A Flaw in Separation Axioms

A New Look at Axiomatic Set Theory

Hannes Hutzelmeyer

Summary

The author has developed an approach to logics that comprises, but also goes beyond predicate logic. The **FUME** method contains two tiers of precise languages: object-language **Funcish** and metalanguage **Mencish**. It allows for a very wide application in mathematics from geometry, number theory, recursion theory and axiomatic set theory with first-order logic, to higher-order logic theory of real numbers etc. . The conventional treatment of axiomatic set theory (ZFC) is replaced by the abstract calcule sigma so that certain shortcomings can be avoided by the use of Funcish-Mencish language hierarchy:

- precise talking about **formula** strings necessitates a formalized metalanguage
- talking about **open arities**, general tuples, open dimensions of spaces, finite systems of open cardinality and so on necessitates a formalized metalanguage. 'dot dot dot … ' just will not do
- the **Axiom of Infinity** is generalized in order to allow for certain other infinite sets besides the natural number representation according to von Neumann (i.e. general recursion)
- the **Axioms of Separation** are modified as it seems more convenient
- there are only enumerably many properties that can be constructed from **formula** strings, as these are finite strings of characters from a finite alphabet; this should be kept in mind in connection with the **Axioms of Replacement**
- a new look at **Cantor's continuum hypothesis** in abstract axiomatic set theory leads to the question of so-called basis-incompleteness versus proof-incompleteness
- the **Axioms of Separation** seem to have a flaw; there is a caveat for axiomatic set theory.

And this leads to another serious problem. It is questionable if one can introduce relation-constants (including the unary case of property-constants) in a proper fashion in axiomatic set theory.

Contact: Hutzelmeyer@pai.de
https://pai.de

1 FUME-method object-language and metalanguage

Axiomatic set theory is commonly claimed as a foundation system of all mathematics or at least most mathematics. The somewhat surprising fact is that only one sort and only first-order-logic (FOL) is needed in axiomatic set theory, or what is usually called **predicate-logic with identity**

The author has put forward a **precise** system of **object-language** and **metalanguage** that overcomes certain difficulties of predicate logic and that extends to a full theory of **types** . In order to describe an **object-language** one also needs a **metalanguage**. According to the author's principle metalanguage has to be absolutely precise as well, normal English will not do. The **FUME**-method contains at least three levels of language:

Funcish	object-language	formalized precise.
Mencish	metalanguage	formalized precise
English	supralanguage	natural

'Calcule' is the name given to a mathematical system with the precise language-metalanguage method Funcish-Mencish . 'Calcule' is an expression coined by the author in order to avoid confusion. The word 'calculus' is conventionally used for real number mathematics and various logical systems. As a German translation 'Kalkul' is proposed for 'calcule' versus conventional 'Kalkül' that usually corresponds to 'calculus'.

A **concrete** calcule talks about a **codex** of concrete **individuals** (given as strings of characters) and concrete **functions** and **relations** that can be realized by 'machines' (called calculators).

An **abstract** calcule talks about **nothing**. It only says: if some entities exist with such and such properties they also have certain other properties. Essentially there are only 'if-then' statements. E.g. 'if there are entities that obey the Euclid axioms the following sentence is true for these entities' .

Calcules with first-order logic FOL are called **haplo-calcules** , calcules with higher-order logic HOL for a theory of types are called **hypso-calcules** . As the following will only deal with first-order logic the expression 'calcule' will always mean 'haplo-calcule' . An **abstract** calcule is based on a finite count or on enumerably many axioms as opposed to a **concrete** calcule whose foundation can be put into practice by a machine. ***Axiom*** strings are certain ***sentence*** strings, they can also be provided with a metalingual ***Axiom*** **mater** (rather than the usual 'scheme' or 'schema', as the expression ***scheme*** has a different meaning in Mencish), that produce enumerably many ***Axiom*** strings.

Calcules are given names based on the Greek **sort** names. Small letters are used for abstract calcules e.g. sigma with sort σ , capital letters for concrete calcule e.g. ALPHA with sort A .

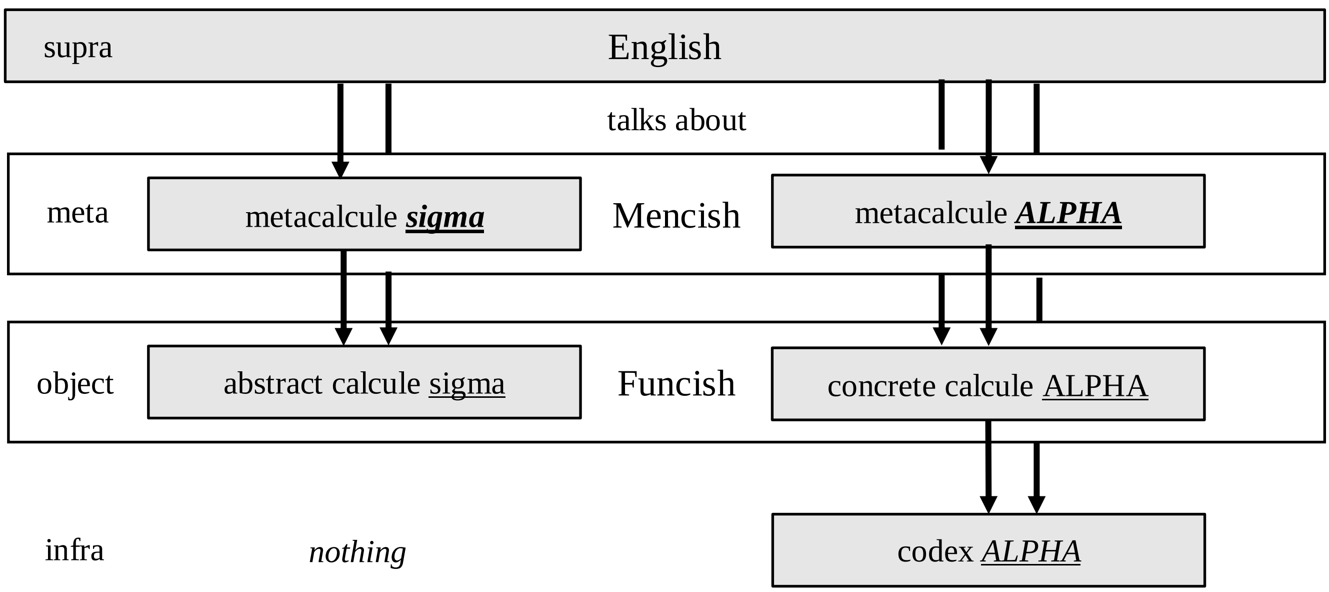

Figure 1 Hierarchy of languages and codices for two example calcules

Metalanguage **Mencish** is chosen with **perfect exactness**, just as object-language Funcish. They both have to meet the **calculation criterion of truth**: every step of reasoning must be such that it can be checked by a calculating machine. Funcish and Mencish sentences and metasentences resp.are understandable without context: 'wherefore by their *words* ye shall know them' (vs. Mathew 7.20).

On first sight Funcish and Mencish look familiar to what one knows from predicate-logic. However, they are especially adapted to a degree of precision so that they can be used universally for all kind of mathematics. And they lend themselves immediately to a treatment by computers, as they have perfect syntax and semantics. It is not the place to go into details. Both Funcish and Mencish have essentially the same syntax. Notice that Funcish has a context-independent notation, which implies that one can determine the **category** of every object uniquely from its syntax. The reader may be puzzled by some expressions that are either newly coined by the author or used slightly different from convention. This is done in good faith; the reason for the so-called **Bavarian notation** is to avoid ambiguities.

The **fonts-method** allows to distinguish between object-language (Arial and Symbol, normal, e.g. $\forall\sigma_1[\ ]$), metalanguage (Arial and Symbol, boldface italics e.g. $\boldsymbol{\sigma_1}$ or ***Axiom***) and supralanguage **English** (Times New Roman)

The essential parts of a language are its **sentences**. A sentence is a **string** of **characters** of a given **alphabet** that fulfills certain syntactical and semantical rules. This means that metalanguage talks about the strings of the object-language. The essential parts of the metalanguage are the metasentences (that are strings of characters as well, just in boldface italics). In supralanguage one talks about the metasentences, just as metalanguage talks about object-language. Here it is not discussed in general what an object-language talks **about** .

Mencish has strictly first order logics, it has some recursion with 3 metafunctions and 6 binary metarelations for the manipulation of Funcish strings, that can be implemented by machines operating on strings of characters. The necessary metafunctions and metarelations:

synaption	$\boldsymbol{(\sigma*\sigma)}$	concatenation of two strings, $\boldsymbol{\sigma\ \sigma}$ except for leading 0
character-deletion	$\boldsymbol{(\sigma\partial\sigma)}$	delete 2nd string in 1st string
string-replacement	$\boldsymbol{(\sigma;\sigma/\sigma)}$	replece 2nd string in 1st string by 3rd string, e.g. variable by constant
matching	$\boldsymbol{\sigma\approx\sigma}$	with respect to string length
shorter	$\boldsymbol{\sigma\prec\sigma}$	with respect to string length
suitably containing	$\boldsymbol{\sigma\supset\sigma}$	e.g. variable
suitably bound-in	$\boldsymbol{\sigma/\sigma}$	variable
suitably free-in	$\boldsymbol{\sigma/\sigma}$	variable
compatible	$\boldsymbol{\sigma\sim\sigma}$	no variable is free in one and bounded in the other string.

A calcule contains ***basis-individual-constant*** , ***basis-relation-constant*** and ***basis-function-constant*** strings, one can introduce ***extra-individual-constant*** , ***extra-relation-constant*** and ***extra-function-constant*** strings by definitions, that are either explicit or implicit. An explicit definition is just an abbreviation whereas an implicit definition necessitates some logical reasoning. Without proper introduction it should be clear what is meant by the following essential metaproperties :

pattern	***term***	no ***variable***	e.g. $(3+A_u)$
	scheme	with ***variable***	e.g. $((A_1+A_1)\times A_3)$
phrase	***sentence***	no free ***variable***	
	formula	with free ***variable*** but no A_0	e.g. $\exists A_3[A_1=(A_3\times A_3)]$
	formulo	with free ***variable*** and free A_0	e.g. $[A_2=(A_0+A_1)]\vee[A_1=(A_0+A_2)]$

Implicit definitions are based on ***UNEX-formulo*** [1)] strings. ***UNEX-formulo*** strings define relations that hold for exactly one value σ_0 for every booking of the input ***variable*** strings according to the arity of the ***UNEX-formulo*** . Hopefully it will become clearer in the examples of section 3.

