Syntax without arithmetic or concatenation

P.D.Welch[1]
School of Mathematics
University of Bristol
England

Draft Oct 10, 2012

Abstract

We make an attempt to derive the basic notions of syntax in the system GJ.

1 Introduction

We are inspired by Devlin [1], but mostly Mathias [5] in Section 2, and Dodd [2] in Section 3. The motivation is to show how in a weak set theory syntax, and then eventually semantics can be introduced. Gandy developed such a theory in [3] and Jensen independently with Karp [4] developed a theory of *rudimentary set functions* which can be formulated in a very similar theory to Gandy's. Following Mathias we use "GJ" for this basic theory, which is essentially some base axioms but together with "Δ_0-closure." Mathias makes an extensive survey of such weak theories paying attention to Gandy's and the others' papers (in which several theories are mentioned). Moreover he derives many common notions in as simple a theory as possible. Any unexplained terminology and notation are taken from this paper.

Mathias devotes two sections to a system of Devlin in [1], which Devlin called "BS." Devlin had hoped to develop a theory that would hold of both the limit levels of the Goedel constructible hierarchy, thus the levels L_λ for limit ordinals λ, and also for the rudimentary closed hierarchy of J_α's developed by Jensen. Two birds would be killed with one stone. Unfortunately as is now well known (see the review in particular of Stanley [6]) BS is not up to the task, although assuming that ω is a set, it cannot prove the existence of sets such as $[\omega]^3$ the set of all 3 element subsets of ω. (Mathias gives an extensive array of models of the various theories illustrating the failure or otherwise of certain properties.) It then becomes difficult to perform elementary syntax. Devlin assumes a language $\mathcal{L}_V$ with constants $\dot{x}$ for every $x \in V$. However even without this, the problem is one of performing the concatenation of codes for formulae which are taken as certain basic finite sequences of sets. As Gandy realised, his system cannot establish the existence of the graph of the addition function (even as a class). Hence the function $f_+(n, m) = k$ is not definable in his theory (and hence neither is the concatenation function $h(u, v) = u \frown v$ for finite sequences). In essence the functions definable in GJ, the rudimentary functions, can only raise rank by a fixed finite constant beyond that of its inputs.

Gandy's solution was to define different numerals than the finite von Neumann ordinals, on which addition could be shown to exist. But the system is notationally somewhat tortuous. Mathias suggests three systems to remedy the defects in BS, one of which - as also suggested by Stanley - is the theory of BS augmented by the rudimentary functions, or in essence GJ augmented by an axiom of infinity. Mathias shows how the approaches to syntax in [1], which involve concatenation, needed for performing the conjunction of two formulae, can be done using "sufficiently long (but finite) attempts at the addition function." (Even this requires that GJ be augmented by some instances of a class Π_1-Foundation Scheme.) Then syntactical notions become of the order of Δ_1^{GJ}. Devlin also naturally defines a relation of satisfaction $\mathrm{Sat}(u, x)$, that a formula coded by u is satisfiable in a structure $\langle x, \in\rangle$. This also requires, as Mathias shows, the use of attempts at addition.

The approach we sketch here sidesteps the difficulty of addition and concatenation (although we adopt the same base theory suggested by Mathias: GJ + certain instances of Π_1-Foundation - which he calls GJ alone but we shall explicitly call $\mathrm{GJ}\Pi_1$). Whereas Devlin used strings to code formula and needed addition to concatenate them, and a $\mathrm{Build}(f, u)$ relation to indicate that the formula coded by u could be seen to be built using the function f, we choose to code formulae by finite trees directly. Thus the code displays the structure of the formula without having to deal with concatenation and a Build predicate, and the latter becomes redundant. One might think that this only delays the problem, since to consider subformulae of a formula is tantamount to taking subtrees of a tree below certain nodes u. And this surely requires the ability to 'unconcatenate', or perform subtraction? Indeed it does, but we can get to a $\mathrm{Sat}(u, x)$ relation without having to consider the set of subformulae of a formula, nor with defining 'Sub' a substitution function, nor the set 'FV' of free variables of a given formula. The way we define the codes comes with a natural enumeration of the tree in the lexicographic ordering; these binary trees are then labelled with sets to reflect the formula being constructed.

For us then 'Fml' and other syntactical notions remain Δ_0^{GJ} and not Δ_1^{GJ}. Mathias asks whether the system GJI (which includes Π_1-Foundation) affords a nicer approach to the problem of defining $\mathrm{Sat}(u, x)$ and we think that the use of trees as codes seems to give a streamlined route to that. [It seems only to require $\mathrm{GJ}\Pi_1$?]

In the third section we look at the relationship between GJ and the theory of basic rudimentary functions R. This account is based heavily on Dodd (where he calls our R instead R_0).

