The quantification calculus (QC)

András Máté

07.11.2025

First-order languages

First-order languages

All the symbols are strings of some given alphabet $\mathcal{A}$.

All the symbols are strings of some given alphabet $\mathcal{A}$.

The class of arities $A = \{\varnothing,\ o,\ oo, \ldots\}$ was defined inductively earlier.

First-order languages

All the symbols are strings of some given alphabet $\mathcal{A}$.

The class of arities $A = \{\varnothing,\ o,\ oo, \ldots\}$ was defined inductively earlier.

A first-order language $\mathcal{L}^1$ is a quintuple

$$< Log, Var, Con, Term, Form >$$

where $Log = \{(,\ ),\ \neg,\ \supset,\ \forall,\ =\}$ is the class of logical constants, Var is the infinite class of variables defined inductively, and $Con = N \cup P = \bigcup_{a \in A} P_a \cup \bigcup_{a \in A} N_a$ is the class of non-logical constants containing all the classes P_a of a-ary predicates and N_a of a-ary name functors.

First-order languages

All the symbols are strings of some given alphabet $\mathcal{A}$.

The class of arities $A = \{\varnothing,\ o,\ oo, \ldots\}$ was defined inductively earlier.

A first-order language $\mathcal{L}^1$ is a quintuple

$$< Log, Var, Con, Term, Form >$$

where $Log = \{(,\ ),\ \neg,\ \supset,\ \forall,\ =\}$ is the class of logical constants, Var is the infinite class of variables defined inductively, and $Con = N \cup P = \bigcup_{a \in A} P_a \cup \bigcup_{a \in A} N_a$ is the class of non-logical constants containing all the classes P_a of a-ary predicates and N_a of a-ary name functors.

It is assumed that for $a_i \neq a_j \in A$, $N_{a_i} \cap N_{a_j} = P_{a_i} \cap P_{a_j} = \emptyset$ and $N \cap P = \emptyset$.

Terms and a-tuples of terms

Terms and a-tuples of terms

The class of a-tuples of terms $a \in A$ is $T(a)$.

The class of a-tuples of terms $a \in A$ is $T(a)$.

The simultaneous inductive definition of the classes $Term$ and $T(a)$:

Terms and a-tuples of terms

The class of a-tuples of terms $a \in A$ is $T(a)$.

The simultaneous inductive definition of the classes $Term$ and $T(a)$:

1. $Var \subseteq Term$
2. $T(\varnothing) = \{\varnothing\}$
3. $(s \in T(a)\ \&\ t \in Term) \Rightarrow \ulcorner s(t) \urcorner \in T(ao)$
4. $(\varphi \in N_a\ \&\ s \in T(a)) \Rightarrow \ulcorner \varphi s \urcorner \in Term$

Formulas

Formulas

1. $\pi \in P_a \;\&\; s \in T(a) \Rightarrow \ulcorner \pi s \urcorner \in Form$
2. $s, t \in Term \Rightarrow \ulcorner s = t \urcorner \in Form$
3. $A \in Form \Rightarrow \ulcorner \neg A \urcorner \in Form$
4. $A, B \in Form \Rightarrow \ulcorner A \supset B \urcorner \in Form$
5. $A \in Form \;\&\; x \in Var \Rightarrow \ulcorner \forall x A \urcorner \in Form$

Formulas

1. $\pi \in P_a \ \& \ s \in T(a) \Rightarrow \ulcorner \pi s \urcorner \in Form$
2. $s, t \in Term \Rightarrow \ulcorner s = t \urcorner \in Form$
3. $A \in Form \Rightarrow \ulcorner \neg A \urcorner \in Form$
4. $A, B \in Form \Rightarrow \ulcorner A \supset B \urcorner \in Form$
5. $A \in Form \ \& \ x \in Var \Rightarrow \ulcorner \forall x A \urcorner \in Form$

Atomic formulas are the formulas generated by the rules 1. and 2.

Formulas

1. $\pi \in P_a \ \& \ s \in T(a) \Rightarrow \ulcorner \pi s \urcorner \in Form$
2. $s, t \in Term \Rightarrow \ulcorner s = t \urcorner \in Form$
3. $A \in Form \Rightarrow \ulcorner \neg A \urcorner \in Form$
4. $A, B \in Form \Rightarrow \ulcorner A \supset B \urcorner \in Form$
5. $A \in Form \ \& \ x \in Var \Rightarrow \ulcorner \forall x A \urcorner \in Form$

Atomic formulas are the formulas generated by the rules 1. and 2.

Other logical constants ($\vee$, $\wedge$, $\equiv$, $\exists$) are introduced by abbreviation conventions.