TRUTH mater of implicit definition of individuals (nullary functions) by nullary ***UNEX-formulo***

∀σ1[∀σ2[[[[**formulo(σ1)**] ∧ [**variable(σ2)**]] ∧ [¬ [σ1 ⊃ σ2)]]] ∧ [**sentence(** ∀σ0[σ1])]] →
[∃σ3[[**individual-constant(** σσ3)] ∧
[**TRUTH(** [∃σ0[[σ1]∧[∀σ2[[(σ1; σ0∫ σ2)]→[σ2=σ0]]]]]→[∀σ0[[σ1]↔[σσ3=σ0]]])]]]]]

TRUTH mater of implicit definition of functions by uanry and multary ***UNEX-formulo***

∀σ1[∀σ2[∀σ3[∀σ4[∀σ5[[[[[[[**formulo(σ1)**] ∧ [**omny(σ2)**]] ∧ [**variable(σ4)**]] ∧
[¬ [σ1 ⊃ σ4)]]] ∧ [**sentence(** σ2 ∀σ0[σ1]σ3]] ∧ [σ5 = (((σ2; ∀[∫;) ∂∀) ∂[ )]] →
[∃σ6[[**standard-function-constant(** σσ6((σ5 ∂)))] ∧ [**TRUTH(**
[σ2 ∃σ0[[σ1]∧[∀σ4[[(σ1; σ0∫ σ4)]→[σ4=σ0]]]]]σ3]→[σ2∀σ0[[σ1]↔[σσ6(σ5)=σ0]]] σ3])]]]]]]]]

where the ***standard-function-constant*** string can be replace by a ***particular-function-constant*** string if it is considered convenient.

You will find some hints on the front of the hompage https://pai.de/ and some short description in the pdf-downloads that can be started there. A thorough description of Funcish and Mencish is forthcoming.

1) ***UNEX*** is short for 'unique existence'

2 Ontological basis and axioms

The obvious question is: how does axiomatic set theory relate to the Funcish-Mencish language system? The answer is straightforward: one can set up the abstract calcule <u>sigma</u> with FOL which exactly describes ZFC-set-theory, Zermelo-Fraenkel-theory with the axiom of selection (or choice).

It is clear that one cannot represent ZFC-set-theory by a concrete calcule (that only would allow a finite string for an ***individual***). Therefore an abstract calcule is introduced in the following. It will be discussed what it means that FOL is sufficient for axiomatic set-theory and what kind of problems are connected with it. So-called 'classes' as 'collections of sets that can be unambiguously defined by a property that all its members share' are not formally contained in ZFC are not treated in this section, although it could be done straightforwardly. Neither does ZFC-set-theory contain any so-called 'urelements', i.e. elements of sets that are not sets themselves.

All strings of Funcish for calcule <u>sigma</u> are constructed by concatenation from the following alphabet, of 128 characters where the first line contains all syntax characters including the Greek letter σ for sort 'set' , the second and third all function symbols followed by all relation symbols, the fourth all Latin letters for constants (i.e.names) and finally the petit numbers for variable suffices.

=	≠	¬	∨	∧	→	↔	∃	∀	[	]	(	)	;	σ												*Symbol*
′	°	◊	♦	√	!	⇑	⇓	+	−	×	/	↑	↓	⊕	©	⊗	®	•	÷	∤	∩	∪	∗	∂	∫	*12*
∇	□	℘	&			#	⊥	|	∤	<	≤	{	≈	~	)	(	⊃	≅	∠	⊂	⊆	∈	—	∴	…	
A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	D	T	U	V	W	X	Y	Z	*Arial*
a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	*10 medium*
0	1	2	3	4	5	6	7	8	9	-																*8 petit*

Table 1 Alphabet for calcule <u>sigma</u> (with enough characters for new defintions)

The admissible ***pattern*** (***scheme*** , ***term***) and ***phrase*** (***formula*** , ***sentence***) strings are formed with respect to the ontological basis according to the semantic rules for Funcish. The following examples should be sufficient in order to understand this publication (all of it can be defined perfectly, at most simple limitive recursion is necessary):

sort ::	σ	*set*
type ::	(σ) ¦ (σ;σ) ¦ σ(σ) ¦ σ(σ;σ) …	*property, binary relation, unary and binary function*
individual-constant ::	σn ¦ σvnl ¦ σvnpo …	*empty set, set of natural numbers, their power set ..*
individual-variable ::	σ_0 ¦ σ_1 ¦ σ_2 …	
omny :: kety ::	∀σ_1[¦ ∀σ_1[∀σ_2[… …]] ¦]	
function-constant ::	σ|(σ;σ) ¦ σ∪(σ) ¦ σ⇑(σ) …	*parition, unition, potention ... in standard notation* [1)]
	(σ|σ) ¦ (∪σ) ¦ (⇑σ) …	*and in corresponding particular notation*
relation-constant ::	∈(σ;σ) ¦⊂(σ;σ) ¦ ~(σ;σ) …	*membrity, subity, card-equality ... in standard notat.*
	σ∈σ ¦ σ⊂σ ¦ σ~σ …	*and in corresponding particular notation*
term ::	σn ¦(|(|(|σn))) ¦ (⇑σvnl)|σn) …	*from* ***function-*** *and* ***individual-constant***
scheme ::	σ_2 ¦(σ_1|σ_4) ¦ (⇑σvnl|(∪σ_2)) …	*from* ***function-*** *and* ***individual-constant*** *and at least one* ***individual-variable*** *(all in particular notation)*

formula strings are formed by proper insertions with = , ≠ , ¬ , ∨ , ∧ , → , ↔ , [,] , ∃ , ∀ from ***term*** , ***scheme*** and ***variable*** strings except for σ_0 . ***formula*** strings have at least one free ***variable*** . ***formulo*** strings are like ***formula*** strings but they must contain σ_0 as a free ***variable***.

sentence strings are formed like ***formula*** strings, however all ***variable*** strings must be bound by ∀ or ∃ . They may contain σ_0 as a bound ***variable*** .

[1)] particular notation is used instead of standard notation for better readability for most of the frequently occuring relations and functions

The ***basis-ingredient*** strings of abstract calcule <u>sigma</u> of axiomatic set theory (usually called ZFC) contain only one ***sort*** σ for sets and only one ***binary-relation-constant*** σ∈σ for **membrity**. The ***extra-individual-constant*** , ***extra-relation-constant*** and ***extra-function-constant*** strings are given in section 3..

nr	***-axiom***	*string or mater*
A1	***Extensionality*** [1]	∀σ1[∀σ2[[σ1=σ2]↔[∀σ3[[σ3∈σ1]↔[σ3∈σ2]]]]]
A2	***Regularity*** [2]	∀σ1[[∃σ2[σ2∈σ1]]→[∃σ2[[σ2∈σ1]∧[¬[∃σ3[[σ3∈σ1]∧[σ3∈σ2]]]]]]]
A3	***Selectivity*** [3]	∀σ1[[[∀σ2[[σ2∈σ1]→[∃σ3[σ3∈σ2]]]]∧ [∀σ2[∀σ3[∀σ4[[[[σ2∈σ1]∧[σ3∈σ1]]∧[σ4∈σ2]]∧[σ4∈σ3]]]]]]→ [∃σ2[∀σ3[[σ3∈σ1]→[∃σ4[[[σ4∈σ2]∧[σ4∈σ3]]∧ [∀σ5[[[σ5∈σ2]∧[σ5∈σ3]]→[σ5=σ4]]]]]]]]]
A4	***Nilition*** [4]	∃σ0[∀σ1[¬[σ1∈σ0]]]
A5	***Parition*** [5]	∀σ1[∀σ2[∃σ0[∀σ3[[σ3∈σ0]↔[[σ3=σ1]∨[σ3=σ2]]]]]]
A6	***Unition*** [6]	∀σ1[∃σ0[∀σ2[[σ2∈σ0]↔[∃σ3[[σ3∈σ1]∧[σ2∈σ3]]]]]]
A7	***Potention*** [7]	∀σ1[∃σ0[∀σ2[[σ2∈σ0]↔[∀σ3[[σ3∈σ2]→[σ3∈σ1]]]]]]
A8	***Recursion*** [8] σ0 from ***σ1*** , ***σ2*** nullary and unary ***UNEX-formulo*** strings iteration from start and no subset with same feature	***∀σ1[∀σ2[∀σ3[∀σ4[∀σ5[[[[[[[*** ***[sentence(*** ∀σ0[***σ1***] ***)]]∧ [sentence(*** ∀σ0[∀σ1[***σ2***]] ***)]]∧*** ***[variable(σ3)]]∧ [variable(σ4)]]∧ [variable(σ5)]]∧*** ***[sentence(*** ∀σ0[∀***σ3***[∀***σ4***[∀***σ5***[***σ1***]]]] ***)]]∧*** ***[sentence(*** ∀σ0[∀σ1[∀***σ3***[∀***σ4***[∀***σ5***[***σ2***]]]]] ***)]] →*** ***[Axiom(*** [[∃σ0[[***σ1***]∧[∀***σ3***[[***(σ1;*** σ0∫ ***σ3)***]→[***σ3***=σ0]]]]]∧ [∀σ1[∃σ0[[***σ2***]∧[∀***σ3***[[***(σ2;*** σ0∫ ***σ3)***]→[***σ3***=σ0]]]]]]]→[∃σ0[[∀***σ3***[[[[***σ3***∈σ0]∧[***(σ1;*** σ0∫ ***σ3)***]]∨[∃σ1[[σ1∈σ0]∧[***(σ2;*** σ0∫ ***σ3)***]]]]∧ [∀***σ4***[[[∀***σ5***[[***σ5***∈***σ4***]→[***σ5***∈σ0]]]∧[∀***σ3***[[[***σ3***∈***σ4***]∧[***(σ1;*** σ0∫ ***σ3)***]]∨ [∃σ1[[σ1∈***σ4***]∧[***(σ2;*** σ0∫ ***σ3)***]]]]]]→[***σ4***=σ0]]]]] ***)]]]]]]***
A9	***Separation*** [9] intersection σ0 by ***scheme σ1*** and relation given by ***formula σ2***	***∀σ1[∀σ2[∀σ3[∀σ4[∀σ5[[[[[*** ***[scheme(σ1)] ∧ [formula(σ2)]] ∧ [variable(σ3)]] ∧ [omny(σ4)]] ∧*** ***[sentence(σ4*** [***σ1*** = ***σ1***]∧[∀σ0[∀***σ3*** [***σ2***]]***σ5)]] →*** ***[Axiom(σ4*** ∃σ0[∀***σ3***[[***σ3***∈σ0]↔[[***σ3***∈***σ1***]∧[***σ2***]]]] ***σ5)]]]]]]***
A10u	***Unary-replacion*** [10]	***∀σ1[[[sentence(*** ∀σ1[∀σ0[***σ1***]]***)] ∧*** ***[sentence(*** ∀σ0[∀σ1[∀σ2[∀σ3[***σ1***]]]]***)]] →*** ***[Axiom(*** [∀σ1[∃σ0[[***σ1***]∧[∀σ2[[***(σ1;*** σ0∫σ2***)***]→[σ2=σ0]]]]]]→ [∀σ2[∃σ3[∀σ0[[σ0∈σ3]↔[∃σ1[[σ1∈σ2]∧[***σ1***]]]]]]] ***)]]***
A10m	***Multary-replacion*** image σ3 of a set σ2 under ***σ1*** ***UNEX-formulo***	***∀σ1[∀σ2[∀σ3[[[[[omny(σ2)] ∧ [sentence(σ2*** ∀σ1[∀σ0[***σ1***]] ***σ3)]] ∧*** ***[formulo(σ1)]] ∧ [sentence(σ2*** ∀σ0[∀σ1[∀σ2[∀σ3[***σ1***]]]]***σ3)]] →*** ***[Axiom(σ2*** [∀σ1[∃σ0[[***σ1***]∧[∀σ2[[***(σ1;*** σ0∫σ2***)***]→[σ2=σ0]]]]]]→ [∀σ2[∃σ3[∀σ0[[σ0∈σ3]↔[∃σ1[[σ1∈σ2]∧[***σ1***]]]]]]]***σ3)]]]]***

Table 2 ***Axiom*** *strings and matres*

The order is chosen so that the three metaproperty axioms are followed by four ***UNEX-formulo*** strings, followed by four ***Axiom*** matres, where the last ones treat unary and multary cases separately.

