NF is consistent (no longer main official version)

M. Randall Holmes

February 8, 2016

1 Version remarks

2/8/2016: No longer the main version. It could be fixed up: needs a better argument that the recursion defining the equivalence of codes is well-founded, which I do know how to supply and have given elsewhere.

1/20/2016: Corrected bug in the definition of relative type conditions on argument lists (replace a \(\lambda\) with a \(\kappa\)).

1/15/2016: Corrected error in the definition of bounded formal element.

1/10/2016: Happy New Year! A typo fix. One should look at thepaper.tex for my current thinking, but I’m maintaining this paper. I may make some changes to bring it closer to section 10 of the other paper, though I will preserve the old formula for parent sets (possibly changing the formula for \(\Pi(\emptyset)\)). Also at least attempted to change all notations for parent set operation to \(\Pi\).

12/24/2015: Fixed some language in the proof of the elementarity property.

12/21/2015: The modified definition of set codes and proof of the substitution property satisfying the points under the note of 12/16/2015 has been imported from the slides. I added some clarification about the invertibility of substitution extensions which needs to be reflected back into the slides.
12/16/2015: This version needs a correction which is outlined in the slides newslides1.pdf. There is a difficulty with the recursion in the definition of the main construction, interacting with the way the Substitution Property is proved, which was correctly handled in the version Bowler and his group read, but was oversimplified in this version. In the slides I outline an approach which I think works and is perhaps easier to read than the version in the paper Bowler’s group read. I expect to install the correction shortly.

12/13/2015: 4 pm Please see in the latest slides a revision of the proof of the Substitution Property (describing the process of construction of substitution extensions explicitly), which will shortly be imported into this document.

12/12/2015: 10 am further changes bringing notation in line with that of the slides.

9:30 am new text for proof of the Substitution Property imported from the slides.

The proof of the substitution property requires a correction (eliminating the List Structure Lemma) which is not yet found in this text but can be seen in the slides. Also, the definition of equivalence for set codes has been restored to an earlier form due to concerns about well foundedness of the definition of equivalence: both of these points will shortly be fixed, and the new text can be seen at least in preliminary form in the slides.

11/17/2015 6 pm: Further revisions of problems discovered while setting up the slides. Further, there really doesn’t seem to be a sensible way to get the Hilbert symbol approach to streamlining ambiguity to play nicely with tangled webs, so appeals to Specker’s original argument appear to be unavoidable. These considerations do not affect our main argument in any way, but it would be nice to improve the situation. Clearing this up would make what we say about $\omega$- and $\alpha-$ models more convincing, though it is entirely standard (adapted from the NFU proof).

11/17/2015: Added a treatment of Specker’s ambiguity results simplified by adding Hilbert symbols to the language of our type theory.
11/10/2015: Preparing slides. Correcting some errors I notice as I go. Notably the injectivity and disjointness conditions on range elements of argument lists were not stated correctly.


11/6/2015: Technical change to argument lists: domains are simply small subsets of \( \kappa \). I think this will simplify discussions of redundancy of arguments, as items can simply be dropped from codes (and argument list types) without reindexing. I have extracted the basic result on redundancy of arguments as a separate Lemma in the section with the Permutation Lemma and List Structure Lemma, which makes the proof that all symmetric sets are codable less cumbersome. This looks like a serious improvement, though it does change the way things look: we no longer have \( g[L \circ M] \) as the code for a parent of an near litter item \( L(\beta) \) in the range of \( L, M \) a strictly increasing sequent of ordinals less than \( \beta \), but \( g[L[M] \), where \( M \) is a subset of the domain of \( L \) included in \( \beta \). 3 pm refinements, correction of typos and typing errors.

The arxiv version is still the previous one. I expect that with a change like this it will take a little while to find all the induced minor type errors. But I think this is useful.

11/5/2015: Index fixed to cover new section 8 and some other bits. Reconciliation of the new section 8 and subsequent sections worked on. 4:30 pm refinements.

11/4/2015: The important result on redundancy of arguments in codes was accidentally omitted here: I’m inserting its statement in the right place and will insert the proof (already stated in earlier versions) as soon as I have time. The correct statement of the theorem that needs redundancy of arguments is now given; the proof of that theorem still needs to be revised (10 am). Added the missing argument though it may need further polish. (7 pm)

11/3/2015: A serious remodel. Section 8 of this version contains a simultaneous development of abstract and concrete notation for atoms in clans. There might be problems with consistency of notation in later sections with that in Section 8.
The index will need repair, as entire sections in which basic concepts are defined have been replaced by section 8, which does not include any index references yet.

11/2/2015: This is in the middle of a major edit. Section 8 in this version is intended to completely replace sections 9, 11 and 15 as a description of the construction (handling introduction of atoms and infinitary notation considered abstractly on its own simultaneously), but has to be supported by additional proofs. I believe there are some errors in the treatment of atoms with parents in the unique pure set which is a parent set in sections 9 and 10, but these are corrected in the section 8 treatment.

5:10 pm corrected an error in the definition of formal element of a set code in section 8 and tightened up the definition of argument list type.

10/22/2015: Some edits to early sections. Corrected a silly mental mistake I was making which made hash of the first draft of the current version of the first discussion of redundancy of arguments. Somehow I was writing $L \setminus M$ and $M^* \setminus L$.

10/16/2015: Minor note to self under the 10/15 remarks. Further revisions 2 pm. Further fiddling with redundancy of arguments passage 5:15.

10/15/2015: Reading through and trying identify rough spots.

**Specker ambiguity results:** It would be nice to have a self-contained account of the passage from TST + Ambiguity to NF which worked for tangled web approaches as well as tangled type theory approaches. This would be especially useful for making it clear that the results about $\omega$-models transfer correctly. At the very least, there should be a discussion of how Specker’s method applies to the results about $\omega$-models. This could also be done by noting that if the language over which the elementarity property of tangled webs holds includes a Hilbert symbol, the passage from TST + Ambiguity to NF is direct (as in tangled type theory), and showing further that a Hilbert symbol is supported. It might also be possible to show that we get a model of TTT from the main construction. None of this has been done in the text; I am thinking about it. This particular rough spot does not affect the
proof of the main result, but it would be useful to clean it up to make it clearer that extensions work as advertised. 10/16 added language covering this to the relevant subsection.

**redundancy of arguments:** The discussion of redundancy of arguments in the abstract is a bit nasty to read, though it is correct and important. 10/16 rewrote this little section.

**considerations of use and mention:** One has to pay attention to the difference between elements of ranges of argument lists and the objects they refer to: there is often abuse of language in this regard, which I am trying to correct where I find it.

**abstract notation arguments:** The fact that the List Structure Lemma holds for the abstract notation needs to be explained, which will require an account of action of permutations on abstract notations. It’s all quite parallel to what happens with concretely grounded notations, but this needs to be laid out.

**cardinals of infinite Specker rank in ZF:** For the methods here to show that these exist, we need to explore whether tangled webs constructed in the manner given here can be replicated in symmetric models over forcing extensions. This is only relevant to remarks in the conclusion: for our main purposes, results in ZFA are sufficient.

**how evident are the abstraction lemmas?** I think that these can be explained reasonably directly. An essay at this appears in the text, 10/16.

**certain passages of prose:** There are some passages of quite dense prose which need to be made more accessible. Redundancy of arguments discussion, already noted. The discussion of why the abstract construction implementing the unlikely definition works. Other prose passages do not seem bad to me but I should expect some readers to complain about some of them.

**October 8, 2015:** Reading through carefully and making local corrections.

**October 6, 2015:** The notion of formal element in the abstract notation needs to be restricted to allow the recursive definition of equivalence to work correctly. The treatment of redundancy of arguments in codes and the notion of formally restricted code has been tweaked.
October 2, 2015: Pulled out a separate section with the results on applying permutations to argument lists in codes. The statements of some of these results in the brief summary that appeared in the previous version were incorrect or misleading (at any rate too facile); writing out the proofs in detail forces one to do it right. What I need to be true is true, and I have proved these things in other versions; this is just one of the perils of writing out a new version. 1:15 pm further improvements.

Further work on formal restriction of arguments.

October 1, 2015: Desiderata noted while reading: add a reference for the general FM construction. I wonder if the Specker results for tangled webs can be tidied up into a form which can be outlined in the paper. The Specker results for $\omega-$ and $\alpha-$ models might seem fragile when working in tangled webs rather than tangled types (they aren’t, but it would be nice to make this clear). Of course if the theory contains a Hilbert symbol which can be used in the elementarity condition of tangled webs all is well; can the theory of the main construction be equipped with such a Hilbert symbol? This seems not impossible at all.

I have added a new section in which notation for elements of clans and parent sets and equivalence of notations is developed in the abstract, parallel to and immediately following the concrete description of the infinitary notation. I also added an explicit definition of a referent function on notations to the concrete description of the infinitary notation. I think this might make for a more intelligible approach, since abstract notational considerations are unavoidable in the final construction, but also very hard to follow without a prior understanding of our intentions.

minor further edits 8 pm.

September 29, 2015: Read through the document and corrected various typos and mental slips. No major changes. I am anticipating a more substantial editing pass shortly to try to restructure some of the uglier parts of the argument.

September 13, 2015: My argument for the realizability of the definition of parent sets was intuitively all right but basically silly as presented. Correction installed, but it looks more like earlier versions. I still think
this order of presentation may have advantages, but construction of atoms from notations seems to be unavoidable...

**September 11, 2015:** Minor edits. I need to format definitions and theorems. There are various false starts in the new material to be tidied up and there are probably reconciliation issues with the shared material. Definition and theorem format added, though no doubt it could be improved.

Correction 5:30 pm because I had forgotten the need for strong support machinery in the combinatorics proofs.

**September 10, 2015:** Working on it. I think this is sound and should soon be posted. It needs critical apparatus. I now start to install the critical apparatus from an older version. Now incorporates the common material from other versions.

**September 8, 2015:** Some details need to be fixed. Infinitary notation is defined.
Contents

1 Version remarks 1

2 The Starting Point 9

3 The Hall of Mirrors: the formulation of New Foundations 11

4 Good News and Bad News: well-known results about New Foundations 13

5 Tangled Type Theories and Tangled Webs of Cardinals 15
   5.1 Refinement of logic to facilitate ambiguity results . . . . . . . . 15
   5.2 Tangled Type Theories . . . . . . . . . . . . . . . . . . . . . . 17
   5.3 Tangled Webs of Cardinals . . . . . . . . . . . . . . . . . . . . . 19
   5.4 \( \omega \) - and \( \alpha \)-models from tangled structures . . . . . . . . 21

6 FM techniques in general 23

7 General description of clans 24

8 Simultaneous description of atoms in clans and codes for elements of parent sets 26

9 Free action of allowable permutations and completeness of parent sets 37

10 Combinatorics of clans 42

11 Parent clans; sizes of iterated power sets of clans 46

12 Convergent cardinalities of iterated power sets 47

13 Isomorphism of iterated power sets 48

14 Consistency of NF 49

15 Conclusions to be drawn about NF 50

16 References and Index 52
2 The Starting Point

The beginning of our story is at a point which might be regarded as the end of the story of the Principia Mathematica of Russell and Whitehead (PM, [20]). This is the system called TST by Thomas Forster (for example, in [2], the best current monograph on the subject of NF), the simple typed theory of sets. This is a first-order theory with sorts indexed by the natural numbers $0, 1, 2 \ldots$ and equality and membership as primitive relations. The sorts are traditionally called “types”. Atomic identity sentences $x = y$ are well-formed iff $x$ and $y$ are variables of the same type; atomic membership sentences $x \in y$ are well-formed iff the index of the type of $y$ is the successor of the index of the type of $x$. The axioms are extensionality (objects of each positive type are equal iff they have the same elements) and comprehension (for any formula $\phi(x)$, there is an $A$ such that $(\forall x. x \in A \leftrightarrow \phi(x))$, where $A$, obviously one type higher than $x$ and unique by extensionality, can be denoted by $\{x : \phi(x)\}$.

We regard this as the end of the story of PM (not necessarily an uncontroversial view) because this is a simple and natural system realizing the aims of PM. The type system of PM is considerably more complicated than that of TST for two reasons: the first reason is that Russell and Whitehead did not know how to implement ordered pairs as sets (Norbert Wiener gave the first implementation in [21], 1914), so PM contains types of $n$-ary relations for every $n$ with arbitrarily complex heterogeneous input types; the second reason was that Russell and Whitehead restricted themselves initially to predicative comprehension, then made their system impredicative by adding an axiom of reducibility; TST follows Ramsey ([12]) in using fully impredicative comprehension and no axiom of reducibility. We do know that Russell tried to abandon reducibility in the second edition of PM, but we also know that much of the mathematics in PM does not survive this. TST and the system of PM with reducibility are mutually interpretable.

It is usual to add axioms of Infinity and Choice to TST, but we do not regard these as part of the basic definition. TST + Infinity (+ Choice) has the same mathematical strength as Zermelo set theory with separation restricted to bounded formulas (a system sometimes called Mac Lane set theory). This is weaker than Zermelo set theory, but easily strong enough for almost all mathematics outside of set theory. TST without Infinity is weaker than arithmetic, but this is no part of our story.