2 Sat for L

We shall consider the language $\mathcal{L} = \mathcal{L}_{\dot\in, \dot=}$ (this is LST in [1] and [5]) that we have been using to date, that can be interpreted in $\in$-*structures*, that is any structure $\langle X, E\rangle$ with a domain a class of sets X and an interpretation E for the $\dot\in$ symbol.

We make therefore a choice of *coding* of the language $\mathcal{L}$ by sets in V. The method of coding itself is not terribly important, there are many ways of doing this, but the essential feature is that we want a mapping of the language into a class of sets with the latter simply definable. As we are mainly interested in the first order language $\mathcal{L}$ we give the definitions in detail just for that. In principle we could do this for any language, thus for any class of first order structures. We choose to code formulae by the trees that can be used to represent the compositional make-up of those formulae.

Some preliminaries: we follow the notation and, we hope, some of the spirit of [5]. We refer to that paper for our theories, and terms, but we define here:

Definition 1. (The theory GJ) *(i) Extensionality ; (ii) Pairing ; (iii) Union*
(iv) Set Foundation: $\forall x(x \neq \varnothing \longrightarrow \exists y \in x(y \cap x = \varnothing))$;
(v) Δ_0*-closure: for* $\Delta_0\ \varphi$ $\forall u \forall \vec{x} \forall v \exists y \forall z(z \in y \leftrightarrow \exists t \in w\,[\,z = \{s \in u \mid \varphi(s, t, \vec{x})\ \}])$.

Other axioms will be added explicitly when required.

Lemma 1. *"$x \in \omega$" is Δ_0^{ReS}, "$x = \bigcup \bigcup a$" is S_0-suitable; ω is S_0-semi-suitable. Let:*

$$
\begin{array}{lll}
\text{“}y = x + 1\text{”} & \longleftrightarrow & x \in \omega \wedge y = x \cup \{x\} \\
\text{“}t \subseteq a \times \omega\text{”} & \longleftrightarrow & (\forall v \in t)(\exists b \in \bigcup \bigcup t \cap a)(\exists u \in \bigcup \bigcup t \cap \omega)(v = \langle b, u\rangle) \\
\text{“}\mathrm{Finseq}(x)\text{”} & \longleftrightarrow & \mathrm{Fun}(x) \wedge \mathrm{dom}(x) \in \omega \\
\text{“}\mathrm{Finseq}_2(x)\text{”} & \longleftrightarrow & \mathrm{Finseq}(x) \wedge \mathrm{ran}(x) \subseteq 2 = \{0, 1\} \\
\text{“}u \in {}^m 2\text{”} & \longleftrightarrow & \mathrm{Finseq}_2(u) \wedge \mathrm{dom}(u) = m;\ \text{“}u \in {}^{\leq m}2\text{”}\ \textit{sim} \\
\text{“}v = u \frown i\text{”} & \longleftrightarrow & \mathrm{Finseq}_2(u) \wedge \forall k \in \mathrm{dom}(u)(u(k) = v(k) \wedge v(k)(\mathrm{dom}(k)) = i \\
\text{“}v = u \restriction k\text{”} & \longleftrightarrow & \mathrm{Finseq}_2(u) \wedge k \in \mathrm{dom}(u) \wedge \forall l < k\ v(l) = u(l) \\
\text{“}k = \|q\|\text{”} & \longleftrightarrow & \mathrm{Finseq}(q) \wedge \mathrm{dom}(q) = k + 1
\end{array}
$$

(For $\mathrm{Finseq}(q)$ *we write* $q=(q_0, q_1, \ldots, q_{\|q\|})$

"$u <_{\mathrm{lex}} v$ " $\longleftrightarrow$ $\mathrm{Finseq}_2(u) \wedge \mathrm{Finseq}_2(v) \wedge \exists k \in \mathrm{dom}(v)[u = v \restriction k \vee (u \restriction k = v \restriction k \wedge u(k) < v(k)]$

"$q_i\, {}^1\!\!\subset q_j$" $\longleftrightarrow$ $\mathrm{Finseq}(q_i), \mathrm{Finseq}(q_j) \wedge q_i \subset q_j$ *and* $\mathrm{dom}(q_j) = \mathrm{dom}(q_i) + 1$.