If there are redundancies in the above ***Axiom*** table, so what: just replace them by the simpler version.

[1] sometimes called 'extension' [2] often called 'foundation' [3] usually called 'selection' or 'choice', sometimes replaced by the weaker 'well-ordering' [4] nullary - usually 'empty set' [5] binary - usually called 'pairing' [6] unary - usually one of the two meanings of 'union' [7] unary - usually called 'power set' [8] generalizes what is usually called 'infinity' [9] often called 'specification' [10] usually called 'replacement'

There are 7 **constructions** of sets, 4 by ***Axiom*** strings (***A4*** to ***A7*** for nilition, parition, unition and potention) and 3 by ***Axiom*** matres (***A8*** to ***A10*** for recursion, replacition and separation) with uniqueness by ***A1*** .

Recursion-axiom mater ***A8*** is taken more general than usual. Rather than starting from the empty set and natural number succession it is generalized in the following sense: start from any set given by a ***UNEX-nullary-formulo*** σ_1 iterate by a unary function that is given by a ***UNEX-unary-formulo*** σ_2 . Perhaps one should generalize it even more: higher arities for σ_2 ? The usual 'Infinity' ***Axiom*** is just one special case that relates to recursion where function succession $('\sigma)$ of section 3 is used that produces von-Neumann natural numbers (conventionally denoted as ω) starting from the empty set.

$\exists\sigma_1[[\exists\sigma_2[\forall\sigma_1[[\sigma_2\in\sigma_1]\wedge[\forall\sigma_3[\neg[\sigma_3\in\sigma_1]]]]]]\wedge[\forall\sigma_2[[\sigma_2\in\sigma_1]\rightarrow[\exists\sigma_3[[\sigma_3\in\sigma_1]\wedge[\forall\sigma_4[[\sigma_4\in\sigma_3]\leftrightarrow$
$[[\sigma_4\in\sigma_2]\vee[\sigma_4=\sigma_2]]]]]]]]]]$

or with the definitions for empty set σn (conventionally denoted as $\varnothing$) and succession $('\sigma)$
$\exists\sigma_1[[\sigma n\in\sigma_1]\wedge[\forall\sigma_2[[\sigma_2\in\sigma_1]\rightarrow[('\sigma_2)\in\sigma_1]]]]$

Separation-axiom mater ***A9*** allow for a more general choice for σ_1 (a ***scheme*** σ_1 depending on ***variable*** σ_1 and more ***variable*** strings) and of ***formula*** σ_2 that depends on ***variable*** σ_1 as well. It makes definition of functions simpler, if it is not at all necessary.

The possibilities of mater ***A9*** include the simple reduced case that is kind of a prototype in usual introductions of axiomatic set theory. It states the existence of a set σ_0 given by a unary ***formula*** σ_2 (with free ***variable*** σ_3 and no ***variable*** σ_0 and σ_1) with respect to a given set σ_1 :

Axiom($\forall\sigma_1[\exists\sigma_0[\forall\sigma_3[[\sigma_3\in\sigma_0]\leftrightarrow[[\sigma_3\in\sigma_1]\wedge[\ \sigma_2\ ]]]]]$ ***)***

Replacition-axiom matres ***A10*** express that 'the image of a set σ_2 under a function given by a ***UNEX-formulo*** σ_1 is a set σ_3 . One has to distinguish the unary and the multary case ***A10u*** and ***A10m*** .The functions are represented by ***UNEX-formulo*** strings. As finite strings ***UNEX-formulo*** are denumerable but not enumerable (enumerable meaning effectively denumerable); unique existence has to be shown in every case. How does that relate to non-denumerable sets? After all axiomatic set theory was introduced to allow for a gargantuan universum of infinities far beyond the denumerable! But that is another problem that will not be treated here.

It is convenient to introduce ***extra-function-constant*** , ***extra-individual-constant*** and ***extra-relation-constant*** strings, some of them are well-known from naive set theory. This is done in the following three sections.

3 *Introduction of* ***extra-function-constant*** *strings*

It is convenient to introduce ***extra-function-constant*** strings.

nr	*name*		*definition*	*ari*
			implicit from ***Axiom A5*** *,* ***A6*** *,* ***A7***	
DF1	parition	$(\sigma\vert\sigma)$	$\forall\sigma_1[\forall\sigma_2[\forall\sigma_3[[\sigma_3\in(\sigma_1\vert\sigma_2)]\leftrightarrow[[\sigma_3=\sigma_1]\vee[\sigma_3=\sigma_2]]]]]$	*2*
DF2	unition	$(\cup\sigma)$	$\forall\sigma_1[\forall\sigma_2[[\sigma_2\in(\cup\sigma_1)]\leftrightarrow[\exists\sigma_3[[\sigma_3\in\sigma_1]\wedge[\sigma_2\in\sigma_3]]]]]$	*1*
DF3	potention	$(\Uparrow\sigma)$	$\forall\sigma_1[\forall\sigma_2[[\sigma_2\in(\Uparrow\sigma_1)]\leftrightarrow[\forall\sigma_3[[\sigma_3\in\sigma_2]\rightarrow[\sigma_3\in\sigma_1]]]]]$	*1*
			explicit	
DF4	singlition [1]	$(\vert\sigma)$	$(\sigma_1\vert\sigma_1)$	*1*
DF5	singlation	$(\vert\vert\sigma)$	$(\sigma_1\vert(\vert\sigma_1))$	*1*
DF6	oparition [2]	$(\sigma-\sigma)$	$((\vert\sigma_1)\vert(\sigma_1\vert\sigma_2))$	*2*
DF7	union [3]	$(\sigma\cup\sigma)$	$(\cup(\sigma_1\vert\sigma_2))$	*2*
DF8	succession	$('\sigma)$	$(\cup(\vert\vert\sigma_1))=(\cup(\sigma_1\vert(\vert\sigma_1)))=(\sigma_1\cup(\vert\sigma_1))$	*1*
			implicit from unary case of ***Separation-axiom*** *mater*	
DF9	intersectition	$(\cap\sigma)$	$\forall\sigma_1[\forall\sigma_2[[\sigma_2\in(\cap\sigma_1)]\leftrightarrow[\forall\sigma_3[[\sigma_3\in\sigma_1]\rightarrow[\sigma_2\in\sigma_3]]]$	*1*
DF10	imponition	$(\Diamond\sigma)$	$\forall\sigma_1[\forall\sigma_2[[\sigma_2\in(\Diamond\sigma_1)]\leftrightarrow[\exists\sigma_3[[\sigma_3\in\sigma_1]\wedge[\sigma_2=(\vert\sigma_3)]]]]]$	*1*
			implicit from binary case of ***Separation-axiom*** *mater*	
DF11	intersection	$(\sigma\cap\sigma)$	$\forall\sigma_1[\forall\sigma_2[\forall\sigma_3[[\sigma_3\in(\sigma_1\cap\sigma_2)]\leftrightarrow[[\sigma_3\in\sigma_1]\wedge[\sigma_3\in\sigma_2]]]]]$	*2*
DF12	complemention	(σ/σ)	$\forall\sigma_1[\forall\sigma_2[\forall\sigma_3[[\sigma_3\in(\sigma_1/\sigma_2)]\leftrightarrow[[\sigma_3\in\sigma_1]\wedge[\neg[\sigma_3\in\sigma_2]]]]]]$	*2*
DF13	production	$(\sigma\times\sigma)$	$\forall\sigma_1[\forall\sigma_2[\forall\sigma_3[[\sigma_3\in(\sigma_1\times\sigma_2)]\leftrightarrow$ $[\exists\sigma_4[\exists\sigma_5[[[\sigma_4\in\sigma_1]\wedge[\sigma_5\in\sigma_2]]\wedge[\sigma_3=(\sigma_4-\sigma_5)]]]]]]]$	*2*
			explicit from implicit above	
DF14	tri-production	$(\sigma\times\sigma\times\sigma)$	$(\sigma_1\times(\sigma_2\times\sigma_3))$	*3*
DF15	bi-potentiation	$(\sigma\times)$	$(\sigma_1\times\sigma_1)$	*1*
DF16	tri-potentiation	$(\sigma\times\times)$	$(\sigma_1\times(\sigma_1\times\sigma_1))$	*1*
			implicit from unary case of ***Separation-axiom*** *mater*	
DF18	diag-production	$(\times\sigma)$	$\forall\sigma_1[\forall\sigma_2[[\sigma_2\in(\times\sigma_1)]\leftrightarrow[[\sigma_2\in(\sigma_1\times)]\wedge[\exists\sigma_3[\sigma_2=(\sigma_3-\sigma_3)]]]]]$	*1*

Table 3 Definition of ***extra-function-constant*** *strings*

The ***Extensionality-axiom A1*** is necessary for uniqueness of functions in the following.

Definition in full detail for binary function **parition** $(\sigma|\sigma)$ as there is ***A5***
$\forall\sigma_1[\forall\sigma_2[\exists\sigma_0[\forall\sigma_3[[\sigma_3\in\sigma_0]\leftrightarrow[[\sigma_3=\sigma_1]\vee[\sigma_3=\sigma_2]]]]]]$ with ***formulo*** $\forall\sigma_3[[\sigma_3\in\sigma_0]\leftrightarrow[[\sigma_3=\sigma_1]\vee[\sigma_3=\sigma_2]]]$
then there is unique existence by ***Axiom*** mater of implicit definition of multary functions:
$\exists\sigma_1(\sigma;\sigma)[\forall\sigma_1[\forall\sigma_2[\forall\sigma_3[[\sigma_3\in\sigma_0]\leftrightarrow[[\sigma_3=\sigma_1]\vee[\sigma_3=\sigma_2]]]]]]$
and one can introduce $(\sigma|\sigma)$ for $\sigma_1(\sigma;\sigma)$ and define it by
$\forall\sigma_1[\forall\sigma_2[\forall\sigma_3[[\sigma_3\in(\sigma_1|\sigma_2)]\leftrightarrow[[\sigma_3=\sigma_1]\vee[\sigma_3=\sigma_2]]]]]$

Definition in full detail for unary function **unition**$(\cup\sigma)$ as there is ***A6***
$\forall\sigma_1[\exists\sigma_0[\forall\sigma_2[[\sigma_2\in\sigma_0]\leftrightarrow[\exists\sigma_3[[\sigma_3\in\sigma_1]\wedge[\sigma_2\in\sigma_3]]]]]]$
then there is unique existence by ***Axiom*** mater of implicit definition of unary functions:
$\exists\sigma_1(\sigma)[\forall\sigma_1[\forall\sigma_2[[\sigma_2\in\sigma_1(\sigma_1)]\leftrightarrow[\exists\sigma_3[[\sigma_3\in\sigma_1]\wedge[\sigma_2\in\sigma_3]]]]]]$
and one can introduce $(\cup\sigma)$ for $\sigma_1(\sigma)$ and define it by
$\forall\sigma_1[\forall\sigma_2[[\sigma_2\in(\cup\sigma_1)]\leftrightarrow[\exists\sigma_3[[\sigma_3\in\sigma_1]\wedge[\sigma_2\in\sigma_3]]]]]$

[1] a singleton is a set with one element [2] the result is usually called 'ordered pair set' [3] pair-union, usually there are two meanings for 'union'

Unary function **potention** ($\Uparrow\sigma$) is also obtained by implicit definition with the ***UNEX-formulo*** from ***A7*** .