There is a very natural interpretation of TST in terms of the familiar
set theory ZFC: let $X_0$ be any set and define $X_{i+1}$ as the power set of $X_i$ for each $i$. Interpret type $i$ variables as ranging over $X_i$. Interpret equality and membership relations between each appropriate pair of types as suitable restrictions of the usual equality and membership relations. Notice that there is no requirement here that the sets interpreting the types be disjoint: we in fact cannot even ask the question in the language of TST as to whether objects of distinct types are equal.

**Definition (initial segments of type theory, natural models):** We define $\text{TST}_n$ as the theory obtained by restricting the language of TST to mention only types with index less than $n$ and having as its axioms exactly the axioms of TST which can be expressed in the restricted language. A natural model of $\text{TST}_n$ is one in which type $i$ is represented by a set $X_i$, with $X_{i+1}$ being of the same size as the power set of $X_i$ for each appropriate $i$, with the membership relation being defined using fixed bijections $f_i$ from $X_{i+1}$ to $\mathcal{P}(X_i)$ for each $i$: $a' \in b^{i+1}$ in the type theory is implemented as $a \in f_i(b)$ in the metatheory. It is an important observation that the first order theory of a natural model of $\text{TST}_n$ is completely determined by the cardinality of the set $X_0$ representing type 0 in the model (it is straightforward to construct an isomorphism between natural models with base sets representing type 0 of the same size).
3 The Hall of Mirrors: the formulation of New Foundations

The next chapter in the story is an observation made by Russell about PM (under the heading of “systematic ambiguity”) and made in a much sharper form by Quine about TST. The system is extremely symmetric, in the sense that there is nothing in the system to distinguish the sorts from one another (one has no information at all about the type of individuals, and uniformly more information about types of higher index, but nothing obvious to show that the type 0 of individuals might not have the same structure as the type 1 of sets of individuals, whence each type \( n \) would have the same structure as type \( n + 1 \)). In TST, this can be stated in a very elegant way. For any sentence \( \phi \), let \( \phi^+ \) be a sentence obtained by raising the type of each variable in \( \phi \) by one (without creating any identifications between variables). Since all variables in \( \phi \) are bound, the exact way that the new variables are chosen does not matter. We observe then that if \( \phi \) is provable, so is \( \phi^+ \), and if we define a mathematical construction using a set abstract \( \{ x : \phi \} \) and our scheme of variable type raising sends \( x \) to \( y \), \( \{ y : \phi^+ \} \) will be a precisely analogous mathematical construction one type higher. This is the “hall of mirrors” aspect of TST: for example, it is natural in TST to define the number three as in effect the set of all sets with three elements, following Frege and also PM, but we then get a new version of the number three in each type \( i + 2 \).

Quine made the stronger suggestion that we should simply identify all the types, to avoid the apparently futile recopying of all theorems and defined objects into ever higher types. This gives the theory which is generally called New Foundations (NF) after the title of the paper \([11]\), 1937, where he made the suggestion. NF is the first order single-sorted theory with equality and membership whose axioms are obtained from those of TST by dropping all distinctions of sort between the variables (without introducing any identifications between variables). That is, the axioms are extensionality (objects with the same elements are the same) and stratified comprehension (\( \{ x : \phi(x) \} \) exists for \( \phi \) a stratified formula), where we say that a formula \( \phi \) is stratified iff there is a function \( \sigma \) from variables to natural numbers with the property that for each atomic subformula \( x = y \) of \( \phi \) we have \( \sigma(x) = \sigma(y) \) and for each atomic subformula \( x \in y \) of \( \phi \) we have \( \sigma(x) + 1 = \sigma(y) \). Notice that the function \( \sigma \) (called a stratification of \( \phi \)) is acting on \( x \) and \( y \) as bits of text, not on their values, so we should perhaps put the variables in quotes (but do
not here do this). Of course, the condition that a formula of the language of set theory is stratified is equivalent to the condition that it is obtainable from a formula of the language of type theory by dropping type distinctions between variables without introducing identifications between variables.

It is a persistent criticism of NF that it is a syntactical trick. Of course, as phrased here, it does look that way. It is possible to give a finite axiomatization of NF, which eliminates the notion of stratification from the explicit definition of the theory (though the very first thing one would do with such a formulation of the theory would be to prove stratified comprehension as a meta-theorem). The standard reference is Hailperin, ([3]).

In terms of the interpretation of TST suggested above, this is a very strange proposal. If type $i + 1$ is represented by exactly the same set as type $i$, it is certainly not represented by (a set the same size as) the power set of the set representing type $i$, which is a larger set by Cantor’s theorem.
4  Good News and Bad News: well-known results about New Foundations

Specker showed in [17], 1962, that NF is equiconsistent with TST + the Ambiguity Scheme which asserts $\phi \leftrightarrow \phi^+$ for each sentence $\phi$, which is not surprising given the motivation of the theory.

Specker also showed, much more surprisingly, in [16], 1953, that Choice is inconsistent with NF, which implies that Infinity is a consequence of NF (as if the universe were finite, every partition, being finite, would have a choice set). [Quine’s argument for Infinity in the original NF paper is fallacious]. This shows that NF is stronger than expected, but it also shows that it is very strange, and caused considerable doubt about this theory.

On the good news side of things, Jensen showed in [10], 1969, that NFU (New Foundations with urelements) is consistent. We outline his approach. The idea is to replace TST with TSTU, in which the axiom of extensionality is weakened to apply only to objects with elements, so that each positive type contains at most one element with each nonempty extension, but may contain many elements with empty extension (urelements or atoms). Note that the individuals of type 0 are not atoms, or at least are not considered as atoms: we simply do not ask the question as to what elements they have.

The results of Specker can be extended to show that NFU (New Foundations with the weaker form of extensionality and full stratified comprehension) is equiconsistent with TSTU + Ambiguity.

We now argue, following Jensen, for the consistency of TSTU + Ambiguity. We work in some familiar set theory (we use nothing like the full power of ZFC, but we may suppose that to be our working theory). Let $X_0$ be a set and define $X_{i+1}$ as the power set of $X_i$ for each $i$. Let each type $i$ variable in the language of TSTU be interpreted as ranging over elements of $X_i$; interpret equality and membership in TSTU as equality and membership suitably restricted. Thus far we have actually interpreted TST. Now we throw in a refinement. Let $s$ be a strictly increasing sequence of natural numbers. An alternative interpretation takes variables of type $i$ as ranging over elements of $X_{s_i}$, takes equality between type $i$ objects as equality suitably restricted, and interprets membership of type $i$ objects in type $i + 1$ objects as holding where the type $i + 1$ object is an element of $X_{s_i+1}$ which has the type $i$ object as an element; note that this interpretation treats all elements of $X_{s_i+1} \setminus X_{s_i+1}$ as urelements (it should be noted that the relation interpreting
membership of type \(i\) objects in type \(i + 1\) objects will not necessarily agree with the relations interpreting membership between other successive types). It is straightforward to establish that this gives an interpretation of TSTU for each increasing sequence \(s\). Now choose a finite set \(\Sigma\) of formulas of the language of TSTU mentioning no types with index higher than \(n - 1\). Define a partition of the \(n\) element subsets \(A\) of the natural numbers determined by the truth values of the sentences in \(\Sigma\) in interpretations of TSTU determined by maps \(s\) which have \(A\) as an initial segment of their range (the truth value of sentences in \(\Sigma\) is entirely determined by the first \(n\) elements of the range of the function \(s\) used). This is a partition of the \(n\) element subsets of \(\mathbb{N}\) into finitely many parts (no more than \(2^{|\Sigma|}\)) and so has an infinite homogeneous set \(H\) which is the range of a strictly increasing map \(h\). In the interpretation of TSTU determined by the map \(h\), we have ambiguity for all formulas in \(\Sigma\). This implies by compactness that full Ambiguity is consistent with TSTU, and by the results of Specker that NFU is consistent.

It is useful to note one could use instead of the sets \(X_i\), sets \(X_\alpha\) indexed by elements of any limit ordinal \(\lambda\) (\(X_0\) an arbitrary transitive set, \(X_{\alpha+1}\) defined as the power set of \(X_\alpha\) for each \(\alpha < \lambda\), and \(X_\mu\) defined as \(\bigcup_{\beta < \mu} X_\beta\) for all limit \(\mu < \lambda\)), whereupon the sequences \(s\) would be strictly increasing sequences of ordinals below \(\lambda\) (still indexed by natural numbers); for example, the stages of the cumulative hierarchy up to any limit level could be used. This is relevant to establishing the consistency of strong extensions of NFU.

It is clear that if Choice holds in our working set theory, Choice will hold in all the approximations to NFU obtained by the method above, and so is consistent with NFU, and that if all \(X_i\)'s are finite, the approximations of NFU obtained by the method above will not satisfy Infinity, and so NFU does not prove Infinity. If \(X_0\) is infinite, Infinity will hold, of course. NFU by itself is weaker than Peano arithmetic, as it happens. NFU + Infinity is a quite usable foundational theory equivalent in strength to TST + Infinity or to Mac Lane set theory.

Other fragments of NF have been shown to be consistent, but the strategy we will follow to show the consistency of full NF follows in its broadest outlines the strategy of Jensen for NFU (though this program turns out to be quite hard to carry out).
5 Tangled Type Theories and Tangled Webs of Cardinals

The first section is a technical note about Specker’s ambiguity result. The second section is part of the intellectual history of this proof. The third section gives the situation we implement in our main construction. The fourth section is relevant to strong extensions of our result discussed in the conclusion.

5.1 Refinement of logic to facilitate ambiguity results

We refine our logic in order to simplify proofs of Specker’s ambiguity results and extensions thereof.

The refinement is to allow the Hilbert symbol $(\epsilon x. \phi)$, both in type theory and in NF or NFU. In typed theories, the type of the term $(\epsilon x. \phi)$ is the same as the type of $x$. The definition of a stratification $\sigma$ of a formula is extended: the function $\sigma$ is assigned values at all terms, both variables and Hilbert symbols, with the value $\sigma('((\epsilon x. \phi))')$ equal to the value $\sigma('x')$. The logical axiom governing this symbol in any context is $(\exists x. \phi) \rightarrow \phi[(\epsilon x. \phi)/x]$. Comprehension axioms are restricted to universal closures of assertions

$$(\exists A.(\forall x. x \in A \leftrightarrow \phi))$$

in which $A$ does not occur in $\phi$, no Hilbert symbol occurs in $\phi$, and $\phi$ is either well-typed or stratified depending on which theory we are considering.

With the language extended in this way, it is straightforward to prove the following Theorem, essentially due to Specker.

**Definition:** Given a map sending variables $x$ of each type $i$ bijectively to variables $x^+$ type $i+1$, we define $\phi^+$ (for any formula $\phi$) as the formula obtained by replacement of each variable $x$ appearing in $\phi$ with the variable $x^+$. It should be evident that $\phi^+$ is well-typed iff $\phi$ is well-typed. We define the scheme of ambiguity as the set of formulas of the form $\phi \leftrightarrow \phi^+$, where $\phi$ is any formula, including those mentioning Hilbert symbols.

**Theorem:** The theory TST(U) + Ambiguity with Hilbert symbols added to its language is consistent iff the theorem NF(U) without Hilbert
symbols added to its language is consistent. Moreover, a sentence \( \phi \) not mentioning Hilbert symbols is a theorem of TST(U) + Ambiguity if and only if the stratified sentence \( \phi^* \) of the language of NF(U) obtained by dropping all distinctions of type between variables in the original sentence \( \phi \) is a theorem of NF(U).

**Proof:** From a model of NF(U) we obtain a model of TST(U) + Ambiguity by using a copy of the model of NF(U) to implement each type, defining the membership and equality of the type theory in the obvious way. The Hilbert symbol can be assigned semantics in NF(U) using a fixed external well-ordering of the model of NF(U): let \( (\epsilon x. \phi) \) denote the first element of the model of NFU in the given well-ordering which witnesses \( \phi \), if there is one, and otherwise the empty set. This induces a semantics for the Hilbert symbol in the model of type theory in the obvious way. Ambiguity clearly holds in a model constructed in this way, for all formulas in the language enriched with Hilbert symbols.

If we have a model of TST(U) + Ambiguity, with Hilbert symbols included in the language of the theory, we first observe that we can obtain a model of the version of TST(U) + Ambiguity (with Hilbert symbols) with its language extended to allow all integer types, satisfying the same sentences not mentioning negative types, by compactness. Now, because of the availability of Hilbert symbols, we can construct a term model of TST(U) + Ambiguity with all integer types and Hilbert symbols, satisfying the same sentences as the initial model with integer types, inhabited by equivalence classes of terms of the language under the relation of being asserted to be equal by the theory of the initial model with integer types. The Ambiguity Scheme then makes it possible to suppress all distinctions of type between elements of the term model, obtaining a model of NF(U) satisfying the type-free form \( \phi^* \) of each sentence \( \phi \) satisfied by the original model of TST(U) + Ambiguity with Hilbert symbols.