Formulas

1. $\pi \in P_a \;\&\; s \in T(a) \Rightarrow \ulcorner \pi s \urcorner \in Form$
2. $s, t \in Term \Rightarrow \ulcorner s = t \urcorner \in Form$
3. $A \in Form \Rightarrow \ulcorner \neg A \urcorner \in Form$
4. $A, B \in Form \Rightarrow \ulcorner A \supset B \urcorner \in Form$
5. $A \in Form \;\&\; x \in Var \Rightarrow \ulcorner \forall x A \urcorner \in Form$

Atomic formulas are the formulas generated by the rules 1. and 2.

Other logical constants ($\vee$, $\wedge$, $\equiv$, $\exists$) are introduced by abbreviation conventions.

Be $A, B \in Form$. B is a subformula of A iff A is of the form uBv $(u, v \in \mathcal{A}^{\circ})$.

Formulas

1. $\pi \in P_a \ \& \ s \in T(a) \Rightarrow \ulcorner \pi s \urcorner \in Form$
2. $s, t \in Term \Rightarrow \ulcorner s = t \urcorner \in Form$
3. $A \in Form \Rightarrow \ulcorner \neg A \urcorner \in Form$
4. $A, B \in Form \Rightarrow \ulcorner A \supset B \urcorner \in Form$
5. $A \in Form \ \& \ x \in Var \Rightarrow \ulcorner \forall x A \urcorner \in Form$

Atomic formulas are the formulas generated by the rules 1. and 2.

Other logical constants ($\vee$, $\wedge$, $\equiv$, $\exists$) are introduced by abbreviation conventions.

Be $A, B \in Form$. B is a subformula of A iff A is of the form uBv ($u, v \in \mathcal{A}^\circ$).

If $x \in Var$ and $A \in Form$, an occurrence of x in A is a bound occurrence of x in A iff it lies in a subformula of A of the form $\forall x B$. Other occurrences are called free occurrences.

Some further auxiliary notions

Some further auxiliary notions

A term is <u>open</u> iff at least one variable is a substring of it; otherways it is <u>closed</u>.

Some further auxiliary notions

A term is <u>open</u> iff at least one variable is a substring of it; otherways it is <u>closed</u>.

A formula is <u>open</u> if it contains at least one free occurrence of a variable; otherwise it is <u>closed</u>. Closed formulas are called <u>sentences</u>.

Some further auxiliary notions

A term is <u>open</u> iff at least one variable is a substring of it; otherways it is <u>closed</u>.

A formula is <u>open</u> if it contains at least one free occurrence of a variable; otherwise it is <u>closed</u>. Closed formulas are called <u>sentences</u>.

A formula A is <u>free from</u> the variable x iff x has no free occurrences in A. $\Gamma \subseteq Form$ is <u>free from</u> x if each member of it is.

Some further auxiliary notions

A term is <u>open</u> iff at least one variable is a substring of it; otherways it is <u>closed</u>.

A formula is <u>open</u> if it contains at least one free occurrence of a variable; otherwise it is <u>closed</u>. Closed formulas are called <u>sentences</u>.

A formula A is <u>free from</u> the variable x iff x has no free occurrences in A. $\Gamma \subseteq Form$ is <u>free from</u> x if each member of it is.

Be $A \in Form,\ x, y \in Var$. y is <u>substitutable</u> for x in A iff for every subformula of A of the form $\forall y B$, B is free from x.

$t \in Term$ is <u>substitutable</u> for x in A iff every variable occurring in t is substitutable. If t is substitutable for x in A, then $A^{t/x}$ denotes (in the metalanguage) the formula obtained from A substituting t for every free occurrence of x in A.

The quantification calculus (**QC**) 1: the axioms

Given a first-order language $\mathcal{L}^1$, the logical axioms (basic formulas) are defined by the help of the following schemes:

The quantification calculus (**QC**) 1: the axioms

Given a first-order language $\mathcal{L}^1$, the logical axioms (basic formulas) are defined by the help of the following schemes:

(B1) $(A \supset (B \supset A))$

(B2) $((A \supset (B \supset C)) \supset ((A \supset B) \supset (A \supset C)))$

(B3) $((\neg B \supset \neg A) \supset (A \supset B))$

Given a first-order language $\mathcal{L}^1$, the logical axioms (basic formulas) are defined by the help of the following schemes:

(B1) $(A \supset (B \supset A))$

(B2) $((A \supset (B \supset C) \supset ((A \supset B) \supset (A \supset C)))$

(B3) $((\neg B \supset \neg A) \supset (A \supset B))$

(B4) $(\forall x A \supset A^{t/x})$

(B5) $(\forall x(A \supset B) \supset (\forall x A \supset \forall x B))$

(B6) $(A \supset \forall x A)$ provided that A is free from x

The quantification calculus (**QC**) 1: the axioms

Given a first-order language $\mathcal{L}^1$, the logical axioms (basic formulas) are defined by the help of the following schemes:

(B1) $(A \supset (B \supset A))$

(B2) $((A \supset (B \supset C) \supset ((A \supset B) \supset (A \supset C)))$

(B3) $((\neg B \supset \neg A) \supset (A \supset B))$

(B4) $(\forall x A \supset A^{t/x})$

(B5) $(\forall x(A \supset B) \supset (\forall x A \supset \forall x B))$

(B6) $(A \supset \forall x A)$ provided that A is free from x

(B7) $(\mathfrak{x} = \mathfrak{x})$

(B8) $((x = y) \supset (A^{x/z} \supset A^{y/z}))$

The quantification calculus (**QC**) 1: the axioms

Given a first-order language $\mathcal{L}^1$, the logical axioms (basic formulas) are defined by the help of the following schemes:

(B1) $(A \supset (B \supset A))$

(B2) $((A \supset (B \supset C) \supset ((A \supset B) \supset (A \supset C)))$

(B3) $((\neg B \supset \neg A) \supset (A \supset B))$

(B4) $(\forall x A \supset A^{t/x})$

(B5) $(\forall x(A \supset B) \supset (\forall x A \supset \forall x B))$

(B6) $(A \supset \forall x A)$ provided that A is free from x

(B7) $(\mathfrak{x} = \mathfrak{x})$

(B8) $((x = y) \supset (A^{x/z} \supset A^{y/z}))$

The class BF of logical axioms is defined inductively:

The quantification calculus (**QC**) 1: the axioms

Given a first-order language $\mathcal{L}^1$, the logical axioms (basic formulas) are defined by the help of the following schemes:

(B1) $(A \supset (B \supset A))$

(B2) $((A \supset (B \supset C) \supset ((A \supset B) \supset (A \supset C)))$

(B3) $((\neg B \supset \neg A) \supset (A \supset B))$

(B4) $(\forall x A \supset A^{t/x})$

(B5) $(\forall x(A \supset B) \supset (\forall x A \supset \forall x B))$

(B6) $(A \supset \forall x A)$ provided that A is free from x

(B7) $(\mathfrak{x} = \mathfrak{x})$

(B8) $((x = y) \supset (A^{x/z} \supset A^{y/z}))$

The class BF of <u>logical axioms</u> is defined inductively:

i If we substitute formulas for A, B, C , variables for x, y, z and terms for t of $\mathcal{L}^1$ in the above schemes, we get members of BF.

Given a first-order language $\mathcal{L}^1$, the logical axioms (basic formulas) are defined by the help of the following schemes:

(B1) $(A \supset (B \supset A))$

(B2) $((A \supset (B \supset C) \supset ((A \supset B) \supset (A \supset C)))$

(B3) $((\neg B \supset \neg A) \supset (A \supset B))$

(B4) $(\forall x A \supset A^{t/x})$

(B5) $(\forall x(A \supset B) \supset (\forall x A \supset \forall x B))$

(B6) $(A \supset \forall x A)$ provided that A is free from x

(B7) $(\mathfrak{x} = \mathfrak{x})$

(B8) $((x = y) \supset (A^{x/z} \supset A^{y/z}))$

The class BF of logical axioms is defined inductively:

i If we substitute formulas for A, B, C , variables for x, y, z and terms for t of $\mathcal{L}^1$ in the above schemes, we get members of BF.

ii If $A \in BF$ and $x \in Var$, then $\ulcorner \forall x A \urcorner \in BF$.

QC 2: deducibility and some metatheorems

QC 2: deducibility and some metatheorems

Base for the inductive definition of $\Gamma \vdash A$: if $A \in \Gamma \cup BF$, then $\Gamma \vdash A$. Inductive rule is detachment: if $\Gamma \vdash A$ and $\Gamma \vdash A \supset B$, then $\Gamma \vdash B$.

QC 2: deducibility and some metatheorems

Base for the inductive definition of $\Gamma \vdash A$: if $A \in \Gamma \cup BF$, then $\Gamma \vdash A$. Inductive rule is detachment: if $\Gamma \vdash A$ and $\Gamma \vdash A \supset B$, then $\Gamma \vdash B$.

- Deduction Theorem: If $\Gamma \cup \{A\} \vdash C$, then $\Gamma \vdash A \supset C$.

QC 2: deducibility and some metatheorems

Base for the inductive definition of $\Gamma \vdash A$: if $A \in \Gamma \cup BF$, then $\Gamma \vdash A$. Inductive rule is detachment: if $\Gamma \vdash A$ and $\Gamma \vdash A \supset B$, then $\Gamma \vdash B$.