The above are all Δ_0^{GJ}*. (Even* Δ_0^{ReS}*?)*

Lemma 2. *The following are* Δ_0^{GJ}*. (Even* Δ_0^{ReS}*?)*

(i) "$q \in \mathrm{FTree}$*" ("q is a finite binary tree")*

$\longleftrightarrow$ $\mathrm{Finseq}(q) \wedge \mathrm{ran}(q) \subseteq {}^{\leq\|q\|}2 \wedge \forall i \leq \|q\| \forall k \in \mathrm{dom}(q_i) \exists j < i\,(q_j = q_i \restriction k) \wedge$
$\forall i, j < \|q\|\,(i < j \longleftrightarrow q_i <_{\mathrm{lex}} q_j)$*:*

(ii) "$q \in \mathrm{FTree}(L)$*" ("q is a finite binary tree with labels from L")*

$\longleftrightarrow$ $\mathrm{Finseq}(q) \wedge \mathrm{ran}(q) \subseteq {}^{\leq\|q\|}(2 \times L) \wedge q_0 \in L \wedge$
$\forall i \leq \|q\| (0 < i \longrightarrow \forall k \in \mathrm{dom}(q_i) \exists j < i\,(q_j = q_i \restriction k) \wedge$
$\forall i, j < \|q\| (\,i < j \longleftrightarrow q_i^{\mathrm{left}} <_{\mathrm{lex}} q_j^{\mathrm{left}})$ *where we set:*
$q_0^{\mathrm{left}} = (\ ) \wedge \forall k \in \mathrm{dom}(q_i)\,(\ q_i^{\mathrm{left}}(k) = (q_i(k))_0\,)$*;*

(iii) "$L(i)$ *exists"; where "*$q_{L(i)}$ *is the immediate left successor of* q_i *in* q *(if it exists)"*

$q \in \mathrm{FTree}(L) \wedge i < \|q\| \longrightarrow [L(i) = j\ \leftrightarrow \exists j < \|q\|\,(q_j \supset^1 q_i \wedge (q_j)_0 = 0)]$

(iv) "$R(i)$ *exists"; where "*$q_{R(i)}$ *is the immediate right successor of* q_i *in* q *(if it exists)"*

$q \in \mathrm{FTree}(L) \wedge i < \|q\| \longrightarrow [R(i) = j\ \leftrightarrow \exists j < \|q\|\,(q_j \supset^1 q_i \wedge (q_j)_0 = 1)]$

In the definition of FTree the tree grows downwards from the root, and the sequence $q = (q_0, q_1, \ldots, q_{\|q\|})$ consists of finite sequences into 2, which are ensured to be a tree by closing under the taking of initial segments; moreover the enumeration of the tree by the q_i is done in a strict lexicographic ordering: if we picture the tree as growing downwards from the root $q_0 = q(0) = (\ )$ (the empty finite sequence) then we branch to the left with a 0 and to the right with a 1; if we enumerate the tree downwards along the leftmost path of nodes not yet enumerated we arrive at the lexicographic ordering. In particular the nodes further down a branch below a node q_i are enumerated later, thus by some q_j with $i < j$. The final rightmost node is $q_{\|q\|}$.

A labelled tree is simply a finite binary tree adorned with labels from L: we want to keep the same enumeration and ordering on the underlying binary tree structure, and hence we order just using the "left" components of the range of each q_i. These are sequences such as ( ), (00), (00110..0) to indicate branching turns, and determine the order; the right part is a sequence of labels such as $(l_0, l_1, \ldots l_m)$ which we can regard as simply attachments to the underlying tree. The topmost label l_0 is attached to the empty sequence $\varnothing = (\ )$.

We wish to define "$\mathrm{Fml}(q)$" with q a finite labelled tree to fulfill the role of being the object counterpart of a formula φ in LST. The labels will pick out the manner in which the formula is coded, and will be elements of $L = 5 \cup \{4\} \times \omega$. We give some further notational definitions. Suppose $q \in \mathrm{FTree}(L)$, $q = (q_0, q_1, \ldots, q_{\|q\|})$, we set, for $i > 0$:

$m_i = \max\{\mathrm{dom}(q_i)\}$ (m_i is thus the length of the sequence from the root to the node.)

$l_i = (q_i(m_i))_1$; $j_i = (q_i(m_i))_0$ where $(u)_0, (u)_1$ are the usual un-pairing functions; l_i is then the label at the end of the path to the node q_i, j_i the 0 or 1 indicating the last turn.

Definition 2. *We set* $\mathrm{Vbl} = \{n \mid n \in \omega \backslash 5\}$.

$\mathrm{Fml} =_{\mathrm{df}} \{q \mid q \in \mathrm{FTree}(L) \wedge \forall i \leq \|q\|\ [$

(i) $l_i = 0, 1, 3 \longrightarrow L(i), R(i)$ *exist* $\wedge$

(ii) $l_i = 0, 1\ \longleftrightarrow\ (l_{L(i)}, l_{R(i)} \in \mathrm{Vbl}) \leftrightarrow (l_{L(i)} \in \mathrm{Vbl} \vee l_{R(i)} \in \mathrm{Vbl})\ \ \wedge$

(iii) $l_i \in \mathrm{Vbl} \longleftrightarrow \forall j \neq i (q_j \not\supset q_i)\ \ \wedge$

(iv) $l_i = 2, \langle 4, n+5\rangle \longrightarrow R(i)$ *does not exist*].