**Closing Remark:** We have not quite proved Specker’s original result, because it takes a little work to show that the Ambiguity Scheme for the language enriched with Hilbert symbols is no stronger than the Ambiguity Scheme for the original language. But the form we have proved is adequate for our actual purposes.
5.2 Tangled Type Theories

Tangled type theory TTT is a first order theory with types indexed by the natural numbers, or more generally by ordinals below a fixed limit ordinal $\lambda$. Sentences $x = y$ are well-formed iff the type of $x$ is equal to the type of $y$. Sentences $x \in y$ are well-formed iff the type of $x$ is strictly less than the type of $y$ (it is important to notice that this is not a cumulative type theory). For any formula of the language of the usual TST and any strictly increasing sequence $s$, if $\phi$ is a formula of the language of TST we get a formula $\phi^s$ of the language of TTT by replacing type $i$ variables in $\phi$ with type $s(i)$ variables in $\phi^s$ while preserving identifications and distinctions of variables. The axioms of TTT are the formulas $\phi^s$ for which $\phi$ is an axiom of TST, for all strictly increasing sequences $s$.

If NF is consistent, TTT is consistent. This is evident: take each type to be the model of NF (or if you prefer let the types be disjoint labelled copies of the model of NF) and use the membership and equality relations of the model of NF to induce the membership and equality relations between each appropriate pair of types in the obvious way.

We show that the consistency of TTT (without primitive Hilbert symbols in its language) implies the consistency of NF.

If TTT is consistent it has a model. Enhance the language of TTT with an additional relation $\leq$. $x \leq y$ is well-formed iff $x$ and $y$ are of the same type and the truth value of $x \leq y$ is determined by whether $x$ appears before $y$ in a fixed well-ordering of the model (or is equal to $y$). Of course $\leq$ does not appear in instances of the comprehension axiom, but it will satisfy the axioms that $\leq$ is a total order and that any definable class (including ones defined in terms of $\leq$) which is nonempty has a $\leq$-minimal element.

Now consider any finite collection of formulas $\Sigma$ of the language of TST enhanced to contain the predicate $\leq$ ($x \leq y$ well-formed under the same conditions as $x = y$). Let $n$ exceed the largest type index appearing in any formula in $\Sigma$. Partition the $n$-element subsets $A$ of $\omega$ into $\leq 2^{[2]}$ compartments by considering the truth values of the formulas $\phi^s$ for $\phi \in \Sigma$ and $s$ such that $s|n = A$. An infinite homogeneous set $H$ exists for this partition by the Ramsey theorem and may be chosen to have order type $\omega$ in the natural order on ordinals, if the types are taken from a $\lambda > \omega$. The model of TST obtained in the obvious way from the types of the model of TTT with index in the set $H$ satisfies Ambiguity ($\phi \leftrightarrow \phi^+$) for each sentence in $\Sigma$.

By compactness the full Ambiguity Scheme $\phi \leftrightarrow \phi^+$ is consistent with
TST with the additional predicate $\leq$, the new predicate not appearing in instances of comprehension, with the new axioms that $\leq$ is a total order and every definable class has a $\leq$-minimal element, plus Ambiguity for all formulas. Now observe that this extended theory supports the construction of a Hilbert symbol: $(\theta x. \phi)$ can be defined as the $\leq$-minimal element of its type which satisfies $\phi$ if there is one and otherwise as a default object (use the empty set). One can then construct a term model of the theory built entirely from Hilbert symbols, abandon the distinctions of type between terms of different types, and obtain a model of NF.

The use of the external well-ordering removes the necessity to appeal to the rather complex argument for the equiconsistency of NF with TST+Ambiguity given originally by Specker, involving saturated models.

We can get to stronger theories by indexing the types in TTT with larger ordinals (sequences $s$ will then be strictly increasing sequences of ordinals, still indexed by natural numbers). The development above works (as already noted) if the types of TTT are taken to be ordinals less than a fixed limit ordinal $\lambda$. In fact, it works if the types are indexed by any linearly ordered set with no maximum element.

TTT does not embed nicely into ordinary set theory as TST does: there are no natural models of TTT. The reason for this is that each type in TTT has many types “immediately above” it, which cannot represent real power sets of the types, as for any two of these “power sets” of a given type, one is ostensibly a “power set” of the other!

To restore the possibility of thinking of “successor types” as power sets, one might modify the type scheme as follows: types are labelled not by ordinals $\leq \lambda$ but by nonempty finite subsets of $\lambda$. Type $A$ is a copy of type $\underline{\text{min}}(A)$ of TTT, and its unique “power type” is type $A \setminus \{\underline{\text{min}}(A)\}$. Sequences of types copied from a given sequence of types in TTT will be elementarily equivalent to one another.
Here is a picture of a part of the type hierarchy, on the left as in TTT and on the right as unfolded.

We show in the following section that we can associate these modified types with cardinals, and embed the entire definition into untyped set theory of the familiar sort.

### 5.3 Tangled Webs of Cardinals

We articulate a hypothesis about cardinals in ZFA (ZFC sans choice and with extensionality weakened to allow atoms) whose consistency with ZFA implies the consistency of NF. Note that we can use Scott’s trick ([14]) to define cardinals in this theory; the usual definition using initial ordinals will not work when choice is not assumed. The treatment here is derived from our [5], but the notation is greatly improved.

Let $\lambda$ be an infinite limit ordinal (it could be taken to be $\omega$ for the purposes
of merely proving Con(NF) but we aim for more generality). Fix a natural number constant \( k \).

We define a \textit{tangled web of order} \( \lambda \) as a function \( \tau \) sending nonempty finite subsets of \( \lambda \) to cardinals with two properties:

\begin{itemize}
  \item \textbf{naturality:} If \( A \) has at least two elements, \( 2^{\tau(A)} = \tau(A \setminus \{\min(A)\}) \)
  \item \textbf{elementarity:} For each \( n \), if \( A \) has at least \( n + k \) elements, the first-order theory of a natural model of TST\(_n\) with type 0 having cardinality \( \tau(A) \) is completely determined by the smallest \( n + k \) elements of \( A \). [the use of \( n + k \) here rather than \( n \) is a technicality which arises in the course of actual constructions of tangled webs: the actual value used is \( k = 1 \).]
\end{itemize}

It may not be immediately evident, but the definition of a tangled web of cardinals is precisely motivated by the desire to replicate the consistency proof of Jensen for NF (the material about tangled type theory at the beginning of the section may make the intellectual genealogy of the proof clearer).

We argue that the existence of a tangled web of cardinals implies the consistency of NF. Let \( \Sigma \) be a finite set of sentences of the language of TST mentioning no variable of type \( \geq n \). Partition the finite subsets \( A \) of \( \lambda \) of size \( n + k \) by considering the truth values of the sentences in \( \Sigma \) in natural models of TST with base type of cardinality \( \tau(B) \) where \( B \) has at least \( n + k \) elements and the smallest \( n + k \) elements of \( B \) are exactly the elements of \( A \). By Ramsey’s theorem there is an infinite homogeneous subset \( H \) of \( \lambda \) for this partition. Note that for any subset \( B \) of \( H \) with at least \( n + k \) elements, the theory of a model of TST\(_n\) with base set representing type 0 of cardinality \( \tau(B) \) will assign the same truth values to sentences in \( \Sigma \). It follows that for any finite subset \( B \) of \( H \) with more than \( n + k \) elements, we find that ambiguity holds for all sentences in \( \Sigma \) in a model with base set representing type 0 of size \( \tau(B) \): notice that the power set of this set will be of size \( \tau(B \setminus \{\min(B)\}) \), and the argument of this expression is also a subset of \( H \) with at least \( n + k \) elements. Thus the theory of the natural model of TST\(_n\) with base type \( T \) of size \( \tau(B) \) agrees with the model of TST\(_n\) with base type \( \mathcal{P}(T) \) about the truth value of each sentence in \( \Sigma \), from which it follows that \( \phi \iff \phi^+ \) holds in the model of TST\(_{n+1}\) with base type \( T \) for each \( \phi \) in \( \Sigma \). Thus we find that ambiguity for \( \Sigma \) is consistent with TST\(_m\) for all \( m \geq n \), so with TST. Thus full Ambiguity is consistent with TST by compactness, and NF is consistent by the results of Specker (in their original form without our device of Hilbert symbols, which is very hard to adapt to this context).
All of the above is an adaptation of Jensen’s proof of the consistency of NFU to the proof of the consistency of NF. Of course, one then actually has to produce a tangled web of cardinals (or a model of tangled type theory) to complete the argument.

5.4 \( \omega \)- and \( \alpha \)-models from tangled structures

Jensen continued in his original paper by showing that for any ordinal \( \alpha \) there is an \( \alpha \)-model of NFU. We show that if \( \lambda \) is taken to be large enough, this argument can be reproduced for NF (given a model of TTT or a tangled web with large enough index \( \lambda \)).

We quote the form of the Erdös-Rado theorem that Jensen uses: Let \( \delta \) be an uncountable cardinal number such that \( 2^\beta < \delta \) for \( \beta < \delta \) (i.e., a strong limit cardinal). Then for each pair of cardinals \( \beta, \lambda < \delta \) and for each \( n > 1 \) there exists a \( \gamma < \delta \) such that for any partition \( f : [\gamma]^n \to \lambda \) there is a set \( X \) of size \( \beta \) such that \( f \) is constant on \( [X]^n \) (\( X \) is a homogeneous set for the partition of size \( \beta \)).

We assume the existence of a model of TTT in which each type contains a well-ordering of type \( \alpha \), elements of whose domain are explicitly named with a component representing the type and an index \( < \alpha \). [or a tangled web of cardinals in which sets of each cardinality in the web have this characteristic]. Note that this does not imply directly that the model of NF we eventually obtain, which comes from an application of compactness, will actually contain such an order: many nonstandard elements might be added to it, and it might cease to be externally a well-ordering at all. We need to do extra work.

Let \( \delta \) be a strong limit cardinal with cofinality greater than \( |\alpha| \). Our types in TTT will be indexed by ordinals \( < \delta \); in a tangled web, our cardinals would be indexed by nonempty finite subsets of \( \delta \). Let \( \Sigma_n \) be the collection of all sentences of the language of TST\(_n\) which begin with an existential quantifier restricted to the domain of the special well-ordering of order type \( \alpha \) in some type, in a language which includes special constants in each type, indexed by ordinals \( < \alpha \), representing the elements of the special well-ordering of type \( \alpha \) in that type, indexed of course in order. Let the partition determined by \( \Sigma_n \) make use not of the truth values of the formulas in \( \Sigma_n \), but of the indices \( < \alpha \) of the minimally indexed witnesses to the truth of each formula, or \( \alpha \) if they are false. The Erdös-Rado Theorem in the form cited tells us that we can find homogeneous sets of any desired size less than \( \delta \) for this partition, and moreover (because of the cofinality of \( \delta \)) we can find homogeneous sets
of any desired size with the same witnesses taken from \( \alpha \) for each sentence in \( \Sigma \). This allows us to see that ambiguity of \( \Sigma_n \) is consistent, and moreover consistent with standard values for witnesses to each of the formulas in \( \Sigma \). We can then extend the determination of truth values and witnesses as many times as desired, because if we expand the set of formulas considered to \( \Sigma_{n+1} \) and partition \((n + 1)\)-element sets instead of \(n\)-element sets, we can restrict our attention to a large enough set homogeneous for the previously given partition to ensure homogeneity for the partition determined by the larger set of formulas. After we carry out this process for each \( n \), we obtain a full description of a model of TST + Ambiguity with standard witnesses for each existential quantifier over the domain of a special well-ordering of type \( \alpha \). If we are working in TTT, we can reproduce our Hilbert symbol trick (add a predicate representing a well-ordering of our model of TTT to the language as above) to pass to a model of NF with the same characteristics. If we are working with tangled webs, an appeal to the original form of Specker’s results appears to be unavoidable.
6 FM techniques in general

We work in ZFA (the usual set theory with extensionality weakened to allow a set of atoms). Our assumptions about how large the set of atoms is will be revealed as we go.

We briefly describe the Frankel-Mostowski technique for constructing class models of ZFA, originally developed to prove the independence of Choice from ZFA. Our treatment is adapted from [9].

A parameter of the construction fixed: We fix an uncountable regular cardinal $\kappa$ for the rest of the paper.

Definition (small and large sets): We refer to all sets of size $< \kappa$ as small and all other sets as large.

Any permutation $\pi$ of the set of atoms is extended to all sets by the rule $\pi(A) = \pi " A$.

Let $G$ be a group of permutations of the atoms. Let $\Gamma$ be a nonempty subset of the collection of subgroups of $G$ with the following properties:

1. The subset $\Gamma$ contains all subgroups $J$ of $G$ such that for some $H \in \Gamma$, $H \subseteq J$.

2. The subset $\Gamma$ includes all subgroups $\bigcap C$ of $G$ where $C \subseteq \Gamma$ and $C$ is small.

3. For each $H \in \Gamma$ and each $\pi \in G$, it is also the case that $\pi H \pi^{-1} \in \Gamma$.

4. For each atom $a$, $\text{fix}_G(a) \in \Gamma$, where $\text{fix}_G(a)$ is the set of elements of $G$ which fix $a$.

A nonempty $\Gamma$ satisfying the first three conditions is what is called a $\kappa$-complete normal filter on $G$.

