \langle \mathsf{LEX}, \mathsf{Rules} \rangle$, with vocabulary Σ and categories CAT.

- $\mathsf{LEX} = \Sigma \times \mathsf{CAT}$
  - $\Sigma = \{\text{Sara}, \text{dress}, \text{wears}, \text{the}, \text{new}\}$,

3.3.1. PSG: English Toy Fragment

We consider a small fragment of English defined by the following **grammar** $G = \langle \mathsf{LEX}, \mathsf{Rules} \rangle$, with vocabulary Σ and categories CAT.

- $\mathsf{LEX} = \Sigma \times \mathsf{CAT}$
  - $\Sigma = \{\text{Sara}, \text{dress}, \text{wears}, \text{the}, \text{new}\}$,
  - $\mathsf{CAT} = \{det, n, np, s, v, vp, adj\}$,

3.3.1. PSG: English Toy Fragment

We consider a small fragment of English defined by the following **grammar** $G = \langle \mathsf{LEX}, \mathsf{Rules} \rangle$, with vocabulary Σ and categories CAT.

- $\mathsf{LEX} = \Sigma \times \mathsf{CAT}$
  - $\Sigma = \{\text{Sara}, \text{dress}, \text{wears}, \text{the}, \text{new}\}$,
  - $\mathsf{CAT} = \{det, n, np, s, v, vp, adj\}$,
  - $\mathsf{LEX} = \{np \longrightarrow \text{Sara}\ det \longrightarrow \text{the}, n \longrightarrow \text{dress}, adj \longrightarrow \text{new}, v \longrightarrow \text{wears}\}$

3.3.1. PSG: English Toy Fragment

We consider a small fragment of English defined by the following **grammar** $G = \langle \mathsf{LEX}, \mathsf{Rules} \rangle$, with vocabulary Σ and categories CAT.

- $\mathsf{LEX} = \Sigma \times \mathsf{CAT}$
  - $\Sigma = \{\text{Sara}, \text{dress}, \text{wears}, \text{the}, \text{new}\}$,
  - $\mathsf{CAT} = \{det, n, np, s, v, vp, adj\}$,
  - $\mathsf{LEX} = \{np \rightarrow \text{Sara}\ det \rightarrow \text{the}, n \rightarrow \text{dress}, adj \rightarrow \text{new}, v \rightarrow \text{wears}\}$
- $\mathsf{Rules} = \{s \rightarrow np\ vp,\ np \rightarrow det\ n,\ vp \rightarrow v\ np, n \rightarrow adj\ n\}$

3.3.1. PSG: English Toy Fragment

We consider a small fragment of English defined by the following **grammar** $G = \langle \mathsf{LEX}, \mathsf{Rules} \rangle$, with vocabulary Σ and categories CAT.

- $\mathsf{LEX} = \Sigma \times \mathsf{CAT}$
  - $\Sigma = \{\text{Sara}, \text{dress}, \text{wears}, \text{the}, \text{new}\}$,
  - $\mathsf{CAT} = \{det, n, np, s, v, vp, adj\}$,
  - $\mathsf{LEX} = \{np \rightarrow \text{Sara}\ det \rightarrow \text{the}, n \rightarrow \text{dress}, adj \rightarrow \text{new}, v \rightarrow \text{wears}\}$
- $\mathsf{Rules} = \{s \rightarrow np\ vp,\ np \rightarrow det\ n,\ vp \rightarrow v\ np, n \rightarrow adj\ n\}$

Among the elements of the language recognized by the grammar, $L(G)$, are

3.3.1. PSG: English Toy Fragment

We consider a small fragment of English defined by the following **grammar** $G = \langle \mathsf{LEX}, \mathsf{Rules} \rangle$, with vocabulary Σ and categories CAT.

- $\mathsf{LEX} = \Sigma \times \mathsf{CAT}$
  - $\Sigma = \{\text{Sara}, \text{dress}, \text{wears}, \text{the}, \text{new}\}$,
  - $\mathsf{CAT} = \{det, n, np, s, v, vp, adj\}$,
  - $\mathsf{LEX} = \{np \longrightarrow \text{Sara } det \longrightarrow \text{the}, n \longrightarrow \text{dress}, adj \longrightarrow \text{new}, v \longrightarrow \text{wears}\}$
- $\mathsf{Rules} = \{s \longrightarrow np\ vp,\ np \longrightarrow det\ n,\ vp \longrightarrow v\ np, n \longrightarrow adj\ n\}$

Among the elements of the language recognized by the grammar, $L(G)$, are

- $det \longrightarrow^* \text{the}$
  —because this is in the lexicon, and

3.3.1. PSG: English Toy Fragment

We consider a small fragment of English defined by the following **grammar** $G = \langle \mathsf{LEX}, \mathsf{Rules} \rangle$, with vocabulary Σ and categories CAT.

- $\mathsf{LEX} = \Sigma \times \mathsf{CAT}$
  - $\Sigma = \{\text{Sara}, \text{dress}, \text{wears}, \text{the}, \text{new}\}$,
  - $\mathsf{CAT} = \{det, n, np, s, v, vp, adj\}$,
  - $\mathsf{LEX} = \{np \longrightarrow \text{Sara}\ det \longrightarrow \text{the}, n \longrightarrow \text{dress}, adj \longrightarrow \text{new}, v \longrightarrow \text{wears}\}$
- $\mathsf{Rules} = \{s \longrightarrow np\ vp,\ np \longrightarrow det\ n,\ vp \longrightarrow v\ np, n \longrightarrow adj\ n\}$

Among the elements of the language recognized by the grammar, $L(G)$, are

- $det \longrightarrow^* \text{the}$
  —because this is in the lexicon, and
- $s \longrightarrow^* \text{Sara wears the new dress}$
  —which is in the language by repeated applications of rules.

$adj \longrightarrow \text{new}$

$adj \longrightarrow$ new

adj
|
new

$adj \longrightarrow$ new

adj
|
new

$n \longrightarrow$ new

$adj \longrightarrow$ new

```
 adj
  |
 new
```

$n \longrightarrow$ new

```
   n
   |
 dress
```

$adj \rightarrow$ new

adj
|
new

$n \rightarrow$ new

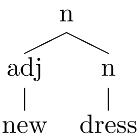
n
|
dress

n $\rightarrow$ adj n

$adj \longrightarrow$ new

$n \longrightarrow$ new

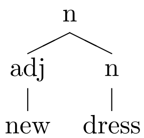
n $\longrightarrow$ adj n

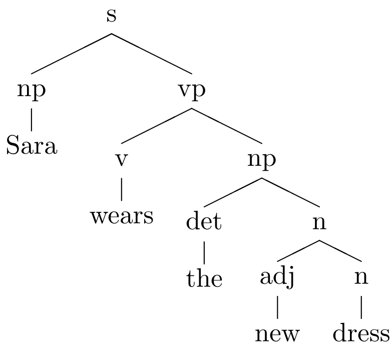


adj → new

n → new

n → adj n





[Sara[wears[the[new dress]$_{n}$]$_{np}$]$_{vp}$]$_{s}$

3.3.3. PSG: Advantages and Disadvantages Advantages

- PSG deals with phrase structures represented as trees.