- Deduction Theorem: If $\Gamma \cup \{A\} \vdash C$, then $\Gamma \vdash A \supset C$.
- Cut: If $\Gamma \vdash A$ and $\Gamma' \cup \{A\} \vdash B$ then $\Gamma \cup \Gamma' \vdash B$.

QC 2: deducibility and some metatheorems

Base for the inductive definition of $\Gamma \vdash A$: if $A \in \Gamma \cup BF$, then $\Gamma \vdash A$. Inductive rule is detachment: if $\Gamma \vdash A$ and $\Gamma \vdash A \supset B$, then $\Gamma \vdash B$.

- Deduction Theorem: If $\Gamma \cup \{A\} \vdash C$, then $\Gamma \vdash A \supset C$.
- Cut: If $\Gamma \vdash A$ and $\Gamma' \cup \{A\} \vdash B$ then $\Gamma \cup \Gamma' \vdash B$.
- Universal generalization: If $\Gamma \vdash A$ and Γ is free from x, then $\Gamma \vdash \forall x A$.

QC 2: deducibility and some metatheorems

Base for the inductive definition of $\Gamma \vdash A$: if $A \in \Gamma \cup BF$, then $\Gamma \vdash A$. Inductive rule is detachment: if $\Gamma \vdash A$ and $\Gamma \vdash A \supset B$, then $\Gamma \vdash B$.

- Deduction Theorem: If $\Gamma \cup \{A\} \vdash C$, then $\Gamma \vdash A \supset C$.
- Cut: If $\Gamma \vdash A$ and $\Gamma' \cup \{A\} \vdash B$ then $\Gamma \cup \Gamma' \vdash B$.
- Universal generalization: If $\Gamma \vdash A$ and Γ is free from x, then $\Gamma \vdash \forall x A$.
- Universal generalization 2.: If $t \in N_{\varnothing}$ s.t. it occurs neither in A nor in the members of Γ and $\Gamma \vdash A^{t/x}$ then $\Gamma \vdash \forall x A$.

QC 2: deducibility and some metatheorems

Base for the inductive definition of $\Gamma \vdash A$: if $A \in \Gamma \cup BF$, then $\Gamma \vdash A$. Inductive rule is detachment: if $\Gamma \vdash A$ and $\Gamma \vdash A \supset B$, then $\Gamma \vdash B$.

- Deduction Theorem: If $\Gamma \cup \{A\} \vdash C$, then $\Gamma \vdash A \supset C$.
- Cut: If $\Gamma \vdash A$ and $\Gamma' \cup \{A\} \vdash B$ then $\Gamma \cup \Gamma' \vdash B$.
- Universal generalization: If $\Gamma \vdash A$ and Γ is free from x, then $\Gamma \vdash \forall x A$.
- Universal generalization 2.: If $t \in N_{\varnothing}$ s.t. it occurs neither in A nor in the members of Γ and $\Gamma \vdash A^{t/x}$ then $\Gamma \vdash \forall x A$.

A definition: If $A \in Form$ and the variables having free occurrences in A are $x_1, x_2, \ldots x_n$, then the universal closure of A is the formula $\forall x_1 \forall x_2 \ldots \forall x_n A$.

Given any logical calculus Σ in a language $\mathcal{L}$ and a class Γ of formulas of $\mathcal{L}$, the class of the consequences of Γ is the class

$$Cns(\Gamma) = \{A \in Form : \Gamma \vdash_{\Sigma} A\}$$

Given any logical calculus Σ in a language $\mathcal{L}$ and a class Γ of formulas of $\mathcal{L}$, the class of the consequences of Γ is the class

$$Cns(\Gamma) = \{A \in Form : \Gamma \vdash_{\Sigma} A\}$$

Γ is <u>inconsistent</u> if $Cns(\Gamma) = Form$, <u>consistent</u> in the other case.

Given any logical calculus Σ in a language $\mathcal{L}$ and a class Γ of formulas of $\mathcal{L}$, the class of the consequences of Γ is the class

$$Cns(\Gamma) = \{A \in Form : \Gamma \vdash_\Sigma A\}$$

Γ is inconsistent if $Cns(\Gamma) = Form$, consistent in the other case.

In first-order logic, Γ is consistent iff there is no $A \in Form$ s. t. both $\Gamma \vdash A$ and $\Gamma \vdash \neg A$.