We call a set $A$ $\Gamma$-symmetric iff the group of permutations in $G$ fixing $A$ belongs to $\Gamma$. The major theorem which we use but do not prove here is the assertion that the class of hereditarily $\Gamma$-symmetric objects (including all the atoms) is a class model of ZFA (usually not satisfying Choice). The assumption that the filter is $\kappa$-complete is not needed for the theorem ("finite" usually appears instead of "small"), but it does hold in our construction.
7 General description of clans

We begin by describing general features of the atoms in our interpretation of ZFA. We fix a limit ordinal $\lambda$ for the rest of the paper.

Definitions (clan and atom notation): The atoms are partitioned into sets called clans. The clans are indexed by the finite subsets of $\lambda$ (including the empty set). For this reason we refer to finite subsets of $\lambda$ as clan indices. The clan indexed by $A \in [\lambda]^{<\omega}$ is written clan$[A]$. The clan clan$[A]$ has an associated parent set $\Pi(A)$. $\Pi(A)$ is a large set. There is a bijection $i_A : \Pi(A) \times \kappa \to $ clan$[A]$ (in the ground interpretation of ZFA, which is the only environment we know about just yet). We use the notation $a^A_\alpha$ to abbreviate $i_A(a, \alpha)$. The superscript $A$ may be omitted when understood from context. Note that all atoms are denoted by notations of this form. The clans are disjoint. Distinct clans have distinct parent sets but the parent sets may not be disjoint, so there might be distinct atoms $a^A_\alpha \neq a^B_\alpha$.

Definition (litters): We define litter$^A(a)$ as $\{a^A_\alpha : \alpha < \kappa\}$ for each $a \in \Pi(A)$. We refer to such sets as litters.

Definition (near-litters): A subset of a clan with small symmetric difference from a litter is called a near-litter. The elements of the symmetric difference of a near-litter from the unique litter for which this difference is small are called the anomalous elements for that near-litter. The set of near-litters included in clan$[A]$ is denoted by clan$^\circ[A]$.

Definition (notions of parent): The parent of an atom $a^A_\alpha$ is $a$. The parent of a litter litter$^A(a)$ is $a$. The parent of a near-litter is the parent of the unique litter from which it has small symmetric difference.

Definition (allowable permutations, support sets, FM interpretation): We give a preliminary description of our group $G$ and the filter $\Gamma$. This will not allow us to determine too much about the structure until we give the (insanely elaborate) definition of the parent sets.

We understand permutations of the atoms to be extended to the universe in the usual way.

An allowable permutation is a permutation of the atoms which fixes clans and parent sets and sends each litter$^A(a)$ to a near-litter with small symmetric difference from litter$^A(\pi(a))$. 
A support set is a small set of atoms and near-litters (these may be elements of and subsets of many distinct clans).

An atom or set $x$ has support $S$ iff each allowable permutation $\pi$ such that $\pi(s) = s$ for each $s \in S$ also satisfies $\pi(x) = x$. Clearly an atom has support its own singleton. We define $G_S$ for each support set $S$ as the set of allowable permutations such that $\pi(s) = s$ for each $s \in S$.

The group $G$ will be the set of allowable permutations. The filter $\Gamma$ will be the collection of all subgroups of $G$ which include a $G_S$.

That the group $G$ and the filter $\Gamma$ satisfy the stated conditions is straightforward to establish. The only condition which is worthy of note is the normality condition. If $\pi \in G$ and $H$ is a subgroup of $G$ including $G_S$, one needs to establish that $\pi H \pi^{-1}$ includes a $G_T$: the appropriate $T$ is $\pi(S)$. The reason we use general near-litters as elements of support sets is to make this proof easy: every symmetric object actually has a support consisting only of atoms and litters.
8 Simultaneous description of atoms in clans and codes for elements of parent sets

In this section we will describe a system of infinitary notations (codes), all of which will be pure sets, and an equivalence relation ∼ on infinitary notations which will correspond to having the same referent under the intended semantics, though the equivalence relation will be defined independently of the semantics. Equivalence classes under ∼ of notations of certain types will be correlated with the atoms in clans, which will be all the atoms in our ambient ZFA.

More parameters of the construction: Fix a limit ordinal λ and a large transitive pure set X (κ would do).

Definition (clan index): A clan index is a finite subset of λ. If A is a nonempty clan index, define $A_1$ as $A \setminus \{\min(A)\}$. Define $A_0$ as A and $A_{n+1}$ as $(A_n)_1$ where this is defined. We say that B downward extends A when $A \subseteq B$ and all elements of $B \setminus A$ are less than all elements of A. We say $B << A$ when B downward extends A and is distinct from A.

It is worth noting that any well-founded tree in which there is a top element and all branches are finite could be used as the system of clan indices in the construction which follows, with the operation $A \rightarrow A_1$ and the relation $<<$ given more abstract definitions. But note also that any such tree can be embedded isomorphically into a tree of the specific kind used here.

Introduction to codes: We will define a system of codes, indicate the nature of their intended referents, and define a relation of equivalence ∼ on codes. Codes will have types correlated with the kinds of referents they are intended to have. It will be important to notice that the relation of equivalence we define will turn out to coincide for codes of the same type with the relation of having the same intended referent, but is not in fact defined in terms of the referents of codes (where the relation holds between codes of different types, these codes will have the same referents as well). We use the notation $\delta(c)$ for the referent of a code $c$ of whatever type: this is safe because the various types of codes are disjoint. All codes will be pure sets.
Types of code: For each clan index $A$, there will be a type $\text{clan}_1(A)$ (codes of this type are intended to refer to atoms in the clan indexed by $A$), a type $\text{clan}_2(A)$ (codes of this type are intended to refer to near-litters included in the clan indexed by $A$), and types $C^n(A)$ for each natural number $n \leq |A|$ (codes of this type are intended to refer to elements of the $n$th iterated power set of the clan indexed by $A$: not all elements of these iterated power sets will be referents of codes). Each type as an object is viewed as the class of codes of that type. These classes are in fact sets, but this requires demonstration.

Codes of type $\text{clan}_1(A)$: A code of type $\text{clan}_1(A)$ is of the form $(1,p,A,\alpha)$, where $p \in X$ if $A$ is empty and otherwise $p$ is a code of type $\text{clan}_2(A_1)$ or else a code of a type $C^{[B]-|A|+1}(B)$ for some $B << A$. The intended referent of $(1,p,A,\alpha)$ is an atom denoted by $\delta(p,A,\alpha)$ (note that we are using the index $A$ to indicate which clan the atom is in rather than the parent set of this clan which we do not yet know how to describe): the intention is that $\delta(p,A,\alpha) = \delta(p',A,\alpha')$ iff $\delta(p) = \delta(p')$ and $A = A'$ and $\alpha = \alpha'$. The equivalence $(1,p,A,\alpha) \sim (1,p',A,\alpha)$ holds iff $p \sim p'$, $A = A'$ and $\alpha = \alpha'$. For $p \in X$, $\delta(p) = p$ and $p \sim q$ holds iff $p = q$ for $p,q \in X$. The notation $p$ is called the formal parent of the notation $(1,p,A,\alpha)$.

All codes are pure sets. For each type $\text{clan}_1(A)$ which is a set and each $\sim$-equivalence class $[c]$ for $c \in \text{clan}_1(A)$ we provide an atom $\delta(c)$ correlated with the $\sim$-equivalence class $[c]$ ($\delta(c) = \delta(d)$ iff $c \sim d$). This stipulation is made in this form to ensure that no more than a set of atoms are postulated in any case. It does turn out that all of these types are sets so we get atoms correlated with all equivalence classes of codes in types $\text{clan}_1(A)$. All atoms in our ambient ZFA are of this kind.

It is a useful observation that it is immediately evident that there is a large collection of mutually inequivalent codes in each of these types: consider atoms with iterated formal parents in $X$.

Clans and litters: The class $\delta^{\sim}\text{clan}_1(A)$ is called $\text{clan}(A)$, and such classes of atoms are called clans. These classes will turn out to be sets, but this requires demonstration. The set $\{p^A_\alpha : \alpha < \kappa\}$ (for appropriate $p$) is called $\text{litter}^A(p)$, and such sets are called litters. A set which is a subset of a clan and has small symmetric difference from a litter is called a near-litter. The class of near-litters included in $\text{clan}(A)$ will
be denoted by clan⁰[A]. The object p is called the parent of p^A as an atom, of litter^A(p) as a litter, and of any near-litter with small symmetric difference from litter^A(p) as a near-litter. The notation clan[A] with brackets is used because the parameter is an index of the clan rather than its parent set, which we do not yet know how to describe.

**Codes of type clan^α_*(A):** The set litter^α_*(p) is defined as

\[ \{ (1, p, A, \alpha) : \alpha < \kappa \}, \]

whenever this is a set of codes of type clan_*(A). A code of type clan^α_*(A) is a subset of type clan_*(A) with small symmetric difference from some litter^A(p) (this p is called the formal parent of the code of type clan^α_*(A)), and with no distinct but ~-equivalent members. Two codes M, N of this type are ~-equivalent iff each element of M is ~-equivalent to an element of N and vice versa. The intended referent of a code M of type clan^α_*(A) is the set of atoms δ^M (easily seen to be a near-litter if all codes involved have referents as expected, and it is also easy to see that each near-litter will have codes of this kind).

**General form of codes of types C^n(A):** A code of any of these types will be of the form \((2, f, L)\), which we will write \(f[L]\) to suggest function application, where f will be a “function code” (to be defined below) and L will be an “argument list” (to be defined below). A code of a type \(C^{k+1}(A)\) is called a set code.

**General features of argument lists:** An argument list will be a function with domain a small set of small ordinals and codomain the union of all types clan_*(A) and clan^α_*(A) which further belongs to an argument list type (these types to be defined below). If L is an argument list and \(\alpha \neq \beta\) belong to the domain of L, then \(L(\alpha) \not\sim L(\beta)\), and moreover if \(L(\alpha)\) and \(L(\beta)\) both belong to the same type clan^α_*(B), then no element of \(L(\alpha)\) is ~-equivalent to any element of \(L(\beta)\). (atomic referents of values at distinct ordinals are distinct; near-litter referents of values at distinct ordinals are disjoint).

**General features of function codes:** Each function code has an input type which is an argument list type and an output type which is a
A code \( f[L] \) is well-formed as a code iff \( L \) belongs to the input type of \( f \), and the code \( f[L] \) will belong to the output type of \( f \).

**Argument list types:** An argument list type \( T \) is a class of argument lists (in fact always a set, but this is to be demonstrated) determined by parameters indicated below. Each choice of parameters \( D_T, \tau_T, \rho_T \) meeting the conditions stated determines an argument list type.

**domain:** Each argument list in \( T \) has the same domain \( D_T \) (a small subset of \( \kappa \)). We define the relation \( L \leq M \) on argument lists as holding when \( L \subseteq M \) and all elements of \( \text{dom}(M) \setminus \text{dom}(L) \) are greater than all elements of \( \text{dom}(L) \). We call this the extension order on argument lists, and say that \( M \) extends \( L \) to mean \( L \leq M \).

**absolute types:** There is a function \( \tau_T \) from \( D_T \) to types such that for each \( \beta \in D_T \), and each \( L \in T \), \( L(\beta) \) belongs to a type determined by \( \tau_T(\beta) \): if \( \tau_T(\beta) = (0, B) \) then \( L(\beta) \) is of type \( \text{clan}^*_A(B) \) and if \( \tau_T(\beta) = (1, B) \) then \( L(\beta) \) is of type \( \text{clan}^*_\circ(B) \); all values of \( \tau_T \) are of one of these two forms.

**relative types:** There is a function \( \rho_T \) with domain \( D_T \) which returns additional type information.

If \( \tau_T(\beta) = (0, A) \) then \( \rho_T(\beta) \) is either an ordinal in \( D_T \cap \beta \) such that for any \( L \in T \), \( L(\beta) \in L(\rho_T(\beta)) \in \text{clan}^*_A(A) \) or \( \rho_T(\beta) = \kappa \) and it is not the case for any \( \gamma \in D_T \) with \( L(\gamma) \) of type \( \text{clan}^*_A(A) \) that \( L(\beta) \) is \( \sim \)-equivalent to any element of \( L(\gamma) \). For the next paragraph, define \( g[M] \downarrow \) as the element of the range of \( M \) equivalent to \( g[M] \) if \( g[M] \) is of a type \( C^0(A) \) (see the way that type \( C^0(A) \) codes are defined below to see that this makes sense), and otherwise as \( g[M] \).

If \( \tau_T(\beta) = (1, A) \) then either \( \rho_T(\beta) = (0, g, M) \) where \( M \) is a subset of \( D_T \cap \beta \) and \( g \) is a function code, and for any \( L \in T \), \( L(\beta) \) has small symmetric difference from \( \text{litter}^*_A(g[L[M] \downarrow]) \) (\( g[L[M] \downarrow \) being a well-formed code), or \( A = \emptyset, \rho_T(\beta) = (1, B), B \in X \) and for any \( L \in T \), \( L(\beta) \) has small symmetric difference from \( \text{litter}^*_A(B) \).