3.3.3. PSG: Advantages and Disadvantages Advantages

- PSG deals with phrase structures represented as trees.
- Trees preserve aspects of the compositional (constituent) structure

3.3.3. PSG: Advantages and Disadvantages **Advantages**

- PSG deals with phrase structures represented as trees.
- Trees preserve aspects of the compositional (constituent) structure

Disadvantages

3.3.3. PSG: Advantages and Disadvantages **Advantages**

- PSG deals with phrase structures represented as trees.
- Trees preserve aspects of the compositional (constituent) structure

Disadvantages

- We are not capturing any general property of natural language assembly.

3.3.3. PSG: Advantages and Disadvantages **Advantages**

- PSG deals with phrase structures represented as trees.
- Trees preserve aspects of the compositional (constituent) structure

Disadvantages

- We are not capturing any general property of natural language assembly.
- Hence, to extend the grammar we have to keep on adding rules each time we add a word of a new category.

3.3.3. PSG: Advantages and Disadvantages

Advantages

- PSG deals with phrase structures represented as trees.
- Trees preserve aspects of the compositional (constituent) structure

Disadvantages

- We are not capturing any general property of natural language assembly.
- Hence, to extend the grammar we have to keep on adding rules each time we add a word of a new category.
- It's difficult to tiedly connect these (syntactic) rewriting rules with semantic rules to obtain meaning representations.

3.3.3. PSG: Advantages and Disadvantages Advantages

- PSG deals with phrase structures represented as trees.
- Trees preserve aspects of the compositional (constituent) structure

Disadvantages

- We are not capturing any general property of natural language assembly.
- Hence, to extend the grammar we have to keep on adding rules each time we add a word of a new category.
- It's difficult to tiedly connect these (syntactic) rewriting rules with semantic rules to obtain meaning representations.
- PSG as such don't handle long-distance dependencies, since there is no connection among categories occurring in different rewriting rules.

4. Parsing as deduction

We look for the Logic that properly models natural language syntax-semantics interface.

- We consider syntactic categories to be logical formulas

4. Parsing as deduction

We look for the Logic that properly models natural language syntax-semantics interface.

- We consider syntactic categories to be logical formulas
- As such, they can be atomic or complex (not just plain A, B, a, b etc.).

4. Parsing as deduction

We look for the Logic that properly models natural language syntax-semantics interface.

- We consider syntactic categories to be logical formulas
- As such, they can be atomic or complex (not just plain A, B, a, b etc.).
- They are related by means of the derivability relation ($\Rightarrow$). E.g

4. Parsing as deduction

We look for the Logic that properly models natural language syntax-semantics interface.

- We consider syntactic categories to be logical formulas
- As such, they can be atomic or complex (not just plain A, B, a, b etc.).
- They are related by means of the derivability relation ($\Rightarrow$). E.g

  $$np_{pl} \Rightarrow np$$

  all expressions that are plural np are also (under-specified) np.

4. Parsing as deduction

We look for the Logic that properly models natural language syntax-semantics interface.

- We consider syntactic categories to be logical formulas
- As such, they can be atomic or complex (not just plain A, B, a, b etc.).
- They are related by means of the derivability relation ($\Rightarrow$). E.g

$$np_{pl} \Rightarrow np$$

  all expressions that are plural *np* are also (under-specified) *np*.
- To recognize that a structure is of a certain category reduces to prove the formulas corresponding to the structure and the category are in a derivability relation $\Gamma \Rightarrow A$:

4. Parsing as deduction

We look for the Logic that properly models natural language syntax-semantics interface.

- We consider syntactic categories to be logical formulas
- As such, they can be atomic or complex (not just plain A, B, a, b etc.).
- They are related by means of the derivability relation ($\Rightarrow$). E.g

$$np_{pl} \Rightarrow np$$

  all expressions that are plural *np* are also (under-specified) *np*.
- To recognize that a structure is of a certain category reduces to prove the formulas corresponding to the structure and the category are in a derivability relation $\Gamma \Rightarrow A$:

$$CAT_{\text{sara}} CAT_{\text{wears}} CAT_{\text{the}} CAT_{\text{new}} CAT_{\text{dress}} \Rightarrow s?$$

4. Parsing as deduction

We look for the Logic that properly models natural language syntax-semantics interface.

- We consider syntactic categories to be logical formulas
- As such, they can be atomic or complex (not just plain A, B, a, b etc.).
- They are related by means of the derivability relation ($\Rightarrow$). E.g

$$np_{pl} \Rightarrow np$$

  all expressions that are plural *np* are also (under-specified) *np*.
- To recognize that a structure is of a certain category reduces to prove the formulas corresponding to the structure and the category are in a derivability relation $\Gamma \Rightarrow A$:

$$CAT_{\text{sara}} CAT_{\text{wears}} CAT_{\text{the}} CAT_{\text{new}} CAT_{\text{dress}} \Rightarrow s?$$

The slogan is:

4. Parsing as deduction

We look for the Logic that properly models natural language syntax-semantics interface.

- We consider syntactic categories to be logical formulas
- As such, they can be atomic or complex (not just plain A, B, a, b etc.).
- They are related by means of the derivability relation ($\Rightarrow$). E.g

$$np_{pl} \Rightarrow np$$

  all expressions that are plural np are also (under-specified) np.
- To recognize that a structure is of a certain category reduces to prove the formulas corresponding to the structure and the category are in a derivability relation $\Gamma \Rightarrow A$:

$$CAT_{\text{sara}} CAT_{\text{wears}} CAT_{\text{the}} CAT_{\text{new}} CAT_{\text{dress}} \Rightarrow s?$$

The slogan is:

> “Parsing as deduction”

4. Parsing as deduction

We look for the Logic that properly models natural language syntax-semantics interface.

- We consider syntactic categories to be logical formulas
- As such, they can be atomic or complex (not just plain A, B, a, b etc.).
- They are related by means of the derivability relation ($\Rightarrow$). E.g

$$np_{pl} \Rightarrow np$$

  all expressions that are plural np are also (under-specified) np.
- To recognize that a structure is of a certain category reduces to prove the formulas corresponding to the structure and the category are in a derivability relation $\Gamma \Rightarrow A$:

$$CAT_{\text{sara}} CAT_{\text{wears}} CAT_{\text{the}} CAT_{\text{new}} CAT_{\text{dress}} \Rightarrow s?$$

The slogan is:

"Parsing as deduction"

The question is:

4. Parsing as deduction

We look for the Logic that properly models natural language syntax-semantics interface.