The following are definitions from unary case of ***Separation-axiom*** mater
$\forall\sigma_1[\exists\sigma_0[\forall\sigma_2[[\sigma_2\in\sigma_0]\leftrightarrow[[\sigma_2\in\boldsymbol{\sigma_1}]\wedge[\boldsymbol{\sigma_2}]]]]]$ with ***sentence(*** $\forall\sigma_1[[\boldsymbol{\sigma_1}=\boldsymbol{\sigma_1}]\wedge[\forall\sigma_2[\boldsymbol{\sigma_2}]]]$ ***)***

for **intersectition** ($\cap\sigma$) choose ($\cup\sigma_1$) as $\boldsymbol{\sigma_1}$ and $\forall\sigma_3[[\sigma_3\in\sigma_1]\rightarrow[\sigma_2\in\sigma_3]]$ as binary $\boldsymbol{\sigma_2}$
$\forall\sigma_1[\exists\sigma_0[\forall\sigma_2[[\sigma_2\in\sigma_0]\leftrightarrow[[\sigma_2\in(\cup\sigma_1)]\wedge[\forall\sigma_3[[\sigma_3\in\sigma_1]\rightarrow[\sigma_2\in\sigma_3]]]]]]]\quad]$
$\forall\sigma_1[\exists\sigma_0[\forall\sigma_2[[\sigma_2\in\sigma_0]\leftrightarrow[[\exists\sigma_3[[\sigma_3\in\sigma_1]\wedge[\sigma_2\in\sigma_3]]]]\wedge[\forall\sigma_3[[\sigma_3\in\sigma_1]\rightarrow[\sigma_2\in\sigma_3]]]]]]]$
$\forall\sigma_1[\exists\sigma_0[\forall\sigma_2[[\sigma_2\in\sigma_0]\leftrightarrow[\forall\sigma_3[[\sigma_3\in\sigma_1]\rightarrow[\sigma_2\in\sigma_3]]]]]]$

for **imponition** ($\Diamond\sigma$) choose ($\cup\sigma_1$) as $\boldsymbol{\sigma_1}$ and $\forall\sigma_3[[\sigma_3\in\sigma_1]\rightarrow[\sigma_2=(|\sigma_3)]]$ as binary $\boldsymbol{\sigma_2}$

$\forall\sigma_1[\exists\sigma_0[\forall\sigma_2[[\sigma_2\in(\Diamond\sigma_1)]\leftrightarrow[\exists\sigma_3[[\sigma_3\in\sigma_1]\wedge[\sigma_2=(|\sigma_3)]]]]]]$

The following are definitions from binary case of ***Separation-axiom*** mater
$\forall\sigma_1[\forall\sigma_2[\exists\sigma_0[\forall\sigma_3[[\sigma_3\in\sigma_0]\leftrightarrow[[\sigma_3\in\boldsymbol{\sigma_1}]\wedge[\boldsymbol{\sigma_2}]]]]]]$ with ***sentence(*** $\forall\sigma_1[\forall\sigma_2[[\boldsymbol{\sigma_1}=\boldsymbol{\sigma_1}]\wedge[\forall\sigma_3[\boldsymbol{\sigma_2}]]]$ ***)***

for **intersection** ($\sigma\cap\sigma$) choose σ_1 as $\boldsymbol{\sigma_1}$ and $\sigma_3\in\sigma_2$ as ternary $\boldsymbol{\sigma_2}$

for **complemention** (σ/σ) choose σ_1 as $\boldsymbol{\sigma_1}$ and $\neg[\sigma_3\in\sigma_2]$ as ternary $\boldsymbol{\sigma_2}$

for **production**($\sigma\times\sigma$) choose ($\Uparrow(\Uparrow(\sigma_1\cup\sigma_2))$) as $\boldsymbol{\sigma_1}$ and $\exists\sigma_4[\exists\sigma_5[[[\sigma_4\in\sigma_1]\wedge[\sigma_5\in\sigma_2]]\wedge[\sigma_3=(\sigma_4-\sigma_5)]]]$ as ternary $\boldsymbol{\sigma_2}$ and obtain

$\forall\sigma_1[\forall\sigma_2[\forall\sigma_3[[\sigma_3\in(\sigma_1\times\sigma_2)]\leftrightarrow[\exists\sigma_4[\exists\sigma_5[[[\sigma_4\in\sigma_1]\wedge[\sigma_5\in\sigma_2]]\wedge[\sigma_3=(\sigma_4-\sigma_5)]]]]]]]$

It looks like using a sledgehammer to crack a nut, as the set ($\Uparrow(\Uparrow(\sigma_1\cup\sigma_2))$) is doubly infinite and one is using only the simplest subsets consisting of one or two members. But one cannot do it otherwise. The set ($\Uparrow(\Uparrow(\sigma_1\cup\sigma_2))$) contains all ordered pairs of members of the two sets σ_1 and σ_2 (and much more, but this is irrelevant as only a subset is aimed for).

4 *Introduction of* ***extra-individual-constant*** *strings*

The introduction of ***extra-individual-constant*** strings makes use of the above functions and contains implicit definitions, for the latter one has ***UNEX-nullary-formulo*** strings.

nr	*name*		*definition*	*ari*
			implicit by **Nilition-axiom A4**	
DI0	***nil***	σn	$\forall\sigma_1[\neg[\sigma_1\in\sigma n]]$	*0*
			explicit from functions	
DI1	***neumann-unus***	σvu	$({}'\sigma n)$	*0*
DI2	***cantor-duo***	σcb	$(\vert(\vert\sigma n))$	*0*
DI3	***zermelo-quattuor***	σzq	$(\Vert(\Vert(\Vert(\Vert\sigma n))))$	*0*
DI4	***fraenkel-tres***	σft	$(\Uparrow(\Uparrow(\Uparrow\sigma n)))$	*0*
	…		*the other ones accordingly*	
			implicit by **Recursion-axiom** *start* σn *and*	
DI11	***neumann-natral***	σvnl	*iteration* $({}'\sigma)$	*0*
DI12	***cantor-natral***	σcnl	*iteration* $(\vert\sigma_2)$	*0*
DI13	***zermelo-natral***	σznl	*iteration* $(\Vert\sigma_2)$	*0*
DI14	***fraenkel-natral***	σfnl	*iteration* $(\Uparrow\sigma_2)$	*0*
			explicit therefrom	
DI15	***neumann-potential***	$\sigma vnpo$	$(\Uparrow\sigma vnl)$	*0*
DI16	***cantor-potential***	$\sigma cnpo$	$(\Uparrow\sigma cnl)$	*0*

Table 4 Definition of ***extra-individual-constant*** *strings*

Definition in full detail: set ***neumann-natral*** σvnl of natural numbers, it is based on two ***UNEX-formulo*** strings for start $\boldsymbol{\sigma_1 =}$ $\sigma_0=\sigma n$ and succession $\boldsymbol{\sigma_2 =}$ $\sigma_0=({}'\sigma_1)$ and from **Recursion-axiom** mater there is unique existence of its 'power-set' obtained by potention $(\Uparrow\sigma cnl)$; *in conventional notation the von-Neumann-set of natural numbers is called* ω : {{},{{},{{}}},{{{},{{}}},{{{},{{}}}}} , … } *with power-set* $\mathbf{P}(\omega)$.

$$\exists\sigma_0[[[\sigma n\in\sigma_0]\wedge[\forall\sigma_2[[\sigma_2=\sigma n]\vee[\exists\sigma_1[[\sigma_1\in\sigma_0]\wedge[\sigma_2=({}'\sigma_1)]]]]]]\wedge$$
$$[\forall\sigma_3[[[\forall\sigma_4[[\sigma_4\in\sigma_3]\rightarrow[\sigma_4\in\sigma_0]]]\wedge$$
$$[[\sigma n\in\sigma_3]\wedge[\forall\sigma_2[[\sigma_2=\sigma n]\vee[\exists\sigma_1[[\sigma_1\in\sigma_3]\wedge[\sigma_2=({}'\sigma_1)]]]]]]]\rightarrow[\sigma_3=\sigma_0]]]]$$

and by application of implicit definition (nullary case) one can introduce σvnl for σ_0 and define it by

$$[[\sigma n\in\sigma vnl]\wedge[\forall\sigma_2[[\sigma_2=\sigma n]\vee[\exists\sigma_1[[\sigma_1\in\sigma vnl]\wedge[\sigma_2=({}'\sigma_1)]]]]]]\wedge[\forall\sigma_3[$$
$$[[\forall\sigma_4[[\sigma_4\in\sigma_3]\rightarrow[\sigma_4\in\sigma_0]]]\wedge[[\sigma n\in\sigma_3]\wedge[\forall\sigma_2[[\sigma_2=\sigma n]\vee[\exists\sigma_1[[\sigma_1\in\sigma_3]\wedge[\sigma_2=({}'\sigma_1)]]]]]]]\rightarrow[\sigma_3=\sigma vnl]]]$$

In the same way one can introduce ***cantor-natral*** σcnl of natural numbers and its 'power-set' obtained by potention $(\Uparrow\sigma cnl)$; *in conventional notation the Cantor-set is given as* {{},{{}}, {{{}}}, … } .
It is based on two ***UNEX-formulo*** strings for start $\boldsymbol{\sigma_1 =}$ $\sigma_0=\sigma n$ and singlition $\boldsymbol{\sigma_2 =}$ $\sigma_0=(\vert\sigma_1)$

$$[[\sigma n\in\sigma cnl]\wedge[\forall\sigma_2[[\sigma_2=\sigma n]\vee[\exists\sigma_1[[\sigma_1\in\sigma cnl]\wedge[\sigma_2=({}'\sigma_1)]]]]]]\wedge[\forall\sigma_3[$$
$$[[\forall\sigma_4[[\sigma_4\in\sigma_3]\rightarrow[\sigma_4\in\sigma_0]]]\wedge[[\sigma n\in\sigma_3]\wedge[\forall\sigma_2[[\sigma_2=\sigma n]\vee[\exists\sigma_1[[\sigma_1\in\sigma_3]\wedge[\sigma_2=(\vert\sigma_1)]]]]]]]\rightarrow[\sigma_3=\sigma cnl]]]$$

It is a matter of taste and convenience, what set one chooses to represent natural numbers. Usually it is von-Neumann's choice, but in the following simple considerations the simpler Cantor-set σcnl is sufficient.