**completeness of conditions:** Any \( L \) which meets the conditions stated under the three headings above is an element of \( T \).
names of argument list types: \((3, D_T, \tau_T, \rho_T)\) meeting the conditions above is a name for an argument list type with \(\delta(3, D_T, \tau_T, \rho_T) = T\).

**Codes of type** \(C^0(A)\): A code of type \(C^0(A)\) will be of the shape \((4, \beta, T^*)[L]\), where \(\delta(T^*) = T\) is an argument list type and \(\beta \in D_T\), satisfying the further conditions that \(L \in T\) and \(\tau_T(\beta) = (0, A)\). \((4, \beta, T^*)[L] \sim (4, \beta', T')[L']\) iff \(L(\beta) \sim L'(\beta')\), and \(\delta((4, \beta, T^*)[L]) = \delta(L(\beta))\). We further provide that \((4, \beta, T^*)[L] \sim c\) holds for \(c\) of type \(\text{clan}_n(A)\) iff \(L(\beta) \sim c\). Codes of this type represent the same referents as codes of type \(\text{clan}_n(A)\), but they are not of type \(\text{clan}_n(A)\). Of course we have implicitly declared \((4, \beta, T^*)[L]\) as a function code with input type \(\delta(T^*) = T\) and output type \(\text{clan}_n(A)\); these are the only function codes with this kind of output type.

Codes of type \(C^0(A)\) are never proper components of other codes, but their function code components do so occur.

**Codes of type** \(C^{k+1}(A)\): A code of type \(C^{k+1}(A)\) will be of the shape \((5, U, T^*, k, A)[L]\), where \(L \in \delta(T^*) = T\), an argument list type, and \(U\) is a set of function codes with input types inhabited by argument lists extending elements of \(T\) (in the technical sense defined above: for any \(L\) in \(T\) and \(M\) in the input type of an element of \(U\), \(L \leq M\) and output type \(C^k(A)\), with the further restriction that the additional argument types appearing in the input types of elements of \(U\) (over and above those appearing in \(T\) all be either types \(\text{clan}_n(B)\) with \(B\) downward extending \(A_k\) (not necessarily properly) or types \(\text{clan}^*(B)\) with \(B\) downward extending \(A_{k-1}\) (not necessarily properly) and \(k > 0\). We have implicitly declared \((5, U, T^*, k, A)\) to be a function code with input type \(\delta(T^*)\) and output type \(C^{k+1}(T)\). All function codes with such output types are of this form.

The referent \(\delta((5, U, T^*, k, A)[L])\) is defined as the class

\[\{\delta(g[M]) : g \in U \land L \leq M\},\]

under the conditions that each such \(\delta(g[M])\) is defined and that this class is a set.

A formal element of \((5, U, T^*, k, A)[L]\) is a code \(g[M]\) with \(g \in U\), \(L \in \delta(T^*) = T\), an argument list type, and \(L \leq M\).
The equivalence

\[(5, U, T^*, k, A)[L] \sim (5, U', (T')^*, k', A')[L']\]

is intended to hold iff each formal element of each of the two codes is \(\sim\)-equivalent to some formal element of the other. Note that we do here allow equivalence between codes of different types of the form \(C^{k+1}(A)\), in the special case where the referents are hereditarily finite pure sets; this is an annoying technical point with no import for the proof, which can be handled in two or three different ways.

Our stated intention cannot describe the actual way that equivalence of set codes is to be computed. The difficulty is that arbitrary formal elements of a code \(f[L]\) involve arbitrary extensions \(L'\) such that \(L \leq L'\), and these may involve atoms of unbounded complexity; the recursion defining equivalence ceases to be well-founded.

We control this by first considering how it is that atom notations of unbounded complexity may appear in the range of an extension \(L'\) of \(L\). An atom notation occurring in \(L'\) is novel iff it is an atom notation not appearing in any near-litter earlier in \(L'\) or an atom notation appearing as an element of a near-litter notation in the range of \(L'\) which does not have the same parent as that near-litter notation.

When computing equivalence of \(f[L]\) and \(g[M]\), we consider only what we call \(M\)-bounded formal elements of \(f[L]\), which are those formal elements \(h[L']\) of \(f[L]\) in which each novel atom notation in \(L'\) either has an iterated parent in \(X\) (there is a large supply of such atom notations in each sort of atom notation, and computing equivalence of these presents no difficulties) or is an element of the range of \(L\) or \(M\) or an element of an element of the range of \(L\) or \(M\).

We define \(f[L] \sim g[M]\) as holding when each \(M\)-bounded formal element of \(f[L]\) is equivalent to some \(L\)-bounded formal element of \(g[M]\) and vice versa. We further force \(\sim\) to be an equivalence relation by defining \(\sim\) as the smallest transitive relation meeting this condition.

\(\sim\) is an equivalence relation: This should be evident by induction on the structure of codes.

Demonstration that function codes make up a set: Recall that a formal parent of an element of \(\text{clan}_c(B)\) \((B\) nonempty) will be either an
element of $\text{clan}_e(B_1)$ or an element of a $C^{[D]-|B|+1}(D)$ where $D << B$. Assign to each type $C^n(D)$ a measure of complexity which is the minimum element of $D_n$ (or $\lambda$ if $D_n$ is empty) and assign each $\text{clan}_e(D)$ complexity $\min(D)$. Note that the complexity of the type of each parent of an element of $\text{clan}_e(B)$ is the minimum element of $B_1$. Now observe that the output types of function codes occurring in the argument list types of elements of $U$ but not in $T$ in a function code $(5, U, T^*, k, A)$ will be of complexity at most the minimum element of $A_k$, and so strictly less than the complexity of the code of type $C_{k+1}(A)$ being constructed, which will be the minimum element of $A_{k+1}$ or possibly $\lambda$.

Note that the set theoretical rank of an argument list type name $T^*$ is displaced by no more than a finite constant above the maximum of $\lambda$ and the maximum of the ranks of function codes embedded in it (and that only a small collection of function codes are embedded in any argument list type name). Notice that the set theoretical rank of a function code $(4, \beta, T^*)$ is displaced by no more than a finite constant above the maximum of $\lambda$ and the rank of $T^*$. We claim further that the set theoretical rank of a function code $(5, U, T^*, k, A)$ is displaced upward from the maximum of $\lambda$ and the rank of $T^*$ by no more than a (non-finite) constant $\nu(\beta)$ depending on the complexity $\beta = \min(A_{k+1})$ (or $\lambda$) of its output type $C^{k+1}(A)$. By inductive hypothesis, each function code in $U$, with output type complexity $\beta' = \min(A_k)$, has rank exceeding the rank of its own argument list type by no more than $\nu(\beta')$. The rank of the input type of an element of $U$ may exceed that of $T^*$, because it may include additional function codes of output type complexity bounded by $\beta'$, whose ranks may exceed that of their own input types by no more than $\nu(\beta')$, but in any case the rank of this input type will not exceed that of $T^*$ by more than $\nu(\beta') \cdot \kappa$ (there will be no more than a small collection of occasions for increments). So $\nu(\beta)$ may be taken to exceed the upper bound of ordinals $\nu(\beta') \cdot \kappa$ for $\beta' < \beta$ by a suitable finite constant. If every function code has a bound on its rank computable from its complexity and the ranks of a small collection of function codes of lower rank, it follows that there is a uniform bound on the rank of all function codes, whence function codes make up a set.

The atoms introduced: We introduce our atoms, correlating each equivalence class of atom notations under $\sim$ with an atom, denoting the atom
correlated with \((0, a,\alpha, A)\) as \(\delta(a)^{\alpha}_A\), where for the moment we think of \(\delta(a)\) as representing the equivalence class \([a]\) of \(a\) under \(\sim\).

We can now meaningfully refer to \(\text{clan}[A]\) as the collection of atoms \(\delta(a)^{\alpha}_A\). We know that \(\text{clan}[A]\) is large, because each clan contains a large collection of atoms with iterated parents in \(X\), recognizing the inequivalence of which presents no difficulties.

**Extended definition and observations (substitution extensions):** Let \(\pi_0\) be a small bijection on atoms, sending any atom in its domain to an atom in the same clan (its domain may intersect many clans). We define a map \(\pi_*\) on notations called the *coded substitution extension* of \(\pi_0\), which is intended to induce a permutation \(\pi\) of atoms, the *substitution extension* of \(\pi_0\), such that \(\pi(\delta(x)) = \delta(\pi_*^*(x))\) for all atom notations \(x\) [and indeed for the set notations as well].

For each pair of equivalence classes \([a], [b]\) under \(\sim\) of notations which can be formal parents of notations in \(\text{clan}_*[A]\), choose a bijection \(f_{[a], [b], A}\) from \(\text{litter}(\delta(a)) \setminus \text{dom}(\pi_0)\) to \(\text{litter}(\delta(b)) \setminus \text{dom}(\pi_0)\): the intention is that if \(\pi_*^*(a) = b\) [so that we expect \(\pi(\delta(a)) = \delta(b)\)], \(f_{[a], [b], A}\) will determine the action of \(\pi\) on \(\text{litter}(\delta(a)) \setminus \text{dom}(\pi_0)\).

Note throughout that we expect to be able to define an “inverse map” using the inverse of \(\pi_0\) and the inverses of each of the \(f_{[a], [b], A}\)’s. This can be justified by supposing that we define \(\pi_*^{-1}\) and \(\pi^{-1}\) simultaneously with \(\pi_*\) and \(\pi\).

We now define \(\pi_*(0, a, \alpha, A)\) on the assumption that we know the value of \(\pi_*^*(a)\). We note that \(\pi_*\) is taken to fix elements of \(X\). If \(\delta(a)^{\alpha}_A\) belongs to \(\text{dom}(\pi_0)\), \(\pi_*(0, a, \alpha, A)\) is defined as a fixed representative of the equivalence class of notations associated with \(\pi_0(\delta(a)_A^{\alpha})\) (all notations \(\sim\)-equivalent to \((0, a, \alpha, A)\) are mapped to the same notation in this case).

Otherwise, \(\pi_*(0, a, \alpha, A)\) is defined as the representative of the class of notations associated with \(f_{[a], [\pi_*(a)], A}(\delta(a)^{\alpha}_A)\) which has parent \(\pi_*(a)\).

We now define \(\pi_*(a)\) for notations for formal parents of atoms. We have already noted that \(\pi_*\) fixes elements of \(X\). We further note that where an atom is a parent, we have already indicated how to compute \(\pi_*\).
It remains to compute $\pi_*$ for notations $f[L] \in C^{k+1}(A)$. We define this as $f[L_{\pi}]$, where $L_{\pi}(\alpha)$ is defined as $\pi_*(L(\alpha))$ if $L(\alpha)$ is an atom notation, and $\pi_*^{-1}L(\alpha)$ if $L(\alpha)$ is a near-litter notation. Note that we can define $\pi_*^{-1}$ in terms of $L_{\pi}$ simultaneously.

We observe that by a relatively straightforward recursion, $L_{\pi}$ is an argument list of the same type as $L$, so $f[L_{\pi}]$ is indeed a code. A crucial point is that since all but a small collection of elements of a litter $A^*(a)$ will be mapped by $\pi_*$ to elements of litter $A^*(\pi_*(a))$, the elementwise image of a near-litter notation actually is a near-litter notation, and if the original notation has formal parent $g[M]$ the parent of the image will be $g[M_{\pi}]$ as expected, preserving appropriate relative type conditions. The map $\pi_*$ sends inequivalent atom notations to inequivalent atom notations, so the “injection” conditions on argument lists are preserved.

Further, if $h[L']$ is a formal element of $f[L]$, it follows quite directly that $h[(L')_{\pi}]$ is a formal element of $f[L_{\pi}]$. If $h[L']$ is an $M$-bounded formal element of $f[L]$ and $\pi$ fixes the novel notations appearing in $L'$ which have iterated parents in $X$ and are not involved in $M$, then $h[(L')_{\pi}]$ is an $M_{\pi}$-bounded formal element of $f[L_{\pi}]$. From these considerations it can be shown for any $\pi$ that if $f[L]$ is $\sim$-equivalent to $g[M]$, which is witnessed by a finite chain of relations of having corresponding appropriately bounded formal elements, then $f[L_{\pi}]$ is $\sim$-equivalent to $g[M_{\pi}]$ (and the converse holds as well). Note that this is certainly true if the formal elements of $f[L]$ and $g[M]$ are atom notations; further, we assume as an inductive hypothesis that this holds for the formal elements of the $f[L]$ and $g[M]$ we consider.

Let $h[L']$ be an $M_{\pi}$-bounded formal element of $L_{\pi}$. A suitable $\sigma$ can be constructed so that $h[(L')_{\sigma}]$ is an $M$-bounded formal element of $f[L]$; it will act as $\pi^{-1}$ on atoms involved in $L$ and $M$ and fix the novel atoms with iterated parents in $X$ involved in $L'$. Now $h[(L')_{\sigma}]$ is equivalent to an $L$-bounded formal element $k[M']$ of $g[M]$. Now a map $\tau$ can be constructed inverting $\sigma$ on atoms involved in $L'$ and $M$ and fixing the additional novel atoms in $M'$: $k[(M')_{\tau}]$ will be an $L_{\pi}$-bounded formal element of $g[M_{\pi}]$ equivalent to $h[L]$. This is what is needed to complete the proof that $f[L] \sim g[M] \rightarrow f[L_{\pi}] \sim g[M_{\pi}]$ in all cases (and consideration of $\pi^{-1}$ shows the converse).
This tells us that $\pi_*$ induces a map $\pi$ on atoms by $\pi(\delta(x)) = \delta(\pi_*(x))$ for each atom notation $x$.