- We consider syntactic categories to be logical formulas
- As such, they can be atomic or complex (not just plain A, B, a, b etc.).
- They are related by means of the derivability relation ($\Rightarrow$). E.g

  $$np_{pl} \Rightarrow np$$

  all expressions that are plural *np* are also (under-specified) *np*.
- To recognize that a structure is of a certain category reduces to prove the formulas corresponding to the structure and the category are in a derivability relation $\Gamma \Rightarrow A$:

  $$CAT_{\text{sara}} CAT_{\text{wears}} CAT_{\text{the}} CAT_{\text{new}} CAT_{\text{dress}} \Rightarrow s?$$

The slogan is:

"Parsing as deduction"

The question is:

which logic do we need?

5. Summing up

- We need a (logical) grammar able

5. Summing up

- We need a (logical) grammar able
  - to both compose (assembly) and decompose (extraction) linguistic structures

5. Summing up

- We need a (logical) grammar able
  - to both compose (assembly) and decompose (extraction) linguistic structures
  - to account for both local dependency and long distance dependency

5. Summing up

- We need a (logical) grammar able
  - to both compose (assembly) and decompose (extraction) linguistic structures
  - to account for both local dependency and long distance dependency
- the logical grammar should

5. Summing up

- We need a (logical) grammar able
  - to both compose (assembly) and decompose (extraction) linguistic structures
  - to account for both local dependency and long distance dependency
- the logical grammar should
  - be at least context free, but actually more –mildly context sensitive.

5. Summing up

- We need a (logical) grammar able
  - to both compose (assembly) and decompose (extraction) linguistic structures
  - to account for both local dependency and long distance dependency
- the logical grammar should
  - be at least context free, but actually more –mildly context sensitive.
  - be computationally appealing (polynomial)

5. Summing up

- We need a (logical) grammar able
  - to both compose (assembly) and decompose (extraction) linguistic structures
  - to account for both local dependency and long distance dependency
- the logical grammar should
  - be at least context free, but actually more –mildly context sensitive.
  - be computationally appealing (polynomial)
  - be tiedly related to meaning representation assembly

5. Summing up

- We need a (logical) grammar able
  - to both compose (assembly) and decompose (extraction) linguistic structures
  - to account for both local dependency and long distance dependency
- the logical grammar should
  - be at least context free, but actually more –mildly context sensitive.
  - be computationally appealing (polynomial)
  - be tiedly related to meaning representation assembly
  - capture the core of natural languages

5. Summing up

- We need a (logical) grammar able
  - to both compose (assembly) and decompose (extraction) linguistic structures
  - to account for both local dependency and long distance dependency
- the logical grammar should
  - be at least context free, but actually more –mildly context sensitive.
  - be computationally appealing (polynomial)
  - be tiedly related to meaning representation assembly
  - capture the core of natural languages
  - capture natural language diversities

6. Classical Logic vs. Non-Classical Logics

6. Classical Logic vs. Non-Classical Logics

Paradoxes of material implication In classical logic holds, for example, that a

- false statements imply any proposition ($\mathtt{F} \rightarrow \mathtt{T}, \mathtt{F} \rightarrow \mathtt{F}$ are both truth). Hence "if I am the pope, then 2+2=5" is true.
  But clearly even if one were the pope, 2+2 would still not be 5.

6. Classical Logic vs. Non-Classical Logics

Paradoxes of material implication In classical logic holds, for example, that a

- false statements imply any proposition ($\texttt{F} \longrightarrow \texttt{T}, \texttt{F} \longrightarrow \texttt{F}$ are both truth). Hence "if I am the pope, then 2+2=5" is true.

  But clearly even if one were the pope, 2+2 would still not be 5.

- in particular, a contradiction still implies everything.

6. Classical Logic vs. Non-Classical Logics

Paradoxes of material implication In classical logic holds, for example, that a

- false statements imply any proposition ($\texttt{F} \longrightarrow \texttt{T}, \texttt{F} \longrightarrow \texttt{F}$ are both truth). Hence "if I am the pope, then 2+2=5" is true.

  But clearly even if one were the pope, 2+2 would still not be 5.
- in particular, a contradiction still implies everything.
- any statement implies a tautology.

6. Classical Logic vs. Non-Classical Logics

Paradoxes of material implication In classical logic holds, for example, that a

- false statements imply any proposition ($\mathtt{F} \rightarrow \mathtt{T}, \mathtt{F} \rightarrow \mathtt{F}$ are both truth). Hence "if I am the pope, then 2+2=5" is true.

  But clearly even if one were the pope, 2+2 would still not be 5.
- in particular, a contradiction still implies everything.
- any statement implies a tautology.

These are examples of the material implication paradox of which some non-classical logics are an answer.

6. Classical Logic vs. Non-Classical Logics

Paradoxes of material implication In classical logic holds, for example, that a

- false statements imply any proposition ($\mathtt{F} \rightarrow \mathtt{T}, \mathtt{F} \rightarrow \mathtt{F}$ are both truth). Hence "if I am the pope, then 2+2=5" is true.

  But clearly even if one were the pope, 2+2 would still not be 5.
- in particular, a contradiction still implies everything.
- any statement implies a tautology.

These are examples of the material implication paradox of which some non-classical logics are an answer.

Non-classical logics are are either weaker or stronger than classical logics.

6.1. Weaker Non-Classical Logics

- reject one or more of the following tautologies that hold in Classical Logic:

6.1. Weaker Non-Classical Logics

- reject one or more of the following tautologies that hold in Classical Logic:
  - Law of the excluded middle: $P \vee \neg P$
    Rejected by **Multi-valued Logics** and **Intuitionistic Logic**

6.1. Weaker Non-Classical Logics

- reject one or more of the following tautologies that hold in Classical Logic:
  - Law of the excluded middle: $P \vee \neg P$
    Rejected by **Multi-valued Logics** and **Intuitionistic Logic**
  - De Morgan duality: every logical operator is dual to another. E.g.
    $\neg(P \wedge Q) = \neg P \vee \neg Q$, $\neg(P \vee Q) = \neg P \wedge \neg Q$
    Rejected by **Intuitionistic Logic**.

6.1. Weaker Non-Classical Logics

- reject one or more of the following tautologies that hold in Classical Logic:
  - Law of the excluded middle: $P \vee \neg P$
    Rejected by **Multi-valued Logics** and **Intuitionistic Logic**
  - De Morgan duality: every logical operator is dual to another. E.g.
    $\neg(P \wedge Q) = \neg P \vee \neg Q$, $\neg(P \vee Q) = \neg P \wedge \neg Q$
    Rejected by **Intuitionistic Logic**.
  - Law of non-contradiction: $\neg(P \wedge \neg P)$
    Rejected by **Paraconsistent Logics** and **Relevant Logic**.

- Reject one or all of the following properties proper of Classical Logic:
  - Monotonicity of entailment: If $\Gamma \Rightarrow B$ then $\Gamma, A \Rightarrow B$
    Rejected by **Relevant Logic**, **Linear Logic** and **Lambek Calculus**.