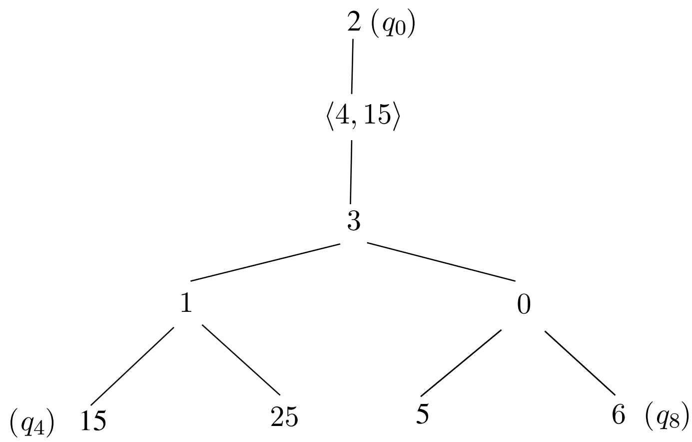
Interpretation: Line *(i)*: below a node labelled 0,1,3 the tree splits. Think of 3 as labelling a conjunction of the two formula coded below it.

Line *(ii)*: "If a label is a variable, then this occurs below a node labelled with 0 or 1 (thought of as for $=$ of $\in$ respectively), below which the other branch is also immediately labelled with a variable (think "$v_n = v_m$" *etc.*). Thus below a 0 or 1 is a code for an atomic formula, which (see *(iii)*) terminates that part of the tree.

Line *(iii)*: Nodes labelled with variables are indeed precisely the terminal nodes.

Line *(iv)*: Nodes labelled with 2, or $\langle 4, n\rangle$ do not split; these represent $\neg$ or "$\exists v_n$" in front of the formula coded below it respectively.

The following picture is illustrative.



$$\ulcorner \neg\exists v_{10}(v_{10} \in v_{20} \wedge v_0 = v_1)\urcorner$$

Each formula φ of LST has a unique code as an element q of $\mathrm{FTree}(L)$. We shall set in this case $q = \ulcorner\varphi\urcorner$. Similarly each tree of $\mathrm{FTree}(L)$ corresponds to a well-formed formula. Note that "$q \in \mathrm{Fml}$" is Δ_0^{GJ}.

Exercise 1. Define a suitable Δ_0^{GJ} expression for $\mathrm{Fml}_0(q)$ where q now is supposed to represent a Δ_0-formula. [This can be done by simply adding clauses to $\mathrm{Fml}(q)$ so that whenever "$\exists v_n$" appears then immediately below it must appear the conjunction "$v_n \in v_m \wedge \psi$" for some m and ψ.]

Lemma 3. (GJ) *For* $q \in \mathrm{Fml}$ *let* $\mathrm{Var}(q)$ *be the set of variables occurring (quantified or free) in* q*. Then* "$\mathrm{Var}(q)$" *is* Δ_0^{GJ}.

Proof: $\mathrm{Var}(q) = \bigcup \{\mathrm{ran}^2(q_i) \mid i \le \|q\|\}\backslash 5.$ QED

Exercise 2. Let $\mathrm{FV}(q)$ be the set of free variables occurring in q. Show that for $q \in \mathrm{Fml}$, "$\mathrm{FV}(q)$" is Δ_0^{GJ}.

Let $\mathrm{GJ}\Pi_1$ denote Gandy-Jensen with Π_1-Foundation. We may define a rank function on finite trees rk_q in general, and on our Fml trees in particular; thus the Vbl nodes get rank zero, and ranks increase in the tree upwards in the usual fashion.

Lemma 4. (GJ) $\forall q \in \mathrm{FTree}(L)\ \ \mathrm{rk}_q \in V$.

Proof: Let $q = (q_0, q_1, \ldots, q_{\|q\|}) \in \mathrm{FTree}(L)$. We define rk_q by induction on $n - \|q\|$. Let $\mathrm{rk}_q(q_{\|q\|}) = 0$ (as this is a terminal node). If $\mathrm{rk}_q(q_j)$ is defined for $j \in [i+1, \ldots, \|q\|]$ let $\mathrm{rk}_q(q_i) = \max(\{\mathrm{rk}_q(q_j) \mid q_j \supset^1 q_i, i < j \leq \|q\|\})$. (Recall our convention on the way q is listed; we set $\max(\varnothing) = 0$.) If for some $k < \|q\|$ $\mathrm{rk}_q(q_{\|q\|-k})$ were undefined by Foundation there would be some least $k < \|q\|$ for which this happened. A contradiction follows as usual. QED

(We do not actually use the above?)