We now argue that if $f[L] \sim g[M]$ then in fact every formal element of $f[L]$ is equivalent to some element of $g[M]$ and vice versa. This is true when the formal elements of $f[L]$ and $g[M]$ are atoms and the functions $f$ and $g$ are in effect projection operators. We assume as an inductive hypothesis that we already know this for the formal elements of the $f[L]$ and $g[M]$ under consideration (which belong to a sort $C^k(A)$ of lower index). We can suppose that $f[L]$ has the same $M$-bounded formal elements up to equivalence as $g[M]$ has $L$-bounded formal elements [if it works for a single step it will work for any finite number of steps].

Let $h[L']$ be a formal element of $f[L]$. Define a suitable map $\sigma$ sending the novel atom notations appearing in $L'$ which are not involved suitably in the list $M$ to atoms with iterated parent in $X$ and fixing all atoms appearing in $L$ and $M$. Then $h[(L')_\sigma]$ will be an $M$-bounded formal element of $f[L]$, so equivalent to an $L$-bounded formal element $k[M']$ of $g[M]$, and $k[(M')_{\sigma^{-1}}]$ will be a formal element of $g[M]$ equivalent to $h[L']$. That our procedure for constructing substitution extensions is invertible should be clear: invert the map $\pi_0$ and invert each of the maps $f|_a, b, A$ to obtain the reverse map.

We also need the converse result: if every formal element of $f[L]$ is equivalent to a formal element of $g[M]$ and vice versa, then every $M$-bounded element of $f[L]$ is equivalent to an $L$-bounded element of $g[M]$ and vice versa. Suppose that $h[L']$ is an $M$-bounded formal element of $f[L]$. $h[L']$ is equivalent to some formal element $k[M'']$ of $g[M]$. We can then apply a map $\sigma$ taking the referent of each novel atom notation in $k[M'']$ which is not involved in the argument list $L'$ to an atom with iterated parent in $X$ and fixing each atom appearing in the range or as an element of a range element in $L'$ (not just $L$) or $M$: $k[(M'')_\sigma]$ will be equivalent to $h[L']$ (and for that matter also to $k[M'']$) and also $L$-bounded. This works because the novel material in $L'$ is already restricted to be taken from material in $M$ or with iterated parent in $X$.

**Extension of the denotation map to set codes:** Now we can justify the evaluation of $\delta(f[L])$ as $\{\delta(g[M]) : g \in \pi_2(f) \wedge L \leq M\}$, since images under this recursively defined map of codes will be equal precisely if the codes are equivalent.
We have defined our map $\pi$ on atoms and extend it to sets in the usual way. We argue that $\pi(\delta(f[L])) = \delta(f[L_\pi]) = \delta(\pi_*(f[L]))$. This is evident for the case where $f[L]$ is an atom. In the set case,

$$\pi(\delta(f[L])) = \pi^\omega(\{\delta(g[M]) : g \in \pi_2(f) \land L \leq M\}) = \{\delta(g[M]) : g \in \pi_2(f) \land L \pi \leq M\} = \delta(f[L_\pi])$$

[ind hyp]

Notational conventions for the rest of the paper: We will use the notation $f_{U,T}(L)$ to abbreviate $\delta((5,U,T^*,k,A)[L])$, and the notation $\pi_\beta(L)$ to abbreviate $\delta((4,\beta,T^*)[L])$ [these are implicit function definitions]. The objects $f_{U,T}$ and $\pi_\beta$ we call “coding functions” (these notations are actually polymorphic as not all type information is included in the notation: more complete notation might be $\pi_\beta,T, f_{U,T,k,A}$, but the additional information is generally in the context).

The parent set of $\text{clan}[A]$, which we will denote by $\Pi(A)$ is actually $\text{clan}[A_1] \cup \bigcup_{B \subset A} \delta^{\omega \mathbb{C}[B]-|A|+1}(A)$. We denote $\delta^{\omega \mathbb{C}[A]}(A) \subseteq \mathcal{P}^n(\text{clan}[A])$ by $\mathcal{P}_n^\omega(\text{clan}[A]):$ this is the set of all codable elements of the given iterated power set of the given clan. Where the parent set $P$ of a clan is given abstractly, we introduce the notation $\text{clan}(P)$ for the clan: note the use of parentheses instead of brackets.

Remark on the odd character of this scheme: It should be noted how very odd the interlocking structure is of clans and parent sets introduced here. Notice that $\Pi(A)$ includes $\text{clan}(\Pi(A_1))$ and so includes a set as large as $\Pi(A_1)$ and so on for each $\Pi(A_n)$. But on the other hand $\Pi(A_n)$ includes $\mathcal{P}_{n+1}^\omega(\text{clan}(\Pi(A))$ (if $n > 0$ and $A_n$ is nonempty) which certainly is at least as large as $\Pi(A)$. We will see below, however, that $\mathcal{P}_{n+1}^\omega(\text{clan}(\Pi(A))$ is the full iterated power set of its clan argument from the standpoint of the FM interpretation we will define.
9 Free action of allowable permutations and completeness of parent sets

Theorem (Substitution Property): For any permutation $\pi^*$ of a small set of atoms, sending each atom in its domain to an atom in the same clan, there is an allowable permutation extending $\pi^*$, all of whose exceptions are in the domain of $\pi^*$.

Proof: This has already been shown. The substitution extension of $\pi^*$ has the desired properties.

Permutation Lemma: If $L$ belongs to argument list type $T$ and $\pi$ is an allowable permutation (so $L_{\pi}$ is certainly defined), $L_{\pi} \in T$ as well; further, for each coding function $f$, $\pi(f(L)) = f(L_{\pi})$.

Proof: The two parts are proved by mutual structural induction.

Certainly $L_{\pi}(\beta)$ will denote an element of the type indicated by $\tau_T(\beta)$ for each $\beta$, since an element of a clan or near-litter included in a clan which happens to be in the range of $L$ will be sent to an element of the same clan or a near-litter included in the same clan, because $L_{\pi}$ is defined. It remains to check the relative type conditions.

For each $\alpha, \beta$, it is evident that $L(\beta)$ denotes an atom belonging to an near-litter denoted by $L(\alpha)$ iff $L_{\pi}(\beta)$ denotes an atom belonging to an near-litter denoted by $L_{\pi}(\alpha)$, and this is enough for the relative type conditions for atoms to be preserved. Further, if $g$ is a coding function and $M$ is a subset of $D_T \cap \beta$, and $L(\beta)$ denotes a near-litter with parent $g(L[M])$, we can suppose as an inductive hypothesis that $\pi(g(L[M])) = g((L[M])_{\pi}) = g(L_{\pi}[M])$ [the coding function $g$ being simpler than the argument list type $T$, since it appears as a component of its specification], and this is the parent of $L_{\pi}(\beta)$, confirming that the relative type conditions for near-litters hold, and the type of $L_{\pi}$ is the same as the type of $L$.

That $\pi(\pi_{\beta}(L)) = \pi_{\beta}(L_{\pi})$ is obvious, as the coding functions with atomic output are in effect projection functions.

Now consider

$$\pi(f_{A,T}(L)) = \{\pi(g(M)) : g \in A \land L \leq M\}$$
(here we apply the inductive hypothesis; $g$ has output type an iterated power set of a clan with smaller index)

$$= \{g(M) : g \in A \land L \leq M \} = \{g(M) : g \in A \land L \pi \leq M \} = f_{A,T}(L \pi).$$

This completes the argument.

**Corollary on supports:** Any coded object $f(L)$ has a support consisting of the referents of range elements of $L$.

**Redundancy Lemma:** If $f[L]$ is a code of type $C^n(A)$, $f[L] \sim f'[L']$, where $L' \subseteq L$ is determined by choosing a small ordinal $\alpha$ and removing all elements from $L$ whose first projection is $\geq \alpha$ and whose second projection is of a type $\text{clan}_\ast(B)$ with $B$ not downward extending $A_n$ or (if $n > 0$) of a type $\text{clan}^\circ(B)$ with $B$ not downward extending $A_n - 1$. The function code $f'$ is obtained from $f$ by removing appropriate elements from all component argument list types (including carrying out the same operation on function codes appearing as components of its argument list type).

**Proof of Redundancy Lemma:** Observe first that in any case $L'$ defined as above is an argument list. An atom code $L'(\beta)$ with $\beta \geq \alpha$ belonging to an $L(\gamma)$ in $\text{clan}_\ast^\circ(A_{n-1})$ with $\gamma \geq \alpha$ would have to have its relative type reset in the argument list type $T'$ of $L'$ to indicate that it belonged to no near-litter code in $L'$, since the near-litter to which it actually belongs is omitted.

A near-litter code $L'(\beta)$ with $\beta > \alpha$ which belongs to a $\text{clan}^\circ(B)$ with $B$ downward extending $A_{n-1}$ either has an atom code formal parent which will remain in the domain of $L'$ because it is in $\text{clan}_\ast^\circ(B_1)$ and $B_1$ downward extends $A_n$, or has formal parent of a type $C^{\lceil C \rceil - \lceil B \rceil + 1}$ for $C$ downward extending $B$ expressed as a function $g[L[M]]$ of earlier arguments in $L$ which can be converted to $g'[L'[M']]$ by a suitable inductive hypothesis, omitting arguments from $L[M]$ which are of a type $\text{clan}_\ast(D)$ with $D$ not downward extending $C^{\lceil C \rceil - \lceil B \rceil + 1} = B_1$, which includes the case of $D$ not downward extending $A_n$, or (if $n > 0$) of a type $\text{clan}^\circ(D)$ with $D$ not downward extending $C^{\lceil C \rceil - \lceil B \rceil} = B$, which includes the case of $D$ not downward extending $A_{n-1}$, without affecting the referent of $g[M]$: in either case the relative type information in $T$ is readily transformed to appropriate information for $T'$.
If \( n = 0 \) the main claim is evident: \((4, \beta, T^*)(L) \sim (4, \beta, (T')^*)(L')\) where \( L' \) contains all and only the elements of \( L \) with second component in the type \( \text{clan}_n(B) \) to which \( L(\beta) \) belongs and \( T' \) is the argument list type of \( L' \). The function codes differ precisely in having different component argument list types in the expected way.

Assume that the result holds for \( n \leq k \). \((5, U, T^*, k, A)[L] \) denotes \( \{ \delta(g[M]) : g \in U \land L \leq M \} \). Note that the restriction on argument types for function codes in \( U \) ensures that \( M \setminus L = M' \setminus L \); the new arguments in argument lists \( M \) here satisfy the range restriction we impose on \( M' \) already. By inductive hypothesis, we know that \( \delta(g[M]) = \delta(g'(M')) \) for each specific \( g \) and \( M \). Any \( \delta(g[M]) \) is equivalent to \( \delta(g'[M']) \) by inductive hypothesis, and certainly \( L' \leq M' \).

What does still need to be shown is that any \( \delta(g'[M]) \) with \( L' \leq M \) is in fact a \( \delta(g'[M']) \). But this follows from the restriction on types of arguments in \( M \setminus L \); no elements of \( M \setminus L \) will be removed. So \( \{ \delta(g[M]) : g \in U \land L \leq M \} = \{ \delta(g'[M']) : g \in U \land L \leq M \} \) (by ind hyp elementwise) = \( \{ \delta(g'[M]) : g \in U \land L' \leq M \} = (5, U', (T')^*, k, A)[L'] \), where \( U' \) is the set of all \( g' \) for \( g \in U \).

**Theorem:** All elements of iterated power sets \( \mathcal{P}^n(\text{clan}[A]) \) of clans which have support are actually referents of infinitary notations, and can in fact be expressed as referents of infinitary notations \( g[M] \) with argument list extending any fixed argument list \( L \), with each element of the range of \( M \setminus L \) belonging to a type \( \text{clan}_n(B) \) with \( B \) downward extending \( A_n \) (not necessarily properly) or (if \( n > 0 \)) to a type \( \text{clan}_n^*(B) \) with \( B \) downward extending \( A_{n-1} \).

**Proof:** Fix an argument list \( L \).

This is clearly true for elements of clans, for which all coding functions are in effect projection functions. In this case \( n = 0 \). One can create an argument list extending \( L \) with a single additional argument denoting an atom in \( \text{clan}[A] \) unless a notation for the atom which is the value is actually in the range of \( L \), and the single additional argument if present is in \( \text{clan}_n(A) \) and \( A \) downward extends \( A_n = A_0 = A \).