- Reject one or all of the following properties proper of Classical Logic:
  - Monotonicity of entailment: If $\Gamma \Rightarrow B$ then $\Gamma, A \Rightarrow B$
    Rejected by **Relevant Logic**, **Linear Logic** and **Lambek Calculus**.
  - Idempotency of entailment: If $\Gamma, A, A \Rightarrow B$ then $\Gamma, A \Rightarrow B$
    Rejected by **BCK-Logic**, **Linear Logic** and **Lambek Calculus**.

- Reject one or all of the following properties proper of Classical Logic:
  - Monotonicity of entailment: If $\Gamma \Rightarrow B$ then $\Gamma, A \Rightarrow B$
    Rejected by **Relevant Logic**, **Linear Logic** and **Lambek Calculus**.
  - Idempotency of entailment: If $\Gamma, A, A \Rightarrow B$ then $\Gamma, A \Rightarrow B$
    Rejected by **BCK-Logic**, **Linear Logic** and **Lambek Calculus**.
  - Commutativity of conjunction:
    $A \wedge B \Leftrightarrow B \wedge A$
    Rejected by **Lambek Calculus**.

- Reject one or all of the following properties proper of Classical Logic:
  - Monotonicity of entailment: If $\Gamma \Rightarrow B$ then $\Gamma, A \Rightarrow B$
    Rejected by **Relevant Logic**, **Linear Logic** and **Lambek Calculus**.
  - Idempotency of entailment: If $\Gamma, A, A \Rightarrow B$ then $\Gamma, A \Rightarrow B$
    Rejected by **BCK-Logic**, **Linear Logic** and **Lambek Calculus**.
  - Commutativity of conjunction:
    $A \wedge B \Leftrightarrow B \wedge A$
    Rejected by **Lambek Calculus**.
  - Associativity of conjunction $(A \wedge B) \wedge C \Leftrightarrow A \wedge (B \wedge C)$
    Rejected by **Lambek Calculus**.

6.2. Stronger Non-Classical Logics

- or extend Classical Logic with other operators. E.g.

6.2. Stronger Non-Classical Logics

- or extend Classical Logic with other operators. E.g.

  Modal Logics are obtained by extending classical logic with non-truth-functional ("intensional") operators (can, could, might, may, must, possibly, necessarily, eventually): the truth value of a complex formula cannot be determined by the truth values of its subformulae.

6.2. Stronger Non-Classical Logics

- or extend Classical Logic with other operators. E.g.

  Modal Logics are obtained by extending classical logic with non-truth-functional ("intensional") operators (can, could, might, may, must, possibly, necessarily, eventually): the truth value of a complex formula cannot be determined by the truth values of its subformulae.

  - Modal Logics speak about **relational structures**, i.e. a set (the domain) together with a collection of n-relations on that set.

6.2. Stronger Non-Classical Logics

- or extend Classical Logic with other operators. E.g.

  Modal Logics are obtained by extending classical logic with non-truth-functional ("intensional") operators (can, could, might, may, must, possibly, necessarily, eventually): the truth value of a complex formula cannot be determined by the truth values of its subformulae.

  - Modal Logics speak about **relational structures**, i.e. a set (the domain) together with a collection of n-relations on that set.
  - The elements of the domain can be: points, worlds, times, ...

6.2. Stronger Non-Classical Logics

- or extend Classical Logic with other operators. E.g.

  Modal Logics are obtained by extending classical logic with non-truth-functional ("intensional") operators (can, could, might, may, must, possibly, necessarily, eventually): the truth value of a complex formula cannot be determined by the truth values of its subformulae.

  - Modal Logics speak about **relational structures**, i.e. a set (the domain) together with a collection of n-relations on that set.
  - The elements of the domain can be: points, worlds, times, ...

    natural language phrases.

6.2. Stronger Non-Classical Logics

- or extend Classical Logic with other operators. E.g.

  Modal Logics are obtained by extending classical logic with non-truth-functional ("intensional") operators (can, could, might, may, must, possibly, necessarily, eventually): the truth value of a complex formula cannot be determined by the truth values of its subformulae.

  - Modal Logics speak about **relational structures**, i.e. a set (the domain) together with a collection of n-relations on that set.
  - The elements of the domain can be: points, worlds, times, . . .

    natural language phrases.
  - Examples of modal operators: Possibility and Necessity in the future, $\Diamond$, $\Box$, respectively, or Possibility and Necessity in the past, ⧫, ■

7. Proof Theory: Gentzen Sequent Calculus

- Gerhard Gentzen introduced Sequent Calculi and distinguished:
  - **Logical rules**: the rules of the **logical constants** $(\wedge, \vee, \rightarrow, \neg)$,

7. Proof Theory: Gentzen Sequent Calculus

- Gerhard Gentzen introduced Sequent Calculi and distinguished:
  - **Logical rules**: the rules of the **logical constants** $(\wedge, \vee, \rightarrow, \neg)$,
  - **Structural rules**: the rules that govern the composition modes of the premises.

7. Proof Theory: Gentzen Sequent Calculus

- Gerhard Gentzen introduced Sequent Calculi and distinguished:
  - ▷ **Logical rules**: the rules of the **logical constants** $(\wedge, \vee, \rightarrow, \neg)$,
  - ▷ **Structural rules**: the rules that govern the composition modes of the premises.
- The logical rules are divided into:

7. Proof Theory: Gentzen Sequent Calculus

- Gerhard Gentzen introduced Sequent Calculi and distinguished:
  - ▷ **Logical rules**: the rules of the **logical constants** $(\wedge, \vee, \rightarrow, \neg)$,
  - ▷ **Structural rules**: the rules that govern the composition modes of the premises.
- The logical rules are divided into:
  - ▷ **Right rules**: rules introducing the logical constants on the right of the turnstile $(\vdash)$ (instead of introduction rules of ND), and

7. Proof Theory: Gentzen Sequent Calculus

- Gerhard Gentzen introduced Sequent Calculi and distinguished:
  - **Logical rules**: the rules of the **logical constants** $(\wedge, \vee, \rightarrow, \neg)$,
  - **Structural rules**: the rules that govern the composition modes of the premises.
- The logical rules are divided into:
  - **Right rules**: rules introducing the logical constants on the right of the turnstile $(\vdash)$ (instead of introduction rules of ND), and
  - **Left rules**: rules introducing the constants on the left (instead of the elimination rules ND).