5 *Introduction of* ***extra-relation-constant*** *strings*

The definition of a ***relation-constant*** by a ***formula*** is straightforward. Depending on the arity of the ***formula*** one should enclose defining ***formula*** strings by $\forall\sigma_1[[\ldots]\leftrightarrow[\ \ldots\ ]]$ or $\forall\sigma_1[\forall\sigma_2[[\ldots]\leftrightarrow[\ \ldots\ ]]]$ or $\forall\sigma_1[\forall\sigma_2[\forall\sigma_3[[\ldots]\leftrightarrow[\ \ldots\ ]]]]$ or $\forall\sigma_1[\forall\sigma_2[\forall\sigma_3[\forall\sigma_4[[\ldots]\leftrightarrow[\ \ldots\ ]]]]]$ resp.

nr	*name*		*definition by* ***formula***	*ar*
DR1	equi-subity	$\sigma\subseteq\sigma$	$\forall\sigma_3[[\sigma_3\in\sigma_2]\rightarrow[\sigma_3\in\sigma_1]]$	*2*
DR2	subity (genuine subset)	$\sigma\subset\sigma$	$[\sigma_2\subseteq\sigma_1]\wedge[\sigma_2\neq\sigma_1]$	*2*
DR3	oparity	$OP(\sigma)$	$\forall\sigma_2[[\sigma_2\in\sigma_1]]\rightarrow[\exists\sigma_3[\exists\sigma_4[\sigma_2=(\sigma_3-\sigma_4)]]]$	*1*
DR4	productivity	$PD(\sigma;\sigma;\sigma)$	$\exists\sigma_4[\exists\sigma_5[[[\sigma_4\in\sigma_1]\wedge[\sigma_5\in\sigma_2]]\wedge[\sigma_3=(\sigma_4-\sigma_5)]]]$	*3*
DR11	potentiality ordered	$PT(\sigma;\sigma)$	$[\sigma_2\in(\Uparrow\sigma_1)]\leftrightarrow[\forall\sigma_3[[\sigma_3\in\sigma_2]\rightarrow[\sigma_3\in\sigma_1]]]$	*2*
DR12	orelity ordered-pair relation	$OR(\sigma;\sigma;\sigma)$	$[\sigma_3\in(\Uparrow(\sigma_1\times\sigma_2))]\leftrightarrow[\forall\sigma_4[[\sigma_4\in\sigma_3]\rightarrow$ $[\exists\sigma_5[\exists\sigma_6[[[\sigma_5\in\sigma_1]\wedge[\sigma_6\in\sigma_2]]\wedge[\sigma_4=(\sigma_5-\sigma_6)]]]]]]$	*3*
DR13	auto-orelity with respect to σ_1	$AOR(\sigma;\sigma)$	$[\sigma_2\in(\Uparrow(\sigma_1\times))]\leftrightarrow[\forall\sigma_3[[\sigma_3\in\sigma_2]\rightarrow$ $[\exists\sigma_4[\exists\sigma_5[[[\sigma_4\in\sigma_1]\wedge[\sigma_5\in\sigma_1]]\wedge[\sigma_3=(\sigma_4-\sigma_5)]]]]]]$	*2*
DR21	jectivity with respect to σ_1 σ_2	$JR(\sigma;\sigma;\sigma)$	$[OR(\sigma_1;\sigma_2;\sigma_3)]\wedge$ $[\forall\sigma_4[[\sigma_4\in\sigma_1]\rightarrow[\exists\sigma_5[[[\sigma_5\in\sigma_2]\wedge[(\sigma_4-\sigma_5)\in\sigma_3)]]\wedge$ $[\forall\sigma_6[[[\sigma_6\in\sigma_2]\wedge[(\sigma_4-\sigma_5)\in\sigma_3)]]\rightarrow[\sigma_6=\sigma_5]]]]]]]$	*3*
DR22	injectivity	$IJR(\sigma;\sigma;\sigma)$	$[JR(\sigma_1;\sigma_2;\sigma_3)]\wedge[\forall\sigma_4[\forall\sigma_5[\forall\sigma_6[\forall\sigma_7[[[[[[[$ $[\sigma_4\in\sigma_1]\wedge[\sigma_5\in\sigma_1]]\wedge[\sigma_4\neq\sigma_5]]\wedge[\sigma_6\in\sigma_2]]\wedge[\sigma_7\in\sigma_2]]\wedge$ $[(\sigma_4-\sigma_6)\in\sigma_3)]]\wedge[(\sigma_5-\sigma_7)\in\sigma_3)]]\rightarrow[\sigma_6\neq\sigma_7]]]]]]$	*3*
DR23	surjectivity	$SJR(\sigma;\sigma;\sigma)$	$[JR(\sigma_1;\sigma_2;\sigma_3)]\wedge$ $[\forall\sigma_4[[\sigma_4\in\sigma_2]\rightarrow[\exists\sigma_5[[\sigma_5\in\sigma_1]\wedge[(\sigma_5-\sigma_4)\in\sigma_3)]]]]]$	*3*
DR24	bijectivity	$BJR(\sigma;\sigma;\sigma)$	$[IJR(\sigma_1;\sigma_2;\sigma_3)]\wedge[SJR(\sigma_1;\sigma_2;\sigma_3)]$	*3*
DR31	auto-jectivity	$AJR(\sigma;\sigma)$	$JR(\sigma_1;\sigma_1;\sigma_2)$	*2*
DR32 DR33 DR34			auto-injectivity, auto-surjectivity and auto-bijectivity accordingly	*2*
DR35	auto-ject-composity	$AJC(\sigma;\sigma;\sigma;\sigma)$	set σ_1 with auto-injectivities σ_2 and σ_3 composed to produce σ_4	*4*
DR41	card-minor-equality	$\sigma\prec\!\sim\sigma$	$\exists\sigma_3[SJR(\sigma_2;\sigma_1;\sigma_3)]$ order relation	*2*
DR42	card-equality	$\sigma\sim\sigma$	$[\exists\sigma_3[SJR(\sigma_1;\sigma_2;\sigma_3)]]\wedge[\exists\sigma_3[SJR(\sigma_2;\sigma_1;\sigma_3)]]$ or $\exists\sigma_3[BJR(\sigma_1;\sigma_2;\sigma_3)]$ equivalence relation	*2*
DR43	card-minority	$\sigma\prec\sigma$	$[\sigma_1\prec\!\sim\sigma_2]\wedge[\neg[\sigma_1\sim\sigma_2]]$ or $[\exists\sigma_3[SJR(\sigma_2;\sigma_1;\sigma_3)]]\wedge[\neg[\exists\sigma_3[SJR(\sigma_1;\sigma_2;\sigma_3)]]]$	*2*
DR44	ordi-minor-equality	$\sigma\prec\prec\sim\sim\sigma$		*2*
DR45	ordi-equality	$\sigma\sim\sim\sigma$		*2*
DR46	ordi-minority	$\sigma\prec\prec\sigma$		*2*
DR51	finity	$\perp\sigma$	$\exists\sigma_2[[\sigma_2\in\sigma\text{cnl}]\wedge[\sigma_1\sim\sigma_2]]$	*1*
DR52	aleph-nullity	$\#\sigma$	$\sigma_1\sim\sigma\text{cnl}$	*1*
DR53	aleph-unity	$\#\#\sigma$	$\sigma_1\sim\sigma\text{cnpo}$	*1*
DR61	transitivity	$TRANSO(\sigma)$	$\forall\sigma_2[\forall\sigma_3[[[\sigma_2\in\sigma_1]\wedge[\sigma_3\in\sigma_2]]\rightarrow[\sigma_3\in\sigma_1]]]$	*1*
DR62	connexity	$CONNO(\sigma)$	$\forall\sigma_2[\forall\sigma_3[[[\sigma_2\in\sigma_1]\wedge[\sigma_3\in\sigma_1]]\rightarrow[[\sigma_2\in\sigma_3]\vee[\sigma_3\in\sigma_2]]]]$	*1*
DR63	minimality	$MINDO(\sigma)$	$\exists\sigma_2[[\sigma_2\in\sigma_1]\wedge[\forall\sigma_3[[\sigma_3\in\sigma_1]\rightarrow[[\sigma_3=\sigma_2]\vee[\sigma_2\in\sigma_3]]]]]$	*1*
DR64	fundamentality	$FUNDO(\sigma)$	$\forall\sigma_2[[\sigma_2\subseteq\sigma_1)]\rightarrow[\exists\sigma_3[[\sigma_3\in\sigma_2]\wedge[\forall\sigma_4[[\sigma_4\in\sigma_2]\rightarrow$ $[[\sigma_4=\sigma_3]\vee[\sigma_3\in\sigma_4]]]]]]]$	*1*
DR65	totality	$TOTO(\sigma)$	$[TRANSO(\sigma_1)]\wedge[CONNO(\sigma_1)]$	*1*
DR66	limitality	$LIMO(\sigma)$	$[TOTO(\sigma_1)]\wedge[MINDO(\sigma_1)]$	*1*
DR67	ordinality	$ORDIO(\sigma)$	$[TOTO(\sigma_1)]\wedge[FUNDO(\sigma_1)]$	*1*
DR68	cardinality	$CARDIO(\sigma)$	$[ORDIO(\sigma_1)]\wedge[\neg[\exists\sigma_2[[\sigma_2\in\sigma_1]\wedge[\sigma_1\sim\sigma_2]]]]]$	*1*

Table 5 Definition of ***extra-relation-constant*** *strings*

Wherever such a relation is used in a string ***σ_2*** it can be replaced by the ***formula σ_1*** where one only has to keep in mind to adapt the bound ***variable*** strings in the ***formula σ_1*** such that the replacement is compatible ***$\sigma_1 \sim \sigma_2$*** meaning that no ***variable*** is free in one and bounded in the other string.

One can only talk about sets in the abstract calcule of <u>sigma</u> , so all statements have to be reduced to talking about sets. E.g. there are no functions except ones that can be introduced via ***extra-function-constant*** strings as it was done in table 3. No quantion is possible with respect to functions and relations, this would only posible with second-order logic e.g. $\forall\sigma_1(\sigma)[$ or $\exists_2(\sigma)[$. However, making use of oparition $(\sigma-\sigma)$ one can **represent** functions and relations as sets that have special features. A unary function of mapping a set σ_1 to its image σ_2 is called **jection**, it is represented by a set σ_3 that fulfills the jection relation JR $(\sigma_1;\sigma_2;\sigma_3)$. One can define functions of any arity in a similar fashion, e.g. a binary mapping of two sets σ_1 and σ_2 to its image set σ_3 is represented by a set σ_4 that fulfills the binary mapping relation BMR$(\sigma_1;\sigma_2;\sigma_3;\sigma_4)$. This method corresponds exactly to the ***UNEX-formulo*** method as introduced in section 1 ; existence has to be expressed as well as uniqueness. Notice that features of functions are given as ***extra-relation-constant*** strings. All functions represented by sets are **partial** (also called 'conditioned') in the sense that they are not defined for all sets.

The opening line of this section was 'the definition of a ***relation-constant*** *by a* ***formula*** *is straightforward'. For the above examples this is indeed the case. However, in section this statement will be revisited.*

6 Cardinality and diagonalization

Do not mix up the two **different appearances of functions** in axiomatic set theory as it was already mentioned in sections 3 and 5 .