We proceed straight to defining satisfaction.

Definition 3. *(i)* $Q_x =_{\mathrm{df}} \{h \mid \mathrm{Finseq}(h) \wedge \mathrm{ran}(h) \subseteq x\}$;
$Q_x^n =_{\mathrm{df}} \{h \mid \mathrm{Finseq}(h) \wedge \mathrm{ran}(h) \subseteq x \wedge \mathrm{dom}(h) = n\}$

(ii) For $y \in x$, $h \in Q_x$, we define $h(y/i)$ by $h(y/i)(k) = h(k)$ $(i \neq k)$, $h(y/i)(i) = y$ if $i \in \mathrm{dom}(h)$.

Such as "$u \in Q_x^n$", "$u = h(y/i)$" are Δ_0^{ReS};

Lemma 5. *(i)* $\mathrm{GJ\Pi}_1 \vdash \forall x \forall n \in \omega \exists u (u = Q_x^n)$. *(ii) "$u = Q_x^n$" is $\Delta_1^{\mathrm{GJ\Pi}_1}$*

Proof: This is Mathias [5] 10.27, for *(i)*; *(ii)* Similar to 10.28. QED

Definition 4. $\mathrm{Sat}_0(q, x, i, h)$ *"h satisfies the formula q below the node i in x"*

$\longleftrightarrow q \in \mathrm{Fml} \wedge h \in Q_x \wedge \mathrm{dom}(h) \geq \max(\mathrm{Var}(q)) \wedge \forall i \leq \|q\| \, [$
$l_i = 0/1 \wedge h(l_{L(i)} - 5) =/\!\in h(l_{R(i)} - 5)) \vee$
$l_i = \langle 4, n+5 \rangle \wedge \exists y \in x \, \mathrm{Sat}_0(x, q, L(i), h(y/n)) \vee$
$l_i = 2 \wedge \quad \neg \mathrm{Sat}_0(x, q, L(i), h)) \vee$
$l_i = 4 \wedge \mathrm{Sat}_0(x, q, L(i), h) \wedge \mathrm{Sat}_0(x, q, R(i), h))]$.

Then we may set

$$\mathrm{Sat}(q, x, h) \longleftrightarrow \mathrm{Sat}_0(q, x, 0, h)$$

This is ostensibly an induction using rk_q, the rank of the tree q, from the leaves upwards to the top. However we justify this as follows.

Theorem 1. $\mathrm{Sat}_0(q, x, i, h), \mathrm{Sat}(q, x, h)$ *are* $\Delta_1^{\mathrm{GJ\Pi}_1}$.

Proof:
"$\mathrm{PSat}(f, Q, n, q)$" ("$f$ is a partial satisfaction sequence from Q for q of length n")

$$\begin{aligned}
&\longleftrightarrow q \in \mathrm{Fml} \wedge \mathrm{Finseq}(f) \wedge \mathrm{dom}(f) = n \wedge \forall i < n\, (\, f(i) \subseteq Q) \wedge k+1 \leq n \longrightarrow \\
&[\; f(\|q\|) = \varnothing \wedge \; l_k \in \mathrm{Vbl} \longrightarrow f(k) = \varnothing \wedge \\
&\quad l_k = 0 \backslash 1 \quad \longrightarrow \quad f(k) = \{h \in Q \mid h(l_{L(k)} - 5) = \backslash \in h(l_{R(k)} - 5)\} \wedge \\
&\quad l_k = 3 \quad \longrightarrow \quad f(k) = Q \setminus f(L(k))) \wedge \\
&\quad l_k = \langle 4, n+5 \rangle \quad \longrightarrow \quad f(k) = \{h \in Q \mid \exists y \in x \, h(y/n) \in f(L(k))\} \wedge \\
&\quad l_k = 2 \quad \longrightarrow \quad f(k) = f(L(k)) \cap f(R(k))\,].
\end{aligned}$$

We are thus working our way backwards through the lexicographic order on the formula tree (hence starting with $f(\|q\|) = \varnothing$ as the latter is always a Vbl node (if $q \neq \varnothing$)). Then

$\mathrm{Sat}(q, x, h) \longleftrightarrow \exists Q \exists f \; (\; \mathrm{PSat}(f, Q, n, q) \wedge Q = Q_x^{\max(\mathrm{Var}(q))} \wedge \mathrm{dom}(f) = \|q\| + 1 \wedge h \in f(0))$.