7.1. Classical Logic: Logical Rules

$$A, \Gamma \vdash A, \Delta$$

$$\frac{A, B, \Gamma \vdash \Delta}{A \wedge B, \Gamma \vdash \Delta}\ (\wedge L) \qquad \frac{\Gamma \vdash \Delta, A \quad \Gamma \vdash \Delta, B}{\Gamma \vdash \Delta, A \wedge B}\ (\wedge R)$$

$$\frac{A, \Gamma \vdash \Delta \quad B, \Gamma \vdash \Delta}{A \vee B, \Gamma \vdash \Delta}\ (\vee L) \qquad \frac{\Gamma \vdash \Delta, A, B}{\Gamma \vdash \Delta, A \vee B}\ (\vee R)$$

$$\frac{\Gamma \vdash \Delta, A \quad B, \Gamma \vdash \Delta}{A \rightarrow B, \Gamma \vdash \Delta}\ (\rightarrow L) \qquad \frac{A, \Gamma \vdash \Delta, B}{\Gamma \vdash \Delta, A \rightarrow B}\ (\rightarrow R)$$

$$\frac{\Gamma \vdash \Delta, A}{\neg A, \Gamma \vdash \Delta}\ (\neg L) \qquad \frac{A, \Gamma \vdash \Delta}{\Gamma \vdash \Delta, \neg A}\ (\neg R)$$

A, B stand for logical formulas. Γ, Δ, Σ stand for **sets** of formulas. Hence the order and quantity of formulas occurrences don't count.

7.1. Classical Logic: Logical Rules

$$A, \Gamma \vdash A, \Delta$$

$$\frac{A, B, \Gamma \vdash \Delta}{A \wedge B, \Gamma \vdash \Delta}\ (\wedge\mathrm{L}) \qquad \frac{\Gamma \vdash \Delta, A \quad \Gamma \vdash \Delta, B}{\Gamma \vdash \Delta, A \wedge B}\ (\wedge\mathrm{R})$$

$$\frac{A, \Gamma \vdash \Delta \quad B, \Gamma \vdash \Delta}{A \vee B, \Gamma \vdash \Delta}\ (\vee\mathrm{L}) \qquad \frac{\Gamma \vdash \Delta, A, B}{\Gamma \vdash \Delta, A \vee B}\ (\vee\mathrm{R})$$

$$\frac{\Gamma \vdash \Delta, A \quad B, \Gamma \vdash \Delta}{A \rightarrow B, \Gamma \vdash \Delta}\ (\rightarrow \mathrm{L}) \qquad \frac{A, \Gamma \vdash \Delta, B}{\Gamma \vdash \Delta, A \rightarrow B}\ (\rightarrow \mathrm{R})$$

$$\frac{\Gamma \vdash \Delta, A}{\neg A, \Gamma \vdash \Delta}\ (\neg\mathrm{L}) \qquad \frac{A, \Gamma \vdash \Delta}{\Gamma \vdash \Delta, \neg A}\ (\neg\mathrm{R})$$

A, B stand for logical formulas. Γ, Δ, Σ stand for **sets** of formulas. Hence the order and quantity of formulas occurrences don't count.

The , stands for "and" when occurring on the right of the turnstile ($\vdash$) and for "or" when occurring on its left. Think of **Tableaux** ($\mathrm{T} \vdash \mathrm{F}$ —axioms contradictions.)

7.2. Classical Logic: Structural Rules

An important discovery in Gentzen's thesis [1935] is that in logic there are rules of inference that don't involve any logical constant. Gentzen called such rules "structural".

7.2. Classical Logic: Structural Rules

An important discovery in Gentzen's thesis [1935] is that in logic there are rules of inference that don't involve any logical constant. Gentzen called such rules "structural".

Hidden in Classical Logic there are the following structural rules.

7.2. Classical Logic: Structural Rules

An important discovery in Gentzen's thesis [1935] is that in logic there are rules of inference that don't involve any logical constant. Gentzen called such rules "structural".

Hidden in Classical Logic there are the following structural rules.

Weakening	$\dfrac{\Lambda \vdash \Sigma}{B, \Lambda \vdash \Sigma}$	$\dfrac{\Lambda \vdash \Sigma}{\Lambda \vdash B, \Sigma}$	Axiom: $\Gamma, A \vdash \Delta, A$
Contraction	$\dfrac{A, A, \Gamma \vdash \Sigma}{A, \Gamma \vdash \Sigma}$	$\dfrac{\Gamma \vdash A, A, \Sigma}{\Gamma \vdash A, \Sigma}$	structures are sets: nr. does not count
Permutation	$\dfrac{A, B, \Gamma \vdash \Sigma}{B, A, \Gamma \vdash \Sigma}$	$\dfrac{\Gamma \vdash A, B, \Sigma}{\Gamma \vdash B, A, \Sigma}$	structures are sets: order doesn't count

Furthermore, the comma is associative: $(A, (B, C)) = ((A, B), C)$.

8. Conditional and Logical Consequence

In particular, the separation of structural rules from logical rules helped highlighting the **crucial role** played by conditional and the residuation **condition below** that captures the tied connection between conditional and logical consequence, i.e. the core of logic.

$$p, q \vdash r \quad \text{iff} \quad p \vdash q \rightarrow r$$

It says that r follows from p together with q just when $q \rightarrow r$ follows from p alone.

8. Conditional and Logical Consequence

In particular, the separation of structural rules from logical rules helped highlighting the **crucial role** played by conditional and the residuation **condition below** that captures the tied connection between conditional and logical consequence, i.e. the core of logic.

$$p, q \vdash r \text{ iff } p \vdash q \to r$$

It says that r follows from p together with q just when $q \to r$ follows from p alone.

However, there is one extra factor in the equation. Not only is there the **turnstile** ($\vdash$), for logical consequence, and the **conditional** ($\to$), encoding consequence inside the language of propositions, there is also the **comma**, indicating the combination of premises. The behaviour of premise combination is also important in determining the behaviour of the conditional.

As the comma's behaviour varies, so does the conditional.

9. The "comma"

The comma's behavior varies accordingly to the structural rules allowed. Hence the latter play an important role.

- There can exist logics that share logical rules while differ with respect to the structures of the premises.

9. The "comma"

The comma's behavior varies accordingly to the structural rules allowed. Hence the latter play an important role.

- There can exist logics that share logical rules while differ with respect to the structures of the premises.
- there are non-classical logics that lack one or all structural rules, while share the residuation condition above, and have been motivated by the rejection of the material implication paradoxes.

9. The "comma"

The comma's behavior varies accordingly to the structural rules allowed. Hence the latter play an important role.

- There can exist logics that share logical rules while differ with respect to the structures of the premises.
- there are non-classical logics that lack one or all structural rules, while share the residuation condition above, and have been motivated by the rejection of the material implication paradoxes.

10. The Core of Logic

The distinction between logical and structural rules helped

10. The Core of Logic

The distinction between logical and structural rules helped

- capturing the **core** of the family of logics we have been discussing,

10. The Core of Logic

The distinction between logical and structural rules helped

- capturing the **core** of the family of logics we have been discussing,
- realize that the structural rules above play an essential role to obtain logics that **avoid the material implication paradoxes**, and

10. The Core of Logic

The distinction between logical and structural rules helped

- capturing the **core** of the family of logics we have been discussing,
- realize that the structural rules above play an essential role to obtain logics that **avoid the material implication paradoxes**, and
- **fine tune logics** on the base of the object of investigation.