Given any logical calculus Σ in a language $\mathcal{L}$ and a class Γ of formulas of $\mathcal{L}$, the class of the consequences of Γ is the class

$$Cns(\Gamma) = \{A \in Form : \Gamma \vdash_{\Sigma} A\}$$

Γ is <u>inconsistent</u> if $Cns(\Gamma) = Form$, <u>consistent</u> in the other case.

In first-order logic, Γ is consistent iff there is no $A \in Form$ s. t. both $\Gamma \vdash A$ and $\Gamma \vdash \neg A$.

The pair $T =< \mathcal{L}^1, \Gamma >$ is a <u>first-order theory</u> if $\mathcal{L}^1$ is a first-order language and Γ is a class of its *closed* formulas (called *axioms* of T).

Given any logical calculus Σ in a language $\mathcal{L}$ and a class Γ of formulas of $\mathcal{L}$, the class of the consequences of Γ is the class

$$Cns(\Gamma) = \{A \in Form : \Gamma \vdash_{\Sigma} A\}$$

Γ is inconsistent if $Cns(\Gamma) = Form$, consistent in the other case.

In first-order logic, Γ is consistent iff there is no $A \in Form$ s. t. both $\Gamma \vdash A$ and $\Gamma \vdash \neg A$.

The pair $T =< \mathcal{L}^1, \Gamma >$ is a first-order theory if $\mathcal{L}^1$ is a first-order language and Γ is a class of its *closed* formulas (called *axioms* of T).

The theorems of T are the members of $Cns(\Gamma)$. T is said consistent resp. inconsistent if Γ is consistent resp. inconsistent.

The first-order theory of canonical calculi: $\mathbf{CC}^*$

We construct $\mathbf{CC}^*$ as a first-order theory, whose intended interpretation is that its terms denote strings, its propositions say that some strings are words, canonical calculi, certain calculi derive certain strings, certain numerals are autonomous numerals etc. It is expected to derive only true propositions of this type.

The first-order theory of canonical calculi: $\mathbf{CC}^*$

We construct $\mathbf{CC}^*$ as a first-order theory, whose intended interpretation is that its terms denote strings, its propositions say that some strings are words, canonical calculi, certain calculi derive certain strings, certain numerals are autonomous numerals etc. It is expected to derive only true propositions of this type.

That is, we want it to do something very similar to the hypercalculus $\mathbf{H}_3$. (See 3rd October presentation.) In practice, we simply rewrite $\mathbf{H}_3$ in the form of a first-order theory. But this first-order theory can be reconstructed as a canonical calculus Σ^* again. There will be a mutual mirroring relationship between $\mathbf{H}_3$ and Σ^*, which we will use to prove metatheorems.

The language $\mathcal{L}^{1*}$

The language $\mathcal{L}^{1*}$

$\mathbf{H}_3$ derives strings like Ka, Wb, aDb, aGb, Aa with the intended meanings 'a is a calculus', . . . 'a is an autonomous number'. We want $\mathbf{CC}^*$ to prove formulas like $K(a), \ldots A(a)$ just in the same case.

$\mathbf{H}_3$ derives strings like Ka, Wb, aDb, aGb, Aa with the intended meanings 'a is a calculus', …'a is an autonomous number'. We want $\mathbf{CC}^*$ to prove formulas like $K(a), \ldots A(a)$ just in the same case.

The language of $\mathbf{CC}^*$ is the first-order language $\mathcal{L}^{1*}$.
Non-logical components :

The language $\mathcal{L}^{1*}$

$\mathbf{H}_3$ derives strings like Ka, Wb, aDb, aGb, Aa with the intended meanings 'a is a calculus', …'a is an autonomous number'. We want $\mathbf{CC}^*$ to prove formulas like $K(a), \ldots A(a)$ just in the same case.

The language of $\mathbf{CC}^*$ is the first-order language $\mathcal{L}^{1*}$.
Non-logical components :

- $N_\varnothing = \{\vartheta,\ \alpha,\ \beta,\ \xi,\ \gg,\ *\}$
  ϑ denotes the empty string, the other name constants denote (autonymously) the letters of $\mathcal{A}_{cc}$.

The language $\mathcal{L}^{1*}$

$\mathbf{H}_3$ derives strings like Ka, Wb, aDb, aGb, Aa with the intended meanings 'a is a calculus', …'a is an autonomous number'. We want $\mathbf{CC}^*$ to prove formulas like $K(a), \ldots A(a)$ just in the same case.

The language of $\mathbf{CC}^*$ is the first-order language $\mathcal{L}^{1*}$.
Non-logical components :

- $N_{\varnothing} = \{\vartheta,\ \alpha,\ \beta,\ \xi,\ \gg,\ *\}$
  ϑ denotes the empty string, the other name constants denote (autonymously) the letters of $\mathcal{A}_{cc}$.
- $N_{oo} = \{\varnothing\}$
  The empty string denotes concatenation (and we omit the parentheses around its arguments), i.e., we write the concatenation of the strings x and y as xy.