This is a $\Sigma_1^{\mathrm{GJ\Pi}_1}$ form, and we get a Π_1 form by:

$\neg\mathrm{Sat}(q,x,h)\longleftrightarrow(h\notin Q_x^{\max(\mathrm{Var}(q))}\vee q\notin\mathrm{FTree}(L))\vee\mathrm{Sat}(r,x,h)$ where if $q=\ulcorner\varphi\urcorner$ then $r=\ulcorner\neg\varphi\urcorner$. QED

We then write:

$\langle x,\in\rangle\vDash q[h]\longleftrightarrow\mathrm{Sat}(q,x,h)$.

We adopt the convention that when writing such as the last line, that the formula φ with $q=\ulcorner\varphi\urcorner$ is *orderly*, that is, if v_j occurs withing the scope of a quantifier $\exists v_k$, that $j<k$. By the ordinary rules of predicate calculus this is harmless. We may then also adopt the convention that we write

$\langle x,\in\rangle\vDash\ulcorner\varphi\urcorner[x_0,x_1,\ldots,x_{n-1}]$ to mean that for any $h\in Q_x$ with $h(i)=x_i$ we have (a) the free variables of φ are amongst (but need not be all) the v_i $(i<n)$ (b) $\mathrm{Sat}(q,x,h\restriction\max(\mathrm{Var}(q)+1))$.

3 Rudimentary Functions

A different presentation of the constructible universe is possible. In this alternative presentation a finite set of very simple functions can be used to create the constructible sets by a recursive process of iterating and closing them off, along all the ordinals. In this way reference to many logical notions could be seemingly eliminated. [Bernays]

These models were closed under certain basic functions in a way that the typical $L_{\beta+1}$ of the constructible hierarchy is not. [Commentary here]

We first give a list of functions which will form the basis of all rudimentary functions: any rudimentary function $f(x_1,\ldots,x_n)$ will a finite composition of them.

Definition 5. (The Basis Functions)

$f_0(x,y)=\{x,y\}$ $\quad f_4(x,y)=\mathrm{dom}(x)$

$f_1(x,y)=\bigcup x$ $\quad f_5(x,y)=\{x``\{z\}\mid z\in y\}$

$f_2(x,y)=x\backslash y$ $\quad f_6(x,y)=\{\langle u,v,w\rangle\mid u\in x\wedge\langle v,w\rangle\in y\}$

$f_3(x,y)=x\times y$ $\quad f_7(x,y)=\{\langle u,w,v\rangle\mid\langle u,v\rangle\in x\wedge w\in y\}$

$f_8(x,y)=\{\langle u,v\rangle\in x\times y\mid u=v\}$

$f_9(x,y)=\{\langle u,v\rangle\in x\times y\mid u\in v\}$

$f_{10}(x,y)=\langle x,y\rangle$

$f_{11}(x,y)=\langle x,v,w\rangle$ *if* $y=\langle v,w\rangle$; 0 *otherwise*

$f_{12}(x,y)=\langle u,y,v\rangle$ *if* $x=\langle u,v\rangle$; 0 *otherwise*

$f_{13}(x,y)=\{\langle x,v\rangle,w\}$ *if* $y=\langle v,w\rangle$; 0 *otherwise*

$f_{14}(x,y)=\{\langle u,y\rangle,v\}$ *if* $y=\langle v,w\rangle$; 0 *otherwise*

$f_{15}(x,y)=\{\langle x,y\rangle\}$

$f_{16}(x,y)=\{t\mid\langle y,t\rangle\in x\}$

$f_{17+i}(x,y)=A_i\cap x$

Remark on: binary functions; connection to *terms*.

Definition 6. *A function* $f:V^n\longrightarrow V$ *is* rudimentary *if it is a (finite) composition of the basis functions.*

Note: to talk about *all* rudimentary functions at this point is a meta-theoretic matter: if we wished to prove some fact about them all, that would require a meta-theoretic induction on the composition tree structure which underlies each such function. Each rudimentary function f can however be defined by a class term for which a defining formula, Φ_f say, can be obtained from the defining formulae of the basis functions $\vec{\Phi}_i$, that go into its composition. Indeed given f and its composition tree, there is an effective way to build Φ_f from $\vec{\Phi}_i$. This point will be returned to below.