Suppose the result to be true for all \( k \)th power sets of clans. We choose any element \( E \) of the \( k + 1 \)-th power set of a clan which has a support \( S \). Choose a code \( g[M] \) for each element of \( E \), with the argument list
$M$ extending an argument list $L'$ which extends the fixed argument list $L$ and whose range includes notations for each element of the given support $S$ of $E$ (subject to a restriction discussed in the last paragraph) and having the property that each element of the range of $M \setminus L'$ belonging to a type $\text{clan}_k[B]$ with $B$ downward extending $A_k$ (not necessarily properly) or (if $n > 0$) to a type $\text{clan}_n[B]$ with $B$ downward extending $A_{k-1}$. We can do this by inductive hypothesis.

The set $f_{A,T}(L')$, where $A$ is the set of codes $g$ used in codes for elements of $E$ and $T$ is the type of $L'$, certainly contains every element of the original set $E$. Now any element of $f_{A,T}(L')$ is of the form $g(M')$ where there is $g(M) \in E$, and we can define a small map $\pi_0$ implementing a substitution, fixing the referent of each element of the range of $L'$, which “sends $M$ to $M'$” [map atoms with referents in $M$ to atoms with referents in corresponding positions in $M'$; further extensions to the small map are needed to handle anomalous elements for near-litters referenced in corresponding positions in $M$ and $M'$.] This substitution map can be extended to a small bijection respecting clans (with additional elements of the domain of the bijection belonging to near-litters referenced in the range of $M$ mapped into near-litters referenced in corresponding positions in $M'$ and additional elements belonging to near-litters referenced in the range of $M'$ having preimages in the near-litters referenced in corresponding positions in the range of $M$), and then to an allowable permutation fixing each referent of an element of the range of $L'$, mapping $g(M) \in E$ to $g(M')$ (having no exceptions other than elements of the domain of the small bijection causes it to treat near-litters referenced in the range of $M$ correctly). But then $g(M') \in E$ as well, since the allowable permutation fixes all referents of elements of the range of $L'$, a support of $E$, by construction, and it follows that $f_{A,T}(L') = E$.

To enforce the restriction on types of arguments in the range of $L'$ we apply the Redundancy Lemma with the ordinal $\alpha$ in the proof of the Lemma being set to the first ordinal dominating the domain of $L$.

**Definition of our FM interpretation:** The group $G$ defining our FM permutation is the group of allowable permutations. For any support set $S$, $G_S$ is defined as the set of allowable permutations which fix each element of $S$. The filter $\Gamma$ is defined as the set of subgroups of $G$ which
include a $G_S$.

That $\Gamma$ is a normal filter is straightforward to establish. The normality condition is the only one which requires any work: if $H$ contains $G_S$, it is straightforward to show that for any $\pi \in G$, $\pi H \pi^{-1}$ includes $G_{\pi(S)}$.

**Observation and convention on iterated power set notation:** We have shown that the subset of $\mathcal{P}^{n+1}(\text{clan}(P))$ which is included in a parent set, if it has non-pure members, is the full iterated power set of the clan in the sense of the FM interpretation. Subsequently in this paper, we will use the notation $\mathcal{P}^{n+1}(\text{clan}(P))$ to denote the iterated power set in the sense of the FM interpretation, unless we specifically say otherwise. We do have the notation $\mathcal{P}^{n+1}(\text{clan}(P))$ available for this set by the previous theorem if we need to draw a distinction.

**Observation about internal vs. external cardinalities:** We argued above that if $A$ and $A_n$ are both nonempty, then $\Pi(A)$ and $\Pi(A_n)$ (and the associated clans) are of the same cardinality in the ground interpretation. But in the FM interpretation $\Pi(A_n)$ includes $\mathcal{P}^{n+1}(\text{clan}(\Pi(A)))$, and so is a much larger set in terms of the FM interpretation. The moral here is that the power sets of the FM interpretation are quite impoverished.
10 Combinatorics of clans

We now discuss the combinatorics of a fixed clan \( \text{clan}(P) \) in the FM model.

The domain of the FM interpretation contains its small subsets: We first observe that every small set of elements of the FM model in an iterated power set of the clan is an element of the FM model. Take the union of supports for each element of the small set to get a support for the small set.

Supports may be taken to contain only litters proper: We note that any support \( S \) can be refined to one in which all near-litter elements of \( S \) are litters (replace near-litter elements with the litters with small symmetric difference from them; add the elements of the symmetric differences (the anomalous elements for the near-litters) to the support). We do this everywhere below; allowing near-litters in supports is important because it simplifies the proof of normality, but here we prefer to eliminate them.

Strong support: definition and lemma A strong support for a set \( A \) is a support obtained from a code \( f[L] \) for \( A \) by adding to the set of referents of elements of the range of \( L \) all anomalous elements for near-litter referents of elements of the range of \( L \) (which allows us to replace near-litters with litters with the same parent). An allowable permutation \( \pi \) which fixes all atomic elements of a strong support of \( A \) and has no exceptions in litter elements of \( S \) other than possibly fixed points of \( \pi \) will fix \( A \): if it moves \( A \), it moves some element of the strong support, which cannot be an atom, so it must move a litter, and it must move a first one in the order on the original argument list, whose parent it must fix as all elements of a support thereof (references to which appear earlier in the list) are fixed, and it can only move a litter whose parent it fixes (and all of whose anomalous elements it fixes) by having an exception in the litter, which will not be a fixed point of \( \pi \), again because the parent is fixed.

Litters are sets of the FM interpretation: Each litter is a set of the FM interpretation with support consisting of its own singleton.

Definition (\( \kappa \)-amorphous set): We call a set \( \kappa \)-amorphous iff all its subsets are small or co-small.
Litters are $\kappa$-amorphous in the FM interpretation: We show that the litters in $\text{clan}(P)$ are $\kappa$-amorphous sets in the FM interpretation. Suppose to the contrary that $L$ is a litter, $A \subseteq L$ is large and $L \setminus A$ is large, and $A$ has strong support $S$. Let $a \in A$ and $b \in L \setminus A$ with neither $a$ nor $b$ belonging to $S$ nor to a fixed strong support of $L$. There will be an allowable permutation $\pi$ extending the small map which interchanges $a$ and $b$ and fixes each atomic element of a strong support of $L$ and each atomic element of $S$ and has no exceptions belonging to any litter in the strong support of $L$ or in $S$ (because in fact it has no exceptions which it moves), and so fixes all elements of $S$. But this is impossible, because such a map would move $A$ while fixing every element of its given support.

Theorem: The subsets of each $\text{clan}(P)$ in the FM interpretation are exactly the sets with small symmetric difference from the unions of small or co-small collections of the litters in the clan.

Proof: We say that a set $A$ cuts a set $B$ iff $B \cap A$ and $B \setminus A$ are both nonempty.

We claim that for any subset $A$ of $\text{clan}(P)$ in the FM interpretation, $A$ can cut only a small collection of litters. Suppose otherwise, that $A$ cuts a large collection of litters and has a strong support $S$. Choose $a$ and $b$ in the same litter $L$, one belonging to $A$ and the other not belonging to $A$, with neither belonging to $S$. There will be a permutation $\pi$ extending the small map exchanging $a$ and $b$ and fixing each atomic element of a strong support of $L$ and each atomic element of $S$, and having no exceptions other than elements of $S$ and of the strong support of $L$; but this is impossible, as this map must also fix $A$, as it fixes all atomic elements of its given support and has no exceptions which it moves belonging to litters in the given support, since the map has no exceptions which it moves.

We claim that for any subset $A$ of $\text{clan}(P)$ in the FM interpretation, it cannot be the case that a large number of litters meet $A$ and a large number of litters do not meet $A$. Suppose that $A$ has strong support $S$ and a large number of litters meet $A$ and a large number of litters do not meet $A$. Choose atoms $a$ and $b$, one belonging to a litter $L$ meeting $A$ and one belonging to a litter $M$ not meeting $A$, chosen so that none of $a, b, L, M$ belong to $S$. There will be a permutation extending the
small map extending $a$ and $b$ and fixing all atomic elements of $S$ and of strong supports of $L, M$ and having no exceptions which it moves belonging to any litter element of $S$ (since $a, b$ are the only exceptions which it moves) and so fixing all litter elements of $S$. This permutation moves $A$ but it cannot do so because it fixes all elements of its given support.

Now there are two kinds of subset of $\text{clan}(P)$ in the FM interpretation:

1. sets which meet a small collection of litters in $P$ and so have small symmetric difference from the union of the litters in this small collection whose intersection with $A$ is co-small,

2. and sets which meet a large collection of litters in $P$ (and so fail to meet only the litters in a small collection), but cut only a small collection of them, which thus have small symmetric difference from the union of the co-small collection of litters which have co-small intersection with $A$.

So every such set $A$ has small symmetric difference from a small or co-small union of litters. Further, it is evident that any small union of litters is a set of the FM interpretation, so any small or co-small union of litters is a set of the FM interpretation, so any set with symmetric difference from a small or co-small union of litters is a set of the FM interpretation.

**Observation:** Note that this tells us that $\text{clan}(P)$ has the same power set in the FM interpretation quite independently of what power set $P$ has in the FM interpretation; sets $\text{clan}(P)$ with parent sets of the same cardinality in terms of the ground interpretation will have power sets in the FM interpretation which are isomorphic in terms of the ground interpretation. This will not be true for further iterated power sets in the FM interpretation.

**Theorem (double power set lemma):** The set $\mathcal{P}^2(\text{cal}(P))$ contains a set the same size as $\mathcal{P}(P)$ according to the FM interpretation.

**Proof:** We argue that a subset of $\text{clan}(P)$ is the same size as a litter $L \subseteq \text{clan}(P)$ in the FM interpretation iff it has small symmetric difference from $L$. First, it is clear that a set which has small symmetric difference from $L$ is the same size as $L$ in the FM interpretation, as a
bijection witnessing this fact can be obtained which has small symmetric difference from the identity map and so certainly is a set of the FM interpretation. Now suppose that there is a bijection \( f \) from \( L \) to a set \( A \) where \( L \Delta A \) is large, with strong support \( S \). Let \( U \) be one of \( L \setminus A \) and \( A \setminus L \) which happens to be large. Let \( g \) be the one of \( f \) and \( f^{-1} \) which is defined on \( U \). We choose two elements \( a, b \) from \( U \) in such a way that none of \( a, b, g(a), g(b) \) belong to \( S \); we choose these so that the elements of each of the pairs \( a, b \) and \( g(a), g(b) \) each belong to the same litter (one of the pairs both belong to \( L \); some litter must have a large intersection with the large set \( A \setminus L \)). There is a permutation \( \pi \) which swaps \( a, b \) and fixes \( g(a) \) and \( g(b) \), and further fixes each atomic element of \( S \) and has no exceptions which it moves in near-litter elements of \( S \) (since it has no exceptions which it moves) so fixes all elements of \( S \). The resulting map will move \( f \), but this is impossible because it fixes all elements of the given support of \( f \).

Thus a reasonable nonce definition for \( \lvert \text{litter}(a) \rvert \) is as the collection of near-litters with small symmetric difference from the litter, as this is exactly the collection of subsets of the same clan with this cardinality. Now the map \( (a \in P \mapsto \lvert \text{litter}(a) \rvert) \) has empty support in the allowable permutations, so is a set of the FM interpretation. Moreover \( (B \subseteq P \mapsto \bigcup_{a \in B} \lvert \text{litter}(a) \rvert) \) is a set of the FM interpretation for the same reason, and is a bijection from \( P(P) \) into \( \mathcal{P}^2(\text{clan}(P)) \) in the sense of the FM interpretation. So the abundance of subsets of \( P \) in the FM interpretation has no effect on the extent of \( \mathcal{P}(\text{clan}(P)) \) in the FM interpretation, but has a strong effect on the extent of \( \mathcal{P}^2(\text{clan}(P)) \).

This is a key idea of the proof: the ability to construct sets which are externally isomorphic (in the sense of the ground interpretation) and have quite different power sets (in the sense of the FM interpretation) is essential for getting an argument for Con(NF) analogous to Jensen’s argument for Con(NFU) to work. Further applications of this machinery allow us to do the same thing with models of initial segments of simple type theory with arbitrarily many types, getting externally isomorphic natural models of \( \text{TST}_n \) in the FM interpretation whose top types have non-isomorphic power sets in the FM interpretation, so the natural models of \( \text{TST}_{n+1} \) extending them are not isomorphic.
11 Parent clans; sizes of iterated power sets of clans

Note that $\mathcal{P}(D)$ in this section denotes the power set of $D$ in the FM interpretation.

Where $P$ is a parent set $\Pi(A)$, we use the notation $P_1$ to represent $\Pi(A_1)$.