The language $\mathcal{L}^{1*}$ (continuation)

The auxiliary letters of the hypercalculi $\mathbf{H}_1$ – $\mathbf{H}_3$ become predicates and we keep the intended meanings:

The auxiliary letters of the hypercalculi $\mathbf{H}_1$ – $\mathbf{H}_3$ become predicates and we keep the intended meanings:

- $P_o = \{I,\ L,\ V,\ W,\ T,\ R,\ K,\ A\}$

The auxiliary letters of the hypercalculi $\mathbf{H}_1$ – $\mathbf{H}_3$ become predicates and we keep the intended meanings:

- $P_o = \{I,\ L,\ V,\ W,\ T,\ R,\ K,\ A\}$
- $P_{oo} = \{D,\ F,\ G\}$

The auxiliary letters of the hypercalculi $\mathbf{H}_1$ – $\mathbf{H}_3$ become predicates and we keep the intended meanings:

- $P_o = \{I,\ L,\ V,\ W,\ T,\ R,\ K,\ A\}$
- $P_{oo} = \{D,\ F,\ G\}$
- $P_{oooo} = \{S\}$
  $S(v)(u)(y)(x)$: if we substitute the word y for the variable x, we get the string v from the string u.

The language $\mathcal{L}^{1*}$ (continuation)

The auxiliary letters of the hypercalculi $\mathbf{H}_1$ – $\mathbf{H}_3$ become predicates and we keep the intended meanings:

- $P_o = \{I,\ L,\ V,\ W,\ T,\ R,\ K,\ A\}$
- $P_{oo} = \{D,\ F,\ G\}$
- $P_{oooo} = \{S\}$
  $S(v)(u)(y)(x)$: if we substitute the word y for the variable x, we get the string v from the string u.

Logical constants, variables (let us write them as $\mathfrak{x}, \mathfrak{x}_1, \ldots$), the syntax of terms and formulas are like in any other first-order language. The intended universe (the domain of the variables) is the class of $\mathcal{A}_{cc}$-strings.

The axioms of $\mathbf{CC}^*$: the language radix-axioms

The axioms of $\mathbf{CC}^*$: the language radix-axioms

First group: The $\mathcal{A}_{cc}$-strings build a language radix (12th September presentation) or in the language of algebra, the free monoid on $\mathcal{A}_{cc}$. In some details:

The axioms of $\mathbf{CC}^*$: the language radix-axioms

First group: The $\mathcal{A}_{cc}$-strings build a language radix (12th September presentation) or in the language of algebra, the free monoid on $\mathcal{A}_{cc}$. In some details:

- The empty string is different from the letters (five axioms).

First group: The $\mathcal{A}_{cc}$-strings build a language radix (12th September presentation) or in the language of algebra, the free monoid on $\mathcal{A}_{cc}$. In some details:

- The empty string is different from the letters (five axioms).
- Strings ending with different letters are different (ten axioms).

The axioms of $\mathbf{CC}^*$: the language radix-axioms

First group: The $\mathcal{A}_{cc}$-strings build a language radix (12th September presentation) or in the language of algebra, the free monoid on $\mathcal{A}_{cc}$. In some details:

- The empty string is different from the letters (five axioms).
- Strings ending with different letters are different (ten axioms).
- Five more axioms about strings:
  1. $\forall\mathfrak{x}(\mathfrak{x}\vartheta = \vartheta\mathfrak{x} = \mathfrak{x})$
  2. $\forall\mathfrak{x}\forall\mathfrak{x}_1(\mathfrak{x}\mathfrak{x}_1 = \vartheta \supset (\mathfrak{x} = \vartheta \wedge \mathfrak{x}_1 = \vartheta)$
  3. $\forall\mathfrak{x}_1\exists\mathfrak{x}(\mathfrak{x}_1 \neq \vartheta \supset (\mathfrak{x}_1 = \mathfrak{x}\alpha \vee \mathfrak{x}_1 = \mathfrak{x}\beta \vee \mathfrak{x}_1 = \mathfrak{x}\xi \vee \mathfrak{x}_1 = \mathfrak{x} \gg \vee \mathfrak{x}_1 = \mathfrak{x}*))$
  4. $\forall\mathfrak{x}\forall\mathfrak{x}_1\forall\mathfrak{x}_2(\mathfrak{x}_1\mathfrak{x} = \mathfrak{x}_2\mathfrak{x} \supset \mathfrak{x}_1 = \mathfrak{x}_2)$