We wish to delineate classes $\langle M, \in \rangle$ that are closed under the rudimentary functions. But what precisely are these? We may in fact give an axiomatic characterisation of such sets. However we first remark on the presence of $f_{17},\ldots$. We may wish to consider models with predicates A_i of the form $\langle M, \in, A_0, \ldots, A_{N-1} \rangle$. This could be to build $L_\alpha[\vec{A_i}]$ models. For just constructible sets, we could consider only $f_0, \ldots f_{16}$ and drop the latter ones (by taking $N = 0$). However with an eye on the future we shall include the possibility of these additional rudimentary functions. Given, *e.g.*, that f_{17} gives us access to a predicate A_0, then we shall formalise talk about such predicates in a language with one-placed predicate symbols $\dot{A}_0 \ldots, \dot{A}_{N-1}$: $\mathcal{L}_{\vec{\dot{A}}}$ (containing still $\dot{\in}$, and $\dot{=}$ still). If we wish to drop it on occasion then this will mostly be harmless: we can always set $A_i = \varnothing$ after all.

We have that each rudimentary $f_i(x_1, x_2) = z$ is given by a class term $f_i = \{\langle\langle x_1, x_2\rangle, z\rangle \mid \Phi_i(x_1, x_2, z)\}$ with Φ_i Σ_0. Given a basis function f_i, one may prove that it is total:

Lemma 6. *Let* $i \leq 16 + N$, $\mathrm{GJ} \vdash \forall x \forall y \exists z\, (z = f_i(x, y))$.

- Examples of rud functions.

The $16 + N$ different conclusions of the last lemma form a set of $16 + N$ axioms that together with (i) and (ii) of Def 1 constitute a finite set of axioms for a theory we shall call R.

However as we have remarked above the theory GJ is surprisingly weak when it comes to arithmetic.

One way to deal with talk about all rudimentary functions is to define object language representations for them. Then when we wish to prove something about all rudimentary functions we may quantify over these representing codes. We effect this as follows: each rudimentary function $f(v_{i_0}, v_{i_1}, \ldots, v_{i_n})$ is a composition of the basis functions above applied to specific variables. It thus has a finite binary tree compositional structure (which we imagine growing downwards) from the root which corresponds to the full function $f(v_{i_0}, v_{i_1}, \ldots, v_{i_n})$. At the bottom are the leaves where the basis functions $f_i(v_j, v_k)$ that go into its composition are situated. We shall thus define the *code* of the function by using these trees much as we did for Fml above.

We shall consider a simple term language with terms consisting of variables and functions f_i for $i \leq 16 + N$. From these we build further terms for their compositions in a manner standard for function symbols in a first order language. These terms will again be coded as finite trees.

Exercise 3. Define a rudimentary function $\mathrm{Fr}(q)$ that returns, for $q \in \mathrm{Fml}_0$, the set of free variables of the formula coded by q. [Note that any branch in the tree terminates in a variable v_n, and if $\langle 4, n+5\rangle$ does not occur on the branch then that occurrence of v_n is free.]

Thus the basic function $f_i(v_j, v_k)$ has as code set a finite three node tree.

Definition 7. $\mathrm{RudF} =_{\mathrm{df}} \{q \in \mathrm{FTree}(\omega) \mid \forall i \leq \|q\|\, ((q_i(k))_1 \leq 16 + N \longleftrightarrow L(i), R(i)\ \textit{exist}\,)\}$.

Thus, in such a tree, the nodes labelled with a number less than $16 + N$ are non-terminal, representing f_i, and those labelled with larger integers $17 + N + m$ are terminal, and represent variables v_m $(m < \omega)$.

Evaluations of functions are done in the obvious computational spirit: variables are assigned sets, and then the various basic functions are invoked ascending the tree.

Definition 8. *For* $q \in \mathrm{RudF}$ *(i) let* $H_q = \{h \mid \mathrm{Fun}(h) \wedge \mathrm{dom}(h) = \{l_i \mid i \in \mathrm{dom}(q)\} \backslash 17 + N$.

(ii) $E_q(h) = y \leftrightarrow_{\mathrm{df}} h \in H_q \wedge \exists g$

$$[\mathrm{Fun}(g) \wedge \mathrm{dom}(g) = \mathrm{dom}(q) \wedge \forall i \in \mathrm{dom}(g)(l_i > 16 + N \longrightarrow g(i) = h(l_i)$$

$$\wedge\, l_i \leq 16 + N \longrightarrow g(i) = f_{l_i}(g(L(i)), g(R(i)))\,) \wedge g(0) = y)$$

$$\leftrightarrow \quad h \in H_q \wedge \forall g$$

$$[\mathrm{Fun}(g) \wedge \mathrm{dom}(g) = \mathrm{dom}(q) \wedge \forall i \in \mathrm{dom}(g)(l_i > 16 + N \longrightarrow g(i) = h(l_i)$$

$\wedge\, l_i \leq 16 + N \longrightarrow g(i) = f_{l_i}(g(L(i)), g(R(i))) \longrightarrow g(0) = y)$

The above two equivalent forms show that for $q \in \mathrm{RudF}$ we have a $\Delta_1^{\mathrm{GJ}}(q, h, y)$ expression for the evaluation relation $E_q(h) = y$, when in the above for instances of "$u = f_i(v, w)$" we write in the defining formula $\Phi_i(v, w, u)$.