On one side there are functions that are introduced by ***function-constant*** strings like e.g. $(\sigma|\sigma)$, $(\cup\sigma)$, $(\Uparrow\sigma)$, $('\sigma)$ or $(\sigma-\sigma)$.

On the other side functions can be **represented** as sets making use of oparition $(\sigma-\sigma)$ and apply the ***UNEX-formulo*** method as introduced in section 1 . These sets represent functions mapping one set to another, an origin and an image set. Some necessary classifications are listed as ***extra-relation-constant*** strings in table 5. The definitions contain three parts, the first part takes care of mapping the origin set to the image set, the second part guarantees the existence of an image value for every origin value and the third part is necessary for uniqueness. The functions that are introduced by ***function-constant*** strings cannot be represented as sets as they are defined for all sets and not for a given origin set.

At the heart of set theory is the concept of cardinality; after all, that is where Cantor started from. Talking about cardinality one does not always need a concept of so-called **cardinal-numbers**. Cardinality so far is just a façon de parler. One can compare sets with respect to their cardinality by binary relations: $\sigma\prec\sim\sigma$ means that a set has less or equal cardinality with respect to another set, $\sigma\prec\sigma$ means that a set has less cardinality with respect to another set, and $\sigma\sim\sigma$ means that two sets have equal cardinality. Appropriate order and equivalence relations are sufficient for the start of cardinality theory. For the proper definitions one has to deal with special jections.

Let's start with the ***extra-relation-constant*** $\mathrm{OR}(\sigma;\sigma;\sigma)$. The ternary ordered-pair relation **orelity** means that the third argument position is an oparition-set that gives a relation between the first two sets.

$$[\mathrm{OR}(\sigma_1;\sigma_2;\sigma_3)]\leftrightarrow[\sigma_3\in(\Uparrow(\sigma_1\times\sigma_2))]\leftrightarrow[\forall\sigma_4[[\sigma_4\in\sigma_3]\rightarrow[\exists\sigma_5[\exists\sigma_6[[[\sigma_5\in\sigma_1]\wedge[\sigma_6\in\sigma_2]]\wedge[\sigma_4=(\sigma_5-\sigma_6)]]]]]]$$

The ternary relation **jectivity** given by ***extra-relation-constant*** $\mathrm{JR}(\sigma;\sigma;\sigma)$ means that the third argument position is a unex-oparition-set that represents a jection of one set to another set (existence and uniqueness required); it refers to a unary function, higher arities would read like e.g. a binary function $\mathrm{JR}(\sigma;\sigma;\sigma;\sigma)$.

$$[\mathrm{JR}(\sigma_1;\sigma_2;\sigma_3)]\leftrightarrow[[\mathrm{OR}(\sigma_1;\sigma_2;\sigma_3)]\wedge$$
$$[\forall\sigma_5[[\sigma_5\in\sigma_1]\rightarrow[\exists\sigma_6[[[\sigma_6\in\sigma_2]\wedge[(\sigma_5-\sigma_6)\in\sigma_3)]]\wedge[\forall\sigma_7[[[\sigma_7\in\sigma_2]\wedge[(\sigma_5-\sigma_7)\in\sigma_3)]]\rightarrow[\sigma_7=\sigma_6]]]]]]]]$$

The ternary relation **injectivity** given by ***extra-relation-constant*** $\mathrm{IJR}(\sigma;\sigma;\sigma)$ means that the third argument position is a unex-oparition-set that represents an injection of one set to another set.

$$[\mathrm{IJR}(\sigma_1;\sigma_2;\sigma_3)]\leftrightarrow[[\mathrm{JR}(\sigma_1;\sigma_2;\sigma_3)]\wedge[\forall\sigma_4[\forall\sigma_5[\forall\sigma_6[\forall\sigma_7[[[[[[[$$
$$[\sigma_4\in\sigma_1]\wedge[\sigma_5\in\sigma_1]]\wedge[\sigma_4\neq\sigma_5]]\wedge[\sigma_6\in\sigma_2]]\wedge[\sigma_7\in\sigma_2]]\wedge[(\sigma_4-\sigma_6)\in\sigma_3)]]\wedge[(\sigma_5-\sigma_7)\in\sigma_3)]]\rightarrow[\sigma_6\neq\sigma_7]]]]]]]$$

The ternary relation **surjectivity** given by ***extra-relation-constant*** $\mathrm{SJR}(\sigma;\sigma;\sigma)$ means that the third argument position is a unex-oparition-set that represents a surjection of one set to another set

$$[\mathrm{SJR}(\sigma_1;\sigma_2;\sigma_3)]\leftrightarrow[[\mathrm{JR}(\sigma_1;\sigma_2;\sigma_3)]\wedge[\forall\sigma_4[[\sigma_4\in\sigma_2]\rightarrow[\exists\sigma_5[[\sigma_5\in\sigma_1]]\wedge[(\sigma_5-\sigma_4)\in\sigma_3)]]]]]]$$

The ternary relation **bijectivity** given by ***extra-relation-constant*** $\mathrm{BJR}(\sigma;\sigma;\sigma)$ means that the third argument position is a unex-oparition-set that epresents a bijection of one set to another set, but one should notice that σ_3 only applies to the direction σ_1 to σ_2 ; the other direction needs a different σ_3 .

$$[\mathrm{BJR}(\sigma_1;\sigma_2;\sigma_3)]\leftrightarrow[[\mathrm{IJR}(\sigma_1;\sigma_2;\sigma_3)]\wedge[\mathrm{SJR}(\sigma_1;\sigma_2;\sigma_3)]]$$

Using these ***extra-relation-constant*** strings one can define the order relations and equivalence relations that allow for introducing cardinality as mentioned before.

Table 5 contains the definitions of three binary relations: card-minor-equality $\sigma\prec\sim\sigma$, card-equality $\sigma\sim\sigma$ and card-minority $\sigma\prec\sigma$. $\sigma\prec\sim\sigma$ is a total order relation as it is reflexive, antisymmetric, transitive and connective. $\sigma\sim\sigma$ is an equivalence relation relation as it is reflexive, symmetric and transitive.

One can express card-equality using surjectivity only as there is the following ***THEOREM***

$$\forall\sigma1[\forall\sigma2[[\sigma1\sim\sigma2]\leftrightarrow[[\exists\sigma3[SJR(\sigma1;\sigma2;\sigma3)]]\wedge[\exists\sigma3[SJR(\sigma2;\sigma1;\sigma3)]]]]]$$

Applying it to a set and its potention set one gets the ***general-diagonal-sentence***

$$\forall\sigma1[\neg[\exists\sigma2[BJR((\Uparrow\sigma1);\sigma1;\sigma2)]]]$$

$$\forall\sigma1[\neg[\sigma1\sim(\Uparrow\sigma1)]]$$

and with choice of σcnl for $\sigma1$ the special ***cantor-diagonal-sentence***

$$\neg[\exists\sigma1[BJR((\Uparrow\sigma cnl);\sigma cnl\ ;\sigma1)]]$$

$$\neg[\sigma cnl\ \sim(\Uparrow\sigma cnl)\ ]$$

With the representations of relations one can do Cantor diagonalization and state the ***general-diagonal-sentence*** as a ***THEOREM*** in calcule <u>sigma</u> with first-order logic FOL :

$$\forall\sigma1[\neg[\exists\sigma2[[\sigma2\in(\Uparrow(\sigma1\times))]\wedge[\forall\sigma3[[\sigma3\in(\Uparrow\sigma1)]\rightarrow[\exists\sigma4[[\sigma4\in\sigma1]\wedge[\forall\sigma5[[\sigma5\in\sigma3]\leftrightarrow[(\sigma4-\sigma5)\in\sigma2]]]]]]]]]$$

It reads in its most condensed form

$$\forall\sigma1[\sigma1\prec(\Uparrow\sigma1)]$$

It is remarkable, but usually not mentioned, that Cantor diagonalization is a general feature of all second-order logic calcules, may they be concrete or abstract. No use is made of 'numbers'. The second-order pseudo-calcule is called <u>Phitonpython</u> with ***sort*** $\phi\varpi$.

The indirect proof of Cantor for the ***THEOREM***
$\neg[\exists 1(\phi\varpi;\phi\varpi)[\forall 1(\phi\varpi)[\exists\phi\varpi1[\forall\phi\varpi2[[1(\phi\varpi2)]\leftrightarrow[1(\phi\varpi1;\phi\varpi2)]]]]]]$

has the choice of a counter-example $\phi\varpi2=\phi\varpi1$ and $[1(\phi\varpi1)]\leftrightarrow[\neg[1(\phi\varpi1;\phi\varpi1)]]$

Following the indirect proof of Cantor for the ***sentence*** one gets the proof of the above ***THEOREM*** of axiomatic set theory by contradiction with a counter-example obtained by diagonalization:

replace $\sigma5\in\sigma3$ by equivalent $(\sigma5-\sigma5)\in(\times\sigma3)$
take $\sigma5=\sigma4$ and subset $(\times\sigma3)=(((\sigma1\times)/\sigma2)\cap(\times\sigma1))$
and realize that $(\sigma4-\sigma4)\in(((\sigma1\times)/\sigma2)\cap(\times\sigma1))$ contradicts $(\sigma4-\sigma4)\in\sigma2$

The simplest case of the ***THEOREM*** is the original ***cantor-diagonal-sentence*** where $\sigma1$ is taken as σcnl

$$\neg[\exists\sigma1[[\sigma1\in(\Uparrow(\sigma cnl\times))]\wedge[\forall\sigma2[[\sigma2\in(\Uparrow\sigma cnl)]\rightarrow[\exists\sigma3[[\sigma3\in\sigma cnl]\wedge[\forall\sigma4[[\sigma4\in\sigma2]\leftrightarrow[(\sigma3-\sigma4)\in\sigma1]]]]]]]]$$

In conventional language it says: the set of natural numbers has less cardinality than its power-set (given by potention). Or there is no surjection of natural numbers to their properties. It reads in its most condensed form as follows:

$$\sigma cnl\prec(\Uparrow\sigma cnl)$$

7 Proof-complete and basis-complete

The expressions **complete** and its negation **incomplete** appear in two different meanings with respect to a logical system. It would be better if a system were called

proof-incomplete if there are true ***sentence*** strings that cannot be proven within the system

basis-incomplete if there are ***sentence*** strings that that do not contradict the system, nor do their negations contradict the system.

Gödel's famous completeness and incompleteness theorems mean proof-completeness and proof-incompleteness. *The author thinks that the expression 'complete' is not a very good choice. There is nothing **missing** in a proof-incomplete system that one could add to make it complete. It would be better to talk about autarkic and non-autarkic systems. In an autarkic system the truth of **sentence** strings is derivable within the system, whereas in a non-autarkic system the truth s comes from the outside.*

*The expression 'complete' is a much better choice in the second case, as the systems actually lack something. At least in certain cases incomplete systems can be made complete by new **Axiom** strings.*

The simplest example for an basis-incomplete system is the abstract calcule gamma of group theory where neither the simple commutability ***sentence*** $\forall\gamma_1[\forall\gamma_2[(\gamma_1\otimes\gamma_2)=(\gamma_2\otimes\gamma_1)]]$ nor its negation contradict the ***Axiom*** strings. Traditionally the most important example is the basis-incomplete absolute planar geometry, where Euclid's ***Axiom*** of unique parallels or its negation, Lobachevsky's ***Axiom*** of multiple parallels can be added. Both Euclidean and Lobachevskyan geometries are basis-complete.