The axioms of $\mathbf{CC}^*$: the calculus-axioms

Second group: The calculus-axioms
To obtain the axioms about calculi, we can simply translate the 34 rules of the hypercalculus $\mathbf{H}_3$ into $\mathcal{L}^{1*}$ -propositions. The rules of the translation are the following:

Second group: The calculus-axioms

To obtain the axioms about calculi, we can simply translate the 34 rules of the hypercalculus $\mathbf{H}_3$ into $\mathcal{L}^{1*}$ -propositions. The rules of the translation are the following:

- The auxiliary letters of $\mathbf{H}_3$ are reinterpreted as predicates of $\mathcal{L}^{1*}$; their arguments are written after them and they are put in parentheses. E.g., instead of xDy we write $D(x)(y)$.

Second group: The calculus-axioms

To obtain the axioms about calculi, we can simply translate the 34 rules of the hypercalculus $\mathbf{H}_3$ into $\mathcal{L}^{1*}$ -propositions. The rules of the translation are the following:

- The auxiliary letters of $\mathbf{H}_3$ are reinterpreted as predicates of $\mathcal{L}^{1*}$; their arguments are written after them and they are put in parentheses. E.g., instead of xDy we write $D(x)(y)$.
- The concatenations of strings in $\mathbf{H}_3$ are reinterpreted as applications of the concatenation functor; it is again a reinterpretation of the same notation only

Second group: The calculus-axioms

To obtain the axioms about calculi, we can simply translate the 34 rules of the hypercalculus $\mathbf{H}_3$ into $\mathcal{L}^{1*}$ -propositions. The rules of the translation are the following:

- The auxiliary letters of $\mathbf{H}_3$ are reinterpreted as predicates of $\mathcal{L}^{1*}$; their arguments are written after them and they are put in parentheses. E.g., instead of xDy we write $D(x)(y)$.
- The concatenations of strings in $\mathbf{H}_3$ are reinterpreted as applications of the concatenation functor; it is again a reinterpretation of the same notation only
- The letters of $\mathcal{A}_{cc}$ are reinterpreted as their own names; at places where the empty string occurs as an argument of some predicate, it is substituted by its name ϑ.

Second group: The calculus-axioms

To obtain the axioms about calculi, we can simply translate the 34 rules of the hypercalculus $\mathbf{H}_3$ into $\mathcal{L}^{1*}$ -propositions. The rules of the translation are the following:

- The auxiliary letters of $\mathbf{H}_3$ are reinterpreted as predicates of $\mathcal{L}^{1*}$; their arguments are written after them and they are put in parentheses. E.g., instead of xDy we write $D(x)(y)$.
- The concatenations of strings in $\mathbf{H}_3$ are reinterpreted as applications of the concatenation functor; it is again a reinterpretation of the same notation only
- The letters of $\mathcal{A}_{cc}$ are reinterpreted as their own names; at places where the empty string occurs as an argument of some predicate, it is substituted by its name ϑ.
- Calculus variables $x, y, z, \dots$ are substituted by the $\mathcal{L}^{1*}$-variables $\mathfrak{x}, \mathfrak{x}_1, \dots$.

CC*'s calculus axioms (continuation)

CC*'s calculus axioms (continuation)

- The arrows are substituted by the conditional sign '$\supset$' and formulas of the form $\ulcorner A \supset B \supset C \urcorner$ are understood as $\ulcorner (A \supset (B \supset C)) \urcorner$.

CC*'s calculus axioms (continuation)

- The arrows are substituted by the conditional sign '$\supset$' and formulas of the form $\ulcorner A \supset B \supset C \urcorner$ are understood as $\ulcorner (A \supset (B \supset C)) \urcorner$.
- The open formulas obtained by the previous rules are substituted by their universal closure.

$\mathbf{CC}^*$'s calculus axioms (continuation)

- The arrows are substituted by the conditional sign '$\supset$' and formulas of the form $\ulcorner A \supset B \supset C \urcorner$ are understood as $\ulcorner (A \supset (B \supset C)) \urcorner$.
- The open formulas obtained by the previous rules are substituted by their universal closure.

The axioms of $\mathbf{CC}^*$ are the 20 language radix-axioms plus the 34 axioms obtained from the rules of $\mathbf{H}_3$. E.g., the rules 12. and 13. of $\mathbf{H}_3$ (defining the extension of K) become the following axioms:

- $\forall \mathfrak{x}(R(\mathfrak{x}) \supset K(\mathfrak{x}))$
- $\forall \mathfrak{x} \forall \mathfrak{x}_1(K(\mathfrak{x}) \supset R(\mathfrak{x}_1) \supset K(\mathfrak{x} * \mathfrak{x}_1))$

$\mathbf{CC}^*$'s calculus axioms (continuation)

- The arrows are substituted by the conditional sign '$\supset$' and formulas of the form $\ulcorner A \supset B \supset C \urcorner$ are understood as $\ulcorner (A \supset (B \supset C)) \urcorner$.
- The open formulas obtained by the previous rules are substituted by their universal closure.