11. Substructural Logics

The common denominator of several non-classical logics is that in their sequent formulation they:

11. Substructural Logics

The common denominator of several non-classical logics is that in their sequent formulation they:

- share rules for **logical constants**

11. Substructural Logics

The common denominator of several non-classical logics is that in their sequent formulation they:

- share rules for **logical constants**
- reject or restrict some **structural rules** proper of/hidden in Classical Logic.

11. Substructural Logics

The common denominator of several non-classical logics is that in their sequent formulation they:

- share rules for **logical constants**
- reject or restrict some **structural rules** proper of/hidden in Classical Logic.

Non-classical logics obtained from Classical Logics by dropping some or all of the structural rules are called "**Substructural Logics**".

11. Substructural Logics

The common denominator of several non-classical logics is that in their sequent formulation they:

- share rules for **logical constants**
- reject or restrict some **structural rules** proper of/hidden in Classical Logic.

Non-classical logics obtained from Classical Logics by dropping some or all of the structural rules are called "**Substructural Logics**".

They are logics sensitive to the number and order of occurrence of assumptions. By this reason, they are sometimes called "**Resource Sensitive Logics**".

11. Substructural Logics

The common denominator of several non-classical logics is that in their sequent formulation they:

- share rules for **logical constants**
- reject or restrict some **structural rules** proper of/hidden in Classical Logic.

Non-classical logics obtained from Classical Logics by dropping some or all of the structural rules are called "**Substructural Logics**".

They are logics sensitive to the number and order of occurrence of assumptions. By this reason, they are sometimes called "**Resource Sensitive Logics**".

In this light, structures are not seen anymore as sets but, for instance, as lists so to distinguish the order and quantity of formulas occurrences.

11.1. Substructural Logics: Examples

The most well-known substructural logics are:

1. Relevant Logic: rejects Weakening.
2. BCK logic: rejects Contraction.
3. Linear logic: rejects Weakening and Contraction.
4. Lambek Calculus: rejects all the substructural rules of Classical Logic.

11.1.1. Relevant Logics: Intuition

Aim Understand the notions of consequence and conditionality where the **conclusion of a valid argument is relevant to the premises**, and where the consequent of a true conditionals is relevant to the antecedent.

We want "**if** A **then** B" to mean that "B follows from A", i.e. we used A in the deduction of B, hence A **is relevant to derive** B. For instance, we do not want that from false everything falls:

> "The moon is made of green cheese. Therefore, either it is raining in Ecuador now or it is not."

this argument commits "fallacies of relevance".

Similarly, we do not want that everything follows from contradiction:

> "It is raining in Ecuador and it is not raining. Therefore, I am teaching in Australia."

Relevant Logic: Rejected Tautologies The following formulas are tautologies of Classical Logic and Intuitionistic Logic:

$$A \to (B \to B) \quad \neg(B \to B) \to A$$

there is no such connection of relevance: The consequent of the main conditional needs not have anything to do with the antecedent.

Relevant Logic: Rejected Tautologies The following formulas are tautologies of Classical Logic and Intuitionistic Logic:

$$A \to (B \to B) \quad \neg(B \to B) \to A$$

there is no such connection of relevance: The consequent of the main conditional needs not have anything to do with the antecedent.

Similarly, in Classical Logic the following formula holds:

$$B \to (\neg B \to A)$$

Relevant Logics: Rejected Structural Rule Proofs:

$$\dfrac{\dfrac{\dfrac{B \vdash B}{A, B \vdash B}\ (\text{Weak})}{A \vdash B \rightarrow B}\ (\rightarrow \mathrm{R})}{\vdash A \rightarrow (B \rightarrow B)}\ (\rightarrow \mathrm{R})$$

$$\dfrac{\dfrac{\dfrac{\dfrac{B \vdash B}{B \vdash A, B}\ (\text{Weak})}{B, \neg B \vdash A}\ (\neg \mathrm{L})}{B \vdash \neg B \rightarrow A}\ (\rightarrow \mathrm{R})}{\vdash B \rightarrow (\neg B \rightarrow A)}\ (\rightarrow \mathrm{R})$$

The proofs are based on weakening. Hence, this structural rule is dropped and axioms are of the form $A \vdash A$, "irrelevant" information cannot be brought in the proof.

Notice, rejecting $B \rightarrow (\neg B \rightarrow A)$ as tautology reduces to reject $\neg(B \wedge \neg B)$, too.

11.1.2. Linear Logic: Intuition

Aim understand and model processes and **resource use**. The idea in this account of deduction is that resources must be used (so premise combination satisfies the relevance criterion) and they do not extend indefinitely. Premises cannot be re-used.

Hence, we want “**if** A **then** B” to mean that **the resource** B **is produced if** A **is consumed**.

In Linear logic, duplicated hypotheses ’count’ differently from single occurrences

> I have a quart of milk from which I can make a pound of butter. If I decide to make butter out of all of my milk, I cannot then conclude that I have both milk and butter!

Linear Logic: Rejected Structural Rules For instance, the proof of the argument below shows that Classical logic is "insensitive" to resources.

$$\{p, p \rightarrow q, p \rightarrow r, (q \wedge r) \rightarrow t, s\} \vdash t$$

s is not used to prove t, p is used twice.

Linear Logic: Rejected Structural Rules

For instance, the proof of the argument below shows that Classical logic is "insensitive" to resources.

$$\{p, p \to q, p \to r, (q \land r) \to t, s\} \vdash t$$

s is not used to prove t, p is used twice.

Hence, in Linear Logic both Contraction and Weakening are dropped:

- The rejection of Contraction means that resources cannot be duplicated (see the p.)

Linear Logic: Rejected Structural Rules

For instance, the proof of the argument below shows that Classical logic is "insensitive" to resources.

$$\{p, p \rightarrow q, p \rightarrow r, (q \wedge r) \rightarrow t, s\} \vdash t$$

s is not used to prove t, p is used twice.

Hence, in Linear Logic both Contraction and Weakening are dropped:

- The rejection of Contraction means that resources cannot be duplicated (see the p.)
- The rejection of Weakening means that resources cannot be thrown away (see the s.)

Structures are seen as **multisets** (the order of occurrence doesn't count, the quantity does).

Linear Logic: Structural Control Contraction and Weakening are reintroduced by marking formulas for which they can hold with unary operators "!" and "?". For instance,

$$\frac{X \vdash Y}{X, !A \vdash Y} \quad \frac{X \vdash Y}{X \vdash ?A, Y}$$

The use of these "markers" introduces the idea of structural rules that are **controlled rather then globally available**.

11.1.3. Lambek Calculus: Intuition

Aim to model natural language assembly, capture the core of natural languages and their diversity.

11.1.3. Lambek Calculus: Intuition

Aim to model natural language assembly, capture the core of natural languages and their diversity.