The following figure shows the various classifications of ***sentence*** strings with respect to ***TRUTH*** in the systems of the FUME-method that are called concrete calcules or abstract calcules.

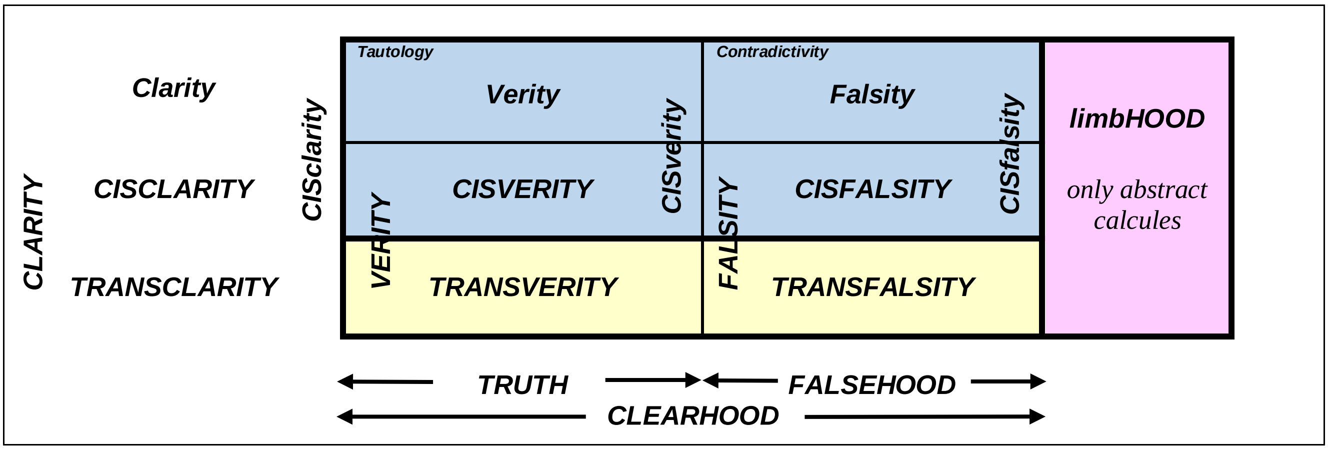

*Figure 2 Classification of **sentence** strings with respect to **TRUTH***

The TRUTH of a clarity needs at most limitive logic (predicate logic with limited quantions).
The TRUTH of a CISCLARITY sentence needs a special proof and quantive logic (predicate logic).
The TRUTH of a TRANSCLARITY sentence cannot be found within the calcule.

For a **proof-incomplete** calcule the yellow area of TRANSCLARITY is not empty: there are true sentences like the famous Gödel-sentence (of Gödel's so-called 'incompletnesss theorem') that cannot be proven from axioms.

In a **basis-complete** calcule it holds that every sentence is either a TRUTH or a FALSEHOOD : e.g. in Euclidean geometry every sentence is either true or false, tertium non datur. In a basis-incomplete calcule the purple area of limbHOOD is not empty: there are sentences that are neither true nor false, they are can be called **limbic** (the negation of **clear**), as they are so to speak 'in limbo'; . The simple example of absolute geometry theory can be made basis-complete by adding an axiom of parallelity. This is not the case for the famous example of axiomatic set theory with the sentence of Cantor's continuum hypothesis that shows that it is not basis-complete.

8 Continuum-hypothesis and basis-incompleteness

An important problem at the very center of set theory is called the **Continuum Hypothesis** CH . Its simplest form is given by the ***cantor-continuum-sentence*** which states that there are no sets with cardinality between σcnl and its potention (⇑σcnl) where σcnl is the infinite set created from the nilset σn by successive singlition (|σ) . *With the concept of cardinal numbers one says that there is no cardinal number between aleph-zero and aleph-one*. Observe the use of boldface italics fonts in ***σcchs =*** for distinguishing metalanguage Mencish from object-language Funcish:

σcchs = ∀σ1[[[σcnl⎨~σ1]∧[σ1⎨~(⇑σcnl)]]→[[σ1~σcnl]∨[σ1~(⇑σcnl)]]]

Alternative formulation

σcchsa = ¬[∃σ2[[σcnl⎨σ2]∧[σ2⎨(⇑σcnl)]]]

The ***general-continuum-sentence*** claims that all sets between a set and its potention have either the cardinality of the set or the cardinality of its potention.

σgchs = ∀σ3[∀σ4[[[σ4⎨~σ3]∧[σ3⎨~(⇑σ4)]]→[[σ3~σ4]∨[σ3~(⇑σ4)]]]

Kurt Gödel had shown in 1938 : if ZF is consistent so is ZFC+CH (Continuum Hypothesis),
Gödel, K. (1940). The Consistency of the Continuum-Hypothesis. Princeton University Press

Paul Cohen had shown in 1963 : if ZF is consistent so is ZFC+ negation of CH
Cohen, Paul J. (December 15, 1963). "The Independence of the Continuum Hypothesis". Proceedings of the National Academy of Sciences of the United States of America. ***50*** *(6): 1143–1148*

The **Gödel-Cohen** metasentence says that Cantor's continuum hypothesis can neither be proven nor can its negation be proven in the framework of axiomatic set theory. The ***cantor-continuum-sentence*** was the first example of a ***sentence*** that was shown to be independent of ZFC . This means that axiomatic set theory is **basis-incomplete** as can be expressed in metalanguage:

∃σ1[[sentence(σ1)] ∧ [¬ [[TRUTH(σ1)] ∨ [TRUTH(¬[***σ1***] ***)]]]]***

The ***cantor-continuum-sentence*** is neither true nor false: it is a **limbic** ***sentence***

¬ [[TRUTH(σcchs)] ∨ [TRUTH(¬[***σcchs***] ***)]]***

9 A weird ***formula*** *and the* ***Separation-axiom***

In sections 1 and 2 ***formula*** strings were introduced in an abbreviated fashion. This should be sufficient for this publication, as ***formula*** strings correspond exactly to the conventional use of the expression 'formula' in logical systems. It is also clear what is meant by a ***unary-formula*** string as ***formula*** that contains exactly one free ***variable*** and a ***norm-formula*** string that has successive free ***variable*** strings σ_1, σ_2, σ_3 … and a ***unary-norm-formula*** string with exactly one free ***variable*** σ_1 .

Returning to the opening line in section 5 'the definition of a ***relation-constant*** by a ***formula*** is straightforward' one can have second thoughts if that is so in general. The reason for this investigation are the results of the two preceding sections on basis-incompleteness and Cantor's continuum hypothesis. One starts with the following ***unary-norm-formula*** that make use of the alternative expression of the ***cantor-continuum-sentence*** :

$$\boldsymbol{\sigma}\textbf{\textit{weird-formula}} = [\sigma_1 \neq \sigma n] \vee [[\sigma_1 = \sigma n] \wedge [\boldsymbol{\sigma}\textbf{\textit{cchsa}}\ ]]$$
$$\boldsymbol{\sigma}\textbf{\textit{weird-formula}} = [\sigma_1 \neq \sigma n] \vee [[\sigma_1 = \sigma n] \wedge [\neg[\exists \sigma_2[[\sigma cnl \prec \sigma_2] \wedge [\sigma_2 \prec (\Uparrow \sigma cnl)]]]]]$$

It is not a ***formula*** that gives a value 'true' or 'false' for every argument, as it is **limbic** at $\sigma_1 = \sigma n$. There is no way to know beforehand in axiomatic set theory if a ***formula*** is clear everywhere. One has to notice the important (usually ignored) general fact that in an incomplete theory there are limbic ***formula*** strings besides limbic ***sentence*** strings as well. In the preceding section 8 the ***cantor-continuum-sentence*** has been introduced which is limbic, and it was mentioned that there are other limbic ***sentence*** strings in axiomatic set theory as well.

There are many ***formula*** strings like $\boldsymbol{\sigma}$***weird-formula*** . One cannot determine beforehand if a ***formula*** is clear or limbic, it has to be checked for every instance, and it poses the same problem in principle as determining if a ***sentence*** string is clear or limbic.

What does that mean in connection with the ***Separation-axiom*** strings ? They are based on ***formula*** strings without any further restrictions. The simplest ***Separation-axiom*** strings are obtained from the following mater with ***unary-norm-formula*** strings $\boldsymbol{\sigma_1}$ with free ***variable*** σ_1 , it says that there exists a set σ_0 whose elements are those of a set σ_2 that also fulfill the ***formula*** strings $\boldsymbol{\sigma_1}$:

$$\forall \sigma_2[\exists \sigma_0[\forall \sigma_1[[\sigma_1 \in \sigma_0] \leftrightarrow [[\sigma_1 \in \sigma_2] \wedge [\boldsymbol{\sigma_1}\ ]]]]]$$

Choose $\sigma_2 = \sigma cnl$ and $\boldsymbol{\sigma_1} = \boldsymbol{\sigma}$***weird-formula***

$$\exists \sigma_0[\forall \sigma_1[[\sigma_1 \in \sigma_0] \leftrightarrow [[\sigma_1 \in \sigma cnl] \wedge [\boldsymbol{\sigma}\textbf{\textit{weird-formula}}\ ]]]]$$
$$\exists \sigma_0[\forall \sigma_1[[\sigma_1 \in \sigma_0] \leftrightarrow [[\sigma_1 \in \sigma cnl] \wedge [[\sigma_1 \neq \sigma n] \vee [[\sigma_1 = \sigma n] \wedge [\ \neg[\exists \sigma_2[[\sigma cnl \prec \sigma_2] \wedge [\sigma_2 \prec (\Uparrow \sigma cnl)]]]]]]]]$$

Choose $\sigma_1 = \sigma n$ as the empty set, observe $\sigma n \in \sigma cnl$,

$$\exists \sigma_0[[\sigma n \in \sigma_0] \leftrightarrow [[\sigma n \in \sigma cnl] \wedge [[\sigma n \neq \sigma n] \vee [[\sigma n = \sigma n] \wedge [\ \neg[\exists \sigma_2[[\sigma cnl \prec \sigma_2] \wedge [\sigma_2 \prec (\Uparrow \sigma cnl)]]]]]]]]$$

$$\exists \sigma_0[[\sigma n \in \sigma_0] \leftrightarrow [\neg[\exists \sigma_2[[\sigma cnl \prec \sigma_2] \wedge [\sigma_2 \prec (\Uparrow \sigma cnl)]]]]]$$
$$\exists \sigma_0[[\sigma n \in \sigma_0] \leftrightarrow [\boldsymbol{\sigma}\textbf{\textit{cchsa}}\ ]]$$

This, however, is not a proper ***TRUTH*** as it is required for a set that must be definite if another set is contained in it or not: The binary ***relation-constant*** $\sigma \in \sigma$ is defined everywhere, one must not exclude certain elements.