The axioms of $\mathbf{CC}^*$ are the 20 language radix-axioms plus the 34 axioms obtained from the rules of $\mathbf{H}_3$. E.g., the rules 12. and 13. of $\mathbf{H}_3$ (defining the extension of K) become the following axioms:

- $\forall \mathfrak{x}(R(\mathfrak{x}) \supset K(\mathfrak{x}))$
- $\forall \mathfrak{x} \forall \mathfrak{x}_1 (K(\mathfrak{x}) \supset R(\mathfrak{x}_1) \supset K(\mathfrak{x} * \mathfrak{x}_1))$

The above rules of translation apply to any string derivable in $\mathbf{H}_3$. Let us denote the translation of the string f into a $\mathcal{L}^{1*}$-formula $Tr(f)$.

$\mathbf{CC}^*$ is now the theory in the language $\mathcal{L}^{1*}$, defined by the set Γ^* of 20 language radix axioms and 34 calculus axioms described above. We want to generate its theorems by the canonical calculus Σ^*.

$\mathbf{CC}^*$ is now the theory in the language $\mathcal{L}^{1*}$, defined by the set Γ^* of 20 language radix axioms and 34 calculus axioms described above. We want to generate its theorems by the canonical calculus Σ^*.

The 54 axioms of $\mathbf{CC}^*$ will be zero input rules of Σ^*. But in order to derive theorems from them, we need to build in the first-order logic of $\mathcal{L}^{1*}$ into Σ^*: first the syntax of the language, then the logical axioms (basic formulas) and derivation rules of first-order logic.

The theory $\mathbf{CC}^*$ and the calculus Σ^*

$\mathbf{CC}^*$ is now the theory in the language $\mathcal{L}^{1*}$, defined by the set Γ^* of 20 language radix axioms and 34 calculus axioms described above. We want to generate its theorems by the canonical calculus Σ^*.

The 54 axioms of $\mathbf{CC}^*$ will be zero input rules of Σ^*. But in order to derive theorems from them, we need to build in the first-order logic of $\mathcal{L}^{1*}$ into Σ^*: first the syntax of the language, then the logical axioms (basic formulas) and derivation rules of first-order logic.

We introduce auxiliary letters/abbreviations for the syntactic notions of $\mathcal{L}^{1*}$. These are written in boldface to distinguish them from the partly overlapping auxiliary letters of $\mathbf{H}_3$ (which are now non-logical constants of $\mathcal{L}^{1*}$).

The syntax of $\mathcal{L}^{1*}$ in the calculus

The syntax of $\mathcal{L}^{1*}$ in the calculus

The syntactic rules of Σ^* derive from the following considerations:
Indexes and **V**ariables of $\mathcal{L}^{1*}$ are generated in the usual way.
The language has six **N**ame constants: (ϑ) for the empty string, letters, arrow and asterisk. Variables and the name constants are **T**erms and the concatenation of two terms is a term again.
There are eight monadic **P**redicates. Applying a monadic predicate to a term yields a formula. The usual rules of formula construction then follow.

The syntactic rules of Σ^* derive from the following considerations:
Indexes and **V**ariables of $\mathcal{L}^{1*}$ are generated in the usual way. The language has six **N**ame constants: (ϑ) for the empty string, letters, arrow and asterisk. Variables and the name constants are **T**erms and the concatenation of two terms is a term again. There are eight monadic **P**redicates. Applying a monadic predicate to a term yields a formula. The usual rules of formula construction then follow.

We need application rules for the identity, the three two-place predicates and the substitution *in the object calculus* (four-place predicate). Then we need an inductive definition for the relation that a string is free of a variable (**FR**).

First-order logic as part of Σ^*

We must inductively define **S**ubstitution *in* Σ^*. Then we can include the logical axioms (**BF**s) and the detachment rule, too.

First-order logic as part of Σ^*

We must inductively define **S**ubstitution *in* Σ^*. Then we can include the logical axioms (**BF**s) and the detachment rule, too.

More details see on pp. 76-81. of the textbook. The whole calculus Σ^* consists of 115 rules.

We must inductively define **S**ubstitution *in* Σ^*. Then we can include the logical axioms (**BF**s) and the detachment rule, too.

More details see on pp. 76-81. of the textbook. The whole calculus Σ^* consists of 115 rules.

NO CLASS NEXT WEEK (14.11)!