- the multiplicity of linguistics material is important, since linguistic elements must generally be used once and only once during an analysis. Thus, we cannot ignore or waste linguistic material (a), nor can we indiscriminately duplicate it (b).

  a) *The coach smiled the ball. ≠ The coach smiled.

  b) *The coach smiled smiled. ≠ The coach smiled.

11.1.3. Lambek Calculus: Intuition

Aim to model natural language assembly, capture the core of natural languages and their diversity.

- the multiplicity of linguistics material is important, since linguistic elements must generally be used once and only once during an analysis. Thus, we cannot ignore or waste linguistic material (a), nor can we indiscriminately duplicate it (b).

  a) *The coach smiled the ball. $\neq$ The coach smiled.

  b) *The coach smiled smiled. $\neq$ The coach smiled.

- natural language structures are neither commutative (c) nor associative (d)

  c) $[[[\text{the}]_{art}\ [\text{coach}]_n]_{np}\ [\text{smiled}]_v]_s \neq^* [[\text{smiled}]_v[[\text{the}]_{art}[\text{coach}]_v]_{np}]_s$

  d) $[[[\text{the}]_{art}\ [\text{coach}]_n]_{np}\ [\text{smiled}]_v]_s \neq [[\text{the}]_{art}\ [[\text{coach}]_n\ [\text{smiled}]_v]]_s$

- No contraction, no weakening, no associativity, no commutativity. Hence,

- No contraction, no weakening, no associativity, no commutativity. Hence,
- Implication is split into:
  - right implication ($A\backslash B$ — if A then B $A \rightarrow B$)
  - left implication (B/A — B if A, $B \leftarrow A$)

$$\dfrac{\dfrac{\dfrac{B \vdash B \quad A \vdash A}{A/B, B \vdash A}\,(/L)}{B, A/B \vdash A}\,(\text{Exc})}{A/B \vdash B\backslash A}\,(\backslash R)$$

Lambek Calculus: No structural rules

- No contraction, no weakening, no associativity, no commutativity. Hence,
- Implication is split into:
  - right implication ($A\backslash B$ — if A then B $A \rightarrow B$)
  - left implication (B/A — B if A, $B \leftarrow A$)

$$\dfrac{\dfrac{\dfrac{B \vdash B \quad A \vdash A}{A/B, B \vdash A}\,(/\mathrm{L})}{B, A/B \vdash A}\,(\mathrm{Exc})}{A/B \vdash B\backslash A}\,(\backslash\mathrm{R})$$

- structures are lists

Lambek Calculus: No structural rules

- No contraction, no weakening, no associativity, no commutativity. Hence,
- Implication is split into:
  - right implication ($A\backslash B$ — if A then B $A \rightarrow B$)
  - left implication (B/A — B if A, $B \leftarrow A$)

$$
\cfrac{\cfrac{\cfrac{B \vdash B \quad A \vdash A}{A/B, B \vdash A}\ (/L)}{B, A/B \vdash A}\ (\text{Exc})}{A/B \vdash B\backslash A}\ (\backslash R)
$$

- structures are lists
- conjunction is seen as fusion

- No contraction, no weakening, no associativity, no commutativity. Hence,
- Implication is split into:
  - right implication ($A\backslash B$ — if A then B $A \rightarrow B$)
  - left implication (B/A — B if A, $B \leftarrow A$)

$$\dfrac{\dfrac{\dfrac{B \vdash B \quad A \vdash A}{A/B, B \vdash A}\,(/\mathrm{L})}{B, A/B \vdash A}\,(\mathrm{Exc})}{A/B \vdash B\backslash A}\,(\backslash \mathrm{R})$$

- structures are lists
- conjunction is seen as fusion
- And no negation.

Lambek Calculus: Structural Control As in other domains, in Linguistic as well, there is the need of locally controlling structural reasoning and account for the different compositional relations linguistic phenomena may exhibit. For instance,

- E.g. Adjectives: "Vestito nuovo" vs. "Nuovo vestito." Need to some sort of commutativity.

Lambek Calculus: Structural Control As in other domains, in Linguistic as well, there is the need of locally controlling structural reasoning and account for the different compositional relations linguistic phenomena may exhibit. For instance,

- E.g. Adjectives: "Vestito nuovo" vs. "Nuovo vestito." Need to some sort of commutativity.
- E.g. Coordination: "[[I want to try to write [. . .]] and [hope to see produced [. . .]] the movie]" need of some form of associativity.

Lambek Calculus: Structural Control As in other domains, in Linguistic as well, there is the need of locally controlling structural reasoning and account for the different compositional relations linguistic phenomena may exhibit. For instance,

- E.g. Adjectives: "Vestito nuovo" vs. "Nuovo vestito." Need to some sort of commutativity.
- E.g. Coordination: "[[I want to try to write [...]] and [hope to see produced [...]] the movie]" need of some form of associativity.

But differently from linear logic, the control is expressed by means of unary or binary logical operators living within the same algebraic structure (residuation).

11.2. Remarks: Residuation

Let $\langle A, \leq_1 \rangle$ and $\langle B, \leq_2 \rangle$ be two partially ordered sets. A pair of functions (f, g) such that $f : A \to B$ and $g : B \to A$ forms a residuated pair if

$$[RES_1] \quad \forall x \in A, y \in B \left(\begin{array}{c} f(x) \leq_2 y \\ \text{iff} \\ x \leq_1 g(y) \end{array} \right)$$

11.2. Remarks: Residuation

Let $\langle A, \leq_1 \rangle$ and $\langle B, \leq_2 \rangle$ be two partially ordered sets. A pair of functions (f, g) such that $f : A \to B$ and $g : B \to A$ forms a residuated pair if

$$[RES_1] \quad \forall x \in A, y \in B \begin{pmatrix} f(x) \leq_2 y \\ \text{iff} \\ x \leq_1 g(y) \end{pmatrix}$$

Let $\langle C, \leq_3 \rangle$ be a third partially ordered set, a triple of functions (f, g, h) such that $f : A \times B \to C$, $g : A \times C \to B$, $h : C \times B \to A$ forms a residuated triple if

$$[RES_2] \quad \forall x \in A, y \in B, z \in C \begin{pmatrix} x \leq_1 h(z, y) \\ \text{iff} \\ f(x, y) \leq_3 z \\ \text{iff} \\ y \leq_2 g(x, z) \end{pmatrix}$$

Similarly, one could have dual-residuated operators where h and g are in the right side of $\leq$ and f on its right.