We know from above that $\mathcal{P}^2(\text{clan}(P))$ contains a subset the same size as $\mathcal{P}(P)$ in the sense of the FM interpretation. This means that it further contains a set the same size as $\mathcal{P}(\text{clan}(P))$. Thus $\mathcal{P}^3(\text{clan}(P))$ contains a set the same size as $\mathcal{P}^2(\text{clan}(P_1))$ which contains a set the same size as $\mathcal{P}(P_1)$. Now we have an argument by induction. Suppose that we have shown that $\mathcal{P}^{n+2}(\text{clan}(P))$ contains a set the same size as $\mathcal{P}(P_n)$. It follows that $\mathcal{P}^{n+3}(\text{clan}(P))$ contains a set the same size as $\mathcal{P}^2(P_n)$ which contains a set the same size as $\mathcal{P}^2(\text{clan}(P_{n+1}))$ which contains a set the same size as $\mathcal{P}(P_{n+1})$. This completes a proof by induction of the following

**Theorem:** $\mathcal{P}^{n+2}(\text{clan}(P))$ contains a set the same size as $\mathcal{P}(P_n)$, in the sense of the FM interpretation, for every $n$ for which $P_n$ is defined.
12 Convergent cardinalities of iterated power sets

We show that if $A_i = B_j$, $|\mathcal{P}^{i+2}(\text{clan}[A])| = |\mathcal{P}^{j+2}(\text{clan}[B])|$ in the FM interpretation. We know from results above that $\mathcal{P}^{i+2}(\text{clan}[A])$ contains a set the same size as $\mathcal{P}(\Pi(A_i))$ and $\mathcal{P}^{j+2}(\text{clan}[B])$ contains a set the same size as $\mathcal{P}(\Pi(B_j))$ (cardinalities here being understood in the sense of the FM interpretation). But $A << A_i$ so $\Pi(A_i) = \Pi(B_j)$ contains $\mathcal{P}^{i+2}(\text{clan}[A]) = \mathcal{P}^{i+1}(\text{clan}[A])$ and similarly contains $\mathcal{P}^{j+1}(\text{clan}[B])$, from which it follows that $\mathcal{P}^{i+2}(\text{clan}[A])$ contains a set the same size as $\mathcal{P}(\Pi(A_i))$ which contains a set the same size as $\mathcal{P}(\mathcal{P}^{j+1}(\text{clan}[B]))$ which is $\mathcal{P}^{j+2}(\text{clan}[B])$, and vice versa, so $|\mathcal{P}^{i+2}(\text{clan}[A])| = |\mathcal{P}^{j+2}(\text{clan}[B])|$ in the FM interpretation by Schröder-Bernstein.
13 Isomorphism of iterated power sets

Every item in the iterated power set $P^n(\text{clan}[A])$ has a code with argument list containing no code for an atom in a $\text{clan}[B]$ with $B$ not downward extending $A_n$ nor any code for a near-litter in a $\text{clan}^0[B]$ with $B$ not downward extending $A_{n-1}$, by the Redundancy Lemma. Call such codes “formally restricted”. Now if $B \setminus B_n = A \setminus A_n$, with $A$ and $B$ nonempty with the same maximum element, then choose any external bijection from $\text{clan}[A_n]$ to $\text{clan}[B_n]$ (we have shown above that all clans with indices with the same maximum element are externally the same size): this bijection can be naturally extended (its action on parents dictating its action on (formal representations of) atoms, its action on (formal representations of) elements dictating its action on (formal representations of) sets) to a map converting any formally restricted code for an element of $P^n(\text{clan}[A])$ to a formally restricted code for an element of $P^n(\text{clan}[B])$ bijectively (with adjustments of type indices by replacing each type index $C$ appearing with $(C \setminus A_n) \cup B_n$; this exactly preserves structure). To show this without the assumption that $A$ and $B$ have the same maximum element, we need the assumption that $\Pi(\emptyset) = X$ is as large as the other $\Pi(A)$’s, so that all parent sets are the same size [which can indeed be arranged, but we choose not to discuss this], but this is not strictly needed as the case where all clan indices considered have the same maximum element is sufficient for the proof (as we will see below). This bijection preserves structure, because facts of membership and equality in the power sets are computable from the infinitary notation considered abstractly, and the features of the infinitary notation which support this computation are preserved by the transformation in question. Note that this transformation acts on all lower indexed iterated power sets of $\text{clan}[A]$ and $\text{clan}[B]$ as well, but there is no reason to expect it even to send sets to sets on $P^{n+1}(\text{clan}[A])$: this transformation is not a function of the FM interpretation.

This shows us that the first order theory of the natural model of the first $n + 1$ types in the FM interpretation whose base type is $\text{clan}[A]$ and whose top type $P^n(\text{clan}[A])$ depends only on $A \setminus A_n$ and the maximum element of $A$ (and in fact does not depend on the maximum element of $A$, though we are not concerned to show this).
14 Consistency of NF

Suppose that $\lambda > \omega$.

The final step is to observe that for any fixed limit ordinal $\alpha < \lambda$, $\tau(A) = |\mathcal{P}^2(\text{clan}(A \cup \{\alpha\}))|$ for nonempty $A$ dominated by $\alpha$ defines a tangled web of cardinals in the FM interpretation.

The cardinality of the power set of $\mathcal{P}^2(\text{clan}[A \cup \{\alpha\}])$ is the same as that of $\mathcal{P}^2(\text{clan}(A_1 \cup \{\alpha\}))$ by results above (if $A$ has at least two elements). Of course $A_1 \cup \{\alpha\} = (A \cup \{\alpha\})_1$. This establishes $2^{\tau(A)} = \tau(A_1)$ for $A$ with at least two elements, which verifies the naturality property of tangled webs for this $\tau$.

Consider natural models of initial segments of simple type theory with base type $\mathcal{P}^2(\text{clan}[A \cup \{\alpha\}])$. The theory of such a model is determined by the cardinality of its base type. The theory of the first $n$ types of this model, that is the theory of the model with top type $\mathcal{P}^{n+1}(\text{clan}[A \cup \{\alpha\}])$, is completely determined by the first $n + 1$ elements of $A$ by results above. And this establishes the elementarity property of a tangled web for this $\tau$.

We have already shown that the existence of a tangled web implies the consistency of NF.

The assumption that $\lambda > \omega$ and the use of $\alpha$ in the definition of tangled webs is purely technical; we have avoided proving that the theory of the natural model of TST$_n$ with bottom type $\text{clan}(A)$ and top type $\mathcal{P}^n(\text{clan}[A])$ depends only on $A \setminus A_n$ independently of the maximum element of $A$, though this does in fact hold, so the tangled web could be defined as $\tau(A) = |\mathcal{P}^2(\text{clan}(A))|$. 
15 Conclusions to be drawn about NF

The conclusions to be drawn about NF are rather unexciting ones.

By choosing the parameter $\lambda$ to be larger (and so to have stronger partition properties) one can show the consistency of a hierarchy of extensions of NF similar to extensions of NFU known to be consistent: one can replicate Jensen’s construction of $\omega$- and $\alpha$-models of NFU to get $\omega$- and $\alpha$-models of NF (details given above). One can show the consistency of NF + Rosser’s Axiom of Counting (see [13]), Henson’s Axiom of Cantorian Sets (see [4]), or the author’s axioms of Small and Large Ordinals (see [6], [7], [15]) in basically the same way as in NFU.

It seems clear that this argument, suitably refined, shows that the consistency strength of NF is exactly the minimum possible on previous information, that of TST + Infinity, or Mac Lane set theory (Zermelo set theory with comprehension restricted to bounded formulas). Actually showing that the consistency strength is the very lowest possible might be technically tricky, of course. I have not been concerned to do this here. It is clear from what is done here that NF is much weaker than ZFC.

By choosing the parameter $\kappa$ to be large enough, one can get local versions of Choice for sets as large as desired, using the fact that any small subset of a type of the structure is symmetric. The minimum value $\omega_1$ for $\kappa$ already enforces Denumerable Choice (Rosser’s assumption in his book) or Dependent Choices. It is unclear whether one can get a linear order on the universe or the Prime Ideal Theorem: that would require major changes in this construction. But certainly the question of whether NF has interesting consequences for familiar mathematical structures such as the continuum is answered in the negative: set $\kappa$ large enough and what our model of NF will say about such a structure will be entirely in accordance with what our original model of ZFC said. It is worth noting that the models of NF that we obtain are not $\kappa$-complete in the sense of containing every subset of their domains of size $\kappa$; it is well-known that a model of NF cannot contain all countable subsets of its domain. But the models of TST from which its theory is constructed will be $\kappa$-complete, so combinatorial consequences of $\kappa$-completeness will hold in the model of NF (which could further be made a $\kappa$-model by making $\lambda$ large enough).

The consistency of NF with the existence of a linear order on the universe or the Prime Ideal theorem is not established: questions about many weak versions of Choice remain.
The question of Maurice Boffa as to whether there is an $\omega$-model of TNT (the theory of negative types, that is TST with all integers as types, proposed by Hao Wang ([18])) is settled: an $\omega$-model of NF yields an $\omega$-model of TNT instantly. This work does not answer the question, very interesting to the author, of whether there is a model of TNT in which every set is symmetric under permutations of some lower type.

The question of the possibility of cardinals of infinite Specker rank (at least in ZFA) is answered, and we see that the existence of such cardinals doesn’t require much consistency strength. For those not familiar with this question, the Specker tree of a cardinal is the tree with that cardinal at the top and the children of each node (a cardinal) being its preimages under $\alpha \mapsto \mathcal{P}\alpha$. It is a theorem of Forster (a corollary of a well known theorem of Sierpinski) that the Specker tree of a cardinal is well-founded (see [2], p. 48), so has an ordinal rank, which we call the Specker rank of the cardinal. NF + Rosser’s Axiom of Counting proves that the Specker rank of the cardinality of the universe is infinite; it was unknown until this point whether the existence of a cardinal of infinite Specker rank was consistent with any set theory in which we had confidence. The possibility of a cardinal of infinite Specker rank in ZFA is established by the construction here; we are confident that standard methods of transfer of results obtained from FM constructions in ZFA to ZF will apply to show that cardinals of infinite Specker rank are possible in ZF.

This work does not answer the question as to whether NF proves the existence of infinitely many infinite cardinals (discussed in [2], p. 52). A model with only finitely many infinite cardinals would have to be constructed in a totally different way. We conjecture on the basis of our work here that NF probably does prove the existence of infinitely many infinite cardinals, though without knowing what a proof will look like.

A natural general question which arises is, to what extent are all models of NF like the ones indirectly shown to exist here? Do any of the features of this construction reflect facts about the universe of NF which we have not yet proved as theorems, or are there quite different models of NF as well?
16 References and Index

References


Index

$D_T$, domain component of an argument list type $T$, 29

$X$, introduced, a parameter of the construction, 26

$\kappa$, parameter of the construction, introduced for the main construction, 23

$\kappa$-amorphous set, 42

$\lambda$, introduced, a parameter of the construction, 26

$\lambda$, parameter first mentioned, 18

$\lambda$, parameter of the construction, introduced, 24

$\rho_T$, relative type component of an argument list type $T$, 29

$\tau_T$, absolute type component of argument list type $T$, 29

Ambiguity Scheme, 13

argument list type names, 30

argument list types, definition, 29

argument lists introduced, 28

argument lists, extension order on, 29

atoms, 23

atoms, free action of permutations on, 37

cardinality of iterated power sets in relation to parent sets, 47

Choice, axiom of, false in NF, 13

clan index, 26

clans, introduced in the code section, 28

codability of all symmetric sets, 39

coded substitution extension, 33

codes for atoms, 27

codes for atoms of type $C^0(A)$, 30

codes for near-litters introduced, 28

codes for sets, 30

codes, types of, 27

coding function, 36

complexity of set code types, 32

double power set lemma, 44

downward extension of clan indices, 26

elementarity property of tangled webs, 20

equivalence of codes introduced, 26

extension order on argument lists, 29

external cardinalities of parent sets and clans, further remark, 41

external cardinality of parent sets first discussed, 36

formal element of a code, 30

formal parent of a code, 27

formal parent of a code for a near-litter, 28

Frankel-Mostowski construction, 23

Frankel-Mostowski construction, our specific one defined, 41

free action of permutations on atoms, 37

function codes introduced, 29

function codes with atomic output type, 30

function codes with set output, 30

functions, coding, 36
index, clan, 26
 Infinity, axiom of, a theorem of NF, 13
 input type of a function code, 29
 isomorphisms between natural models of type theory, external, 48
 large sets, 23
 litters are $\kappa$-amorphous, 43
 litters, introduced in the code section, 28
 natural model of TST$_n$, 10
 naturality property of tangled webs, 20
 near-litters, introduced in the code section, 28
 NF defined, 12
 NFU, consistency of, 13
 normal filter, 23
 novel atom notation in an extension of an argument list, 31
 output type of a function code, 29
 parent sets computed from clan indices, 36
 parent, formal, of a code, 27
 parent, formal, of a code for a near-litter, 28
 Permutation Lemma, 37
 power set of a clan fully described in FM interpretation, 44
 rank, set theoretical, of codes, 32
 Redundancy Lemma, 38, 40
 set code, 28
 set code types, complexity of, 32
 set codes defined, 30
 sethood of the collection of function codes, 32
 small sets, 23
 subsets of a clan fully described in FM interpretation, 44
 substitution extension, 33
 Substitution Property, 37
 supports from argument lists, 38
 symmetric sets are codable, 39
 tangled web of cardinals, defined, 20
 tangled web, consistency of NF proved from a, 20
 tangled web, the actual one constructed is defined, 49
 TST, 9
 TST$_n$, 10
 type raising, 11
 type theory, simple, 9
 types of code, 27
 types of set codes, complexity of, 32
 ZFA, 23