An even simpler weird ***formula*** could be taken as $[\sigma_1 = \sigma_1] \wedge [\boldsymbol{\sigma}\textbf{\textit{cchsa}}\ ]$ and one could construct many more limbic ***formula*** strings.

Other ***formula*** strings may be limbic (i.e. indefinite with respect to truth) but the ***formula*** expressed by $\sigma_1 \in \sigma_2$ with the basis-relation of membrity $\sigma \in \sigma$ must be clear (i.e. definite with respect to truth for all instances).

In other words: one can obtain ***sentence*** strings as ***Separation-axiom*** strings that are not meaningful ***TRUTH*** strings. One has to refute ***Separation-axiom*** strings in the usual fashion. And it seem doubtful if one can replace them by something else, as there is no way to determine beforehand if a ***formula*** is clear, i.e. if it is true or false for all instances.

Is there more than a flaw in axiomatic set theory ?

The situation is different for the appearance of ***formulo*** strings in the ***Recursion-axiom*** and ***Replacition-axiom*** strings as they are qualified as ***UNEX-formulo*** strings, a feature that can be expressed within metalanguage.

In some axiomatic set theories there are no separation axioms, as the corresponding sentences can be derived as **theorems**, but this does not change the problem, it just moves it to another level.

The author is deeply worried as the ***Separation-axiom*** strings are used right from the very beginning: e.g. intersectition $(\cap\sigma)$, intersection $(\sigma\cap\sigma)$, complemention (σ/σ) , production $(\sigma\times\sigma)$ and bi-potentiation $(\sigma\times)$ necessitate ***Separation-axiom*** strings. And so one has to end with the question:

How can one solve the ***Separation-axiom*** *problem of axiomatic set theory ?*

It would be no problem to replace the ***Separation-axiom*** mater by a finite number of ***Axiom*** strings that alloe for the introduction of intersection, complemention, production etc. . *However, the author is afraid that this will not be enough, but that is not his problem.*

10 Classes ?

The problem of limbic ***formula*** strings (i.e. indefinite with respect to truth) is also very important in connection with **t**he **comprehension** principle that is necessary for the definition of so-called **classes**. Classes are introduced as mathematical objects that can be unambiguously defined by a property that all its members share. For classes in axiomatic set theory this means that their member sets fulfill a ***formula***. A ***formula*** string is the only way to represent a property in the first-order logic system of ZFC.

Now it is important to notice three facts with respect to ***formula*** strings:

- they are enumerable as they are strings with charcaters of afinite alphabet, which implies that classes are **enumerable** . The cardinality of classes is so-to-say aleph-zero.
- they are not necessarily clear (i.e. definite with respect to truth) which implies that one does **not know** in general if a ***formula*** string leads to a class
- even clear ***formula*** strings pose a problem with respect to equivalence; it is **not decidable** in general if two clear ***formula*** strings are true for the same instances and therefore lead to the same class.

By the way: in the appendix a proposal is put forward how to incorporate classes into set theory in a rigorous fashion.

In literature one finds flowery statements like '*classes are described as equivalence classes of logical formulas*' . Or '*classes are collections of sets that can be unambiguously defined by a property that all its members share'* whatever a **collection** is - there is no such expression in first-order logic.

11 Properties and relation-constant strings in incomplete calcules

Functions can be represented by ***UNEX-formulo*** strings in all calcules, where this metaproperty has to be proven in every single case (indicated by the capital letters). In concrete calcules and basis-complete abstract calcule ***formula*** strings represent relations. However, in abstract calcules that are basis-incomplete things may be different.

The problem of limbic ***formula*** strings like ***σweird-formula*** of section 9 leads to a second look at the introduction of ***extra-relation-constant*** strings in section 5 . It was done in two steps, firstly one set up the ***relation-constant*** strings as names, secondly one put life into them by writing down the ***formula*** string they are supposed to replace. Remember, ***extra-relation-constant*** strings were introduced as abbreviations of ***formula*** strings, both having the same arity where a little care had to be taken with respect to ***variable*** strings. Nobody had any scruples when introducing the two examples, binary equi-subity $\sigma\subseteq\sigma$ and ternary jectivity JR(σ;σ;σ) . But should one have had scruples?

The abstract calcule of axiomatic set-theory <u>sigma</u> is basis-incomplete, as was discussed in section 7. This lead to the statement that <u>sigma</u> allows for limbic ***sentence*** strings, which was not astonishing. However, in section 8 it was shown that there are limbic ***formula*** strings mas well. Actually one does not know beforehand, if a ***formula*** string is limbic or clear. The real problem lies in the fact that one **cannot even express** this using proper object and metalanguages, Funcish and Mencish, but only in supralanguage. There is no metaproperty ~~***COMPLETE-formula***~~ that could be used to properly introduce ***extra-relation-constant*** strings. One has to show it for every ***formula*** string by a special consideration. But this means that one **cannot** use in general an ***extra-relation-constant*** string to introduce a relation in incomplete calcules.

By the way: as a first-order logic calcule axiomatic set-theory <u>sigma</u> does **not allow** to talk about **properties** of sets.Not even about properties of natural numbers that are represented by natral set σvnl of table 4 . Properties of natural numbers are not enumerable (uncountable), whereas natural numbers are enumerable (countable). The best one can do is to introduce ***extra-property-constant*** strings. However, as strings of a finite alphabet they are enumerable (countable) and therefore cannot represent all properties, a feature that is usually ignored in the investigations of axiomatic set theory. This is another problem besides the fact that ***extra-property-constant*** are not necessarrily clear (truth-definite).

It looks like one has to abandon table 5 of section 5. Even for the simple binary relation of equi-subity $\sigma\subseteq\sigma$ one cannot be sure if it can be introduced appropriately. Or can you derive from the ***Axiom*** strings that the defining ***formula*** $\forall\sigma3[[\sigma3\in\sigma2]\rightarrow[\sigma3\in\sigma1]]$ is either true or false for all instances of σ1 and σ2 ? And how would you you express this in object or metalanguage? One has reached quite a **dramatic** point in the investigation of the abstract calcule of axiomatic set-theory <u>sigma</u> .

*Our friend Vincent van Gogh went so far as to talk about **hoaxiomatic** set theory*

*Axiomatic set theory is considered as the means to talk about so-called **actual infinities** in a precise way, using first-order logic. Axiomatic set theory is thought to be **the** mathematical theory on infinity.*

*But perhaps one **cannot outsmart infinity.***

Appendix Proposal for a rigorous introduction of classes in bi-calcule sigma_sisi

In order to introduce classes in a rigorous fashion the following is proposed. Replace calcule sigma by an enriched bi-calcule sigma_sisi with two ***sort*** strings, besides σ for sets one has another one, 'sisi' σσ for classes, furthermore the ontological basis contains two more binary relations σ∈σσ and σσ∈σσ as well as a property (unary relation) setity ∴σσ , meaning that a class corresponds uniquely to a set. In addition to ***Axiom*** strings ***A1*** to ***A10m*** and ***A11*** there are more ***Axiom*** strings for the new relations.

I do not know if this corresponds in some way to Arnold Oberschlep's kind of class-theoretical LC-logic. Jürgen-Michael Glubrecht, Arnold Oberschelp, Günter Todt: Klassenlogik. Bibliographisches Institut, Mannheim u. a. 1983. Probably my fault, but I do not understand this book.

Abstraction principle is usually formulated only for the unary case, *y∈{x:A(x)} iff A(y)* ; it is expressed properly with a ***unary-formula σ1*** . However later it is made use of the multary case without mentioning it. Therefore there are two ***Axiom*** matres for the unary and the multary case.

A12u ***∀σ1[[sentence(*** ∀σ1[***σ1***] ***)] → [Axiom(*** ∃σσ1[∀σ1[[σ1∈σσ1]↔[***σ1***]]] ***)]]***

A12m ***∀σ1[∀σ2[∀σ3[[[omny(σ2)] ∧ [sentence(σ2*** ∀σ1[***σ1***] ***σ3)]] →***
[Axiom(σ2 ∃σσ1[∀σ1[[σ1∈σσ1]↔[***σ1***]]] ***σ3)]]]]]***

Only then one can use ***binary-formula σ1 =*** σ1∈σ2 in order to show the trivial ***THEOREM*** that every set is a class.. One also defines the ***individual-constant*** collection class of all sets σσcs by putting ***unary-formula σ1 =*** σ1=σ1 into abstraction principle: There is ***Euriscom*** (implicit definition)
DI21 ∀σ1[[σ1∈σσcs]↔[σ1=σ1]]

But there is no possibility to express (the other way round) as an ***Axiom*** mater that for every class σσ1 there exists a ***unary-formula σ1*** with free ***variable*** σ1 such that ∀σ1[[σ1∈σσ1]↔[***σ1***]]

Extensionality principle two classes are equal if their set elements are the same *A = B iff (x∈A iff x∈B)* ; it is expressed properly. It gives ***Axiom***

A13 ∀σσ1[∀σσ2[[σσ1=σσ2]↔[∀σ1[[σ1∈σσ1]↔[σ1∈σσ2]]]]]

Comprehension principle *{x:A(x)}∈B iff there exists y such that y={x:A(x)} and y∈B*
It firstly necessitates the definition of σσ∈σσ which only holds if the left argument of σσ1∈σσ2 is a set, i.e. ∴σσ1 ! There are the two ***Euriscom*** strings

DR71 ∀σσ1[∀σσ2[[σσ1∈σσ2]↔[∃σ1[[∀σ2[[σ2∈σσ1]↔[σ2∈σ1]]]∧[σ1∈σσ2]]]]]
DR72 ∀σσ1[[∴σσ1]↔[∃σ1[∀σ2[[σ2∈σσ1]↔[σ2∈σ1]]]]]

Then there is the trivial mater that expresses that a class can be the member of another class only if there is a set with the same extension (big deal !). It gives ***THEOREM*** (not an ***Axiom***) **mater of comprehension**:

∀σ1[[sentence(∀σ1[***σ1***] ***)] →***
[THEOREM(∀σσ1[[∀σ1[[***σ1***]↔[σ1∈σσ1]]]→[∀σσ2[[σσ1∈σσ2]↔[∃σ1[[***σ1***]∧[σ1∈σσ2]]]]]] ***)]]***

One can define as ***extra-relation-constant*** strings σσ⊆σσ , σσ⊂σσand ***extra-function-constant*** strings (σσ∩σσ) , (σσ∩σσ) , (σσ×σσ) e.g. the most complicated case of class production:

∀σ1[∀σ2[[[sentence(∀σ1[***σ1***] ***)] ∧ [sentence(*** ∀σ1[***σ2***] ***)]] →***
[Euriscom([[∀σσ1[∀σ1[[σ1∈σσ1]↔[***σ1***]]]]∧[∀σσ2[∀σ1[[σ1∈σσ1]↔[***σ2***]]]]]→
[∀σ1[[σ1∈(σσ1×σσ2)]↔[∃σ2[∃σ3[[[***(σ1;*** σ1∫σ2***)***]∧[***(σ2;*** σ1∫σ3***)***]]∧[σ1∈(σ2–σ3)]]]]]] ***)]]]***