11.2.1. Residuation: Intuition

For instance, in Maths:

$$[RES_2] \quad \begin{pmatrix} 2 \leq \frac{9}{4} \\ \text{iff} \\ 2 \times 4 \leq 9 \\ \text{iff} \\ 4 \leq \frac{9}{2} \end{pmatrix}$$

$$p, q \vdash r \quad \text{iff} \quad p \vdash q \longrightarrow r$$

11.2.1. Residuation: Intuition

For instance, in Maths:

$$[RES_2] \quad \begin{pmatrix} 2 \leq \frac{9}{4} \\ \text{iff} \\ 2 \times 4 \leq 9 \\ \text{iff} \\ 4 \leq \frac{9}{2} \end{pmatrix}$$

$$p, q \vdash r \quad \text{iff} \quad p \vdash q \longrightarrow r$$

For instance, in linguistics:

$$[RES_2] \quad \begin{pmatrix} np : mary \leq \frac{s}{iv:walks} \\ \text{iff} \\ np : mary \times iv : walks \leq s \\ \text{iff} \\ walks : iv \leq \frac{s}{np:mary} \end{pmatrix}$$

11.2.2. Residuation: Tonicity and Composition

Saying that (f, g) is a residuated pair is equivalent to the conditions $i)$ and $ii)$,

$i)$ Tonicity: $f(+)$ and $g(+)$.

they preserve the order of their arguments, i.e. $f(x) \leq f(y)$ if $x \leq y$.

$$ii)\text{Composition}: \quad \forall y \in B, x \in A \begin{pmatrix} f(g(y)) \leq_2 y \\ \text{and} \\ x \leq_1 g(f(x)) \end{pmatrix}$$

11.2.2. Residuation: Tonicity and Composition

Saying that (f, g) is a residuated pair is equivalent to the conditions $i)$ and $ii)$,

$i)$ Tonicity: $f(+)$ and $g(+)$.

they preserve the order of their arguments, i.e. $f(x) \leq f(y)$ if $x \leq y$.

$$ii)\text{Composition}: \quad \forall y \in B, x \in A \begin{pmatrix} f(g(y)) \leq_2 y \\ \text{and} \\ x \leq_1 g(f(x)) \end{pmatrix}$$

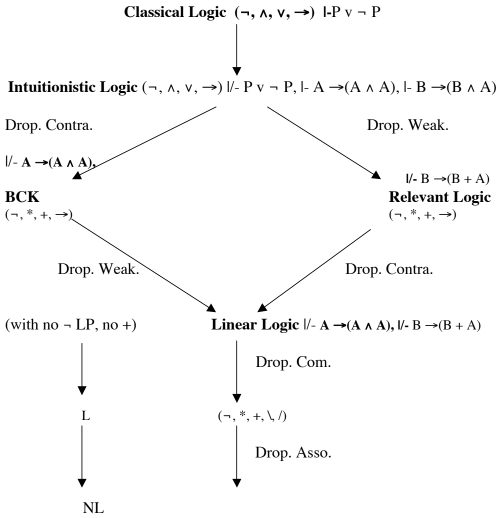
Similarly, saying that (f, g, h) is a residuated triple is equivalent to requiring

i)Tonicity: $f(+,+), g(-,+)$ and $h(+,-)$

where $-$ means, it reverses the order of its argument, i.e. $g(y, z) \leq g(x, z)$ if$x \leq y$.

$$ii)\text{Composition}: \quad \forall x \in A, y \in B, z \in C \begin{pmatrix} f(x, g(x, z)) \leq_3 z \\ \text{and} \\ y \leq_2 g(x, f(x, y)) \\ \text{and} \\ f(h(z, y), y) \leq_3 z \\ \text{and} \\ x \leq_1 h(f(x, y), y) \end{pmatrix}$$

12. Conclusion: Substructural Logic



Plus, the + has its own residuated operators.. the co-implications.

13. Conclusion: Logical Grammar

Tomorrow we'll look at the Lambek Calculus as Logical Grammar and show it's able to

13. Conclusion: Logical Grammar

Tomorrow we'll look at the Lambek Calculus as Logical Grammar and show it's able to

- both compose (assembly) and decompose (extraction) linguistic structures.

13. Conclusion: Logical Grammar

Tomorrow we'll look at the Lambek Calculus as Logical Grammar and show it's able to

- both compose (assembly) and decompose (extraction) linguistic structures.
- account for local dependency.

13. Conclusion: Logical Grammar

Tomorrow we'll look at the Lambek Calculus as Logical Grammar and show it's able to

- both compose (assembly) and decompose (extraction) linguistic structures.
- account for local dependency.
- compositionally account for the assembly of meaning representation.

13. Conclusion: Logical Grammar

Tomorrow we'll look at the Lambek Calculus as Logical Grammar and show it's able to

- both compose (assembly) and decompose (extraction) linguistic structures.
- account for local dependency.
- compositionally account for the assembly of meaning representation.

Moreover, we'll look at its expressivity w.r.t. Chomsky Hierarchy and its limitations:

13. Conclusion: Logical Grammar

Tomorrow we'll look at the Lambek Calculus as Logical Grammar and show it's able to

- both compose (assembly) and decompose (extraction) linguistic structures.
- account for local dependency.
- compositionally account for the assembly of meaning representation.

Moreover, we'll look at its expressivity w.r.t. Chomsky Hierarchy and its limitations:

- long distance dependencies

13. Conclusion: Logical Grammar

Tomorrow we'll look at the Lambek Calculus as Logical Grammar and show it's able to

- both compose (assembly) and decompose (extraction) linguistic structures.
- account for local dependency.
- compositionally account for the assembly of meaning representation.

Moreover, we'll look at its expressivity w.r.t. Chomsky Hierarchy and its limitations:

- long distance dependencies
- non local scope construal

13. Conclusion: Logical Grammar

Tomorrow we'll look at the Lambek Calculus as Logical Grammar and show it's able to

- both compose (assembly) and decompose (extraction) linguistic structures.
- account for local dependency.
- compositionally account for the assembly of meaning representation.

Moreover, we'll look at its expressivity w.r.t. Chomsky Hierarchy and its limitations:

- long distance dependencies
- non local scope construal

Finally, we will indicate solutions proposed to overcome these limitations and explain the main "philosophy" w.r.t. to

13. Conclusion: Logical Grammar

Tomorrow we'll look at the Lambek Calculus as Logical Grammar and show it's able to

- both compose (assembly) and decompose (extraction) linguistic structures.
- account for local dependency.
- compositionally account for the assembly of meaning representation.

Moreover, we'll look at its expressivity w.r.t. Chomsky Hierarchy and its limitations:

- long distance dependencies
- non local scope construal

Finally, we will indicate solutions proposed to overcome these limitations and explain the main "philosophy" w.r.t. to

- the core on natural languages

13. Conclusion: Logical Grammar

Tomorrow we'll look at the Lambek Calculus as Logical Grammar and show it's able to

- both compose (assembly) and decompose (extraction) linguistic structures.
- account for local dependency.
- compositionally account for the assembly of meaning representation.

Moreover, we'll look at its expressivity w.r.t. Chomsky Hierarchy and its limitations:

- long distance dependencies
- non local scope construal

Finally, we will indicate solutions proposed to overcome these limitations and explain the main "philosophy" w.r.t. to

- the core on natural languages
- natural language diversities