Theorem 2. *For any* Δ_0, φ *in* $\mathcal{L}_{\vec{A}}$ $R \vdash \varphi \Longleftrightarrow \mathrm{GJ} \vdash \varphi$. *Hence* GJ *is finitely axiomatisable.*

Proof: Lemma 6 shows that $R \subseteq \mathrm{GJ}$. For the converse it suffices to show that Δ_0-closure, *(v)*$_\varphi$, holds for every Δ_0 φ. Two sublemmas enable this, the first of which is key and lengthy, and the second is merely technical. However, we are not motivated merely by a finite axiomatisability result, as the first lemma in particular shows how we may go about directly defining a satisfaction relation for models of R.

Lemma 7. *For any* Δ_0 φ *there is a rudimentary function* F *so that*

$$(7) \quad F(a, v_0, \ldots, v_{k-1}, v_{k+1}, \ldots v_n) = \{v_k \in a \mid \varphi(v_0, \ldots v_n)\}.$$

Proof: We have adopted two conventions: (a) for a formula φ an expression such as $\varphi(v_0, \ldots v_n)$ denotes also the formula φ but indicates that the free variables are amongst the v_0, $\ldots v_n$, but need not be all of them; (b) that whenever v_i is a variable occurring within the scope of a quantifier $\exists v_j$ then $i < j$. The argument is in essence a (meta-theoretic) induction on the structure of Δ_0 φ. At the same time the reader should bear in mind how the constructions of the functions in the clauses below could be done within our formal theory, and would prove the existence of an effective term for a function $k_0\colon \mathrm{Fml}_0 \times \omega \longrightarrow \mathrm{RudF}$ with $(\ulcorner\varphi\urcorner, n) \rightarrowtail f_\varphi$ sending our previously defined code of φ as a member of Fml_0, to the code of the tree defining the F_φ we shall define below: in building these, clauses (i)-(xiv) prescribe how we may obtain a tree by simple finite effective alterations to the tree for F_ψ which by induction we shall assume we are in possession of. We shall not spell these details out more than once.

(i) $\varphi(v_0, \ldots v_n)$ *denotes the same formula as* $\psi(v_0, \ldots v_{n-1})$.

(We shall henceforth abbreviate this as saying that the two are expressions 'are the same'.) Assume we have already a function F_ψ satisfying (7). Thus v_n is not a free variable of φ. Define:

$F_\varphi(v_0, \ldots, v_n) =_{\mathrm{df}} F_\psi(v_0, \ldots, v_{n-1}) \times v_n = f_3(F_\psi(v_0, \ldots, v_{n-1}), v_n)$.

(ii) $\varphi(v_0, \ldots v_n)$ *is* $\psi(v_0, \ldots, v_{n+1})$ *and we have* F_ψ *satisfying* (7).

Define

$F_\varphi(v_0, \ldots, v_n) =_{\mathrm{df}} \mathrm{dom}(F_\psi(v_0, \ldots, v_n, \{\varnothing\}) = f_4(F_\psi(v_0, \ldots, v_n, f_0(f_2(v_0, v_0), f_2(v_0, v_0))), v_0)$.

[If we were giving more details of the map k_0 above, we should be assuming that we have $q = k_0(\ulcorner\psi\urcorner, n+1) \in \mathrm{RudF}$ defined. Then $\bar{q} = k_0(\ulcorner\varphi\urcorner, n)$ would be obtained by taking the tree $q = (q_0, \ldots, q_{\|q\|})$, and replacing all terminal nodes with a variable code for v_{n+1} by the 7 node tree for $f_0(f_2(v_0, v_0), f_2(v_0, v_0)))$; then at the top of the tree we'd have to 'prepend' for the top node of q as follows $(\bar{q}_0)_1 = 4$, $(\bar{q}_1)_1 = (q_0)_1$; then for all other $i \leq \|q\|$ we should set $\bar{q}_{i+1+6.m} = q_i$ where m is the number of occurrences of the variable v_{n+1} appearing on nodes lexicographically to the left of q_i. The rightmost node has $\|\bar{q}\| = \|q\| + 2 + 6.M$ with $(\bar{q}_{\|\bar{q}\|})_1 = 17 + N$ where now M is the total number of occurrences of v_{n+1} in q. It is complicated but it can be done. We shall not be giving these details for any of the other cases below, but leave them to the reader's curiosity. The point of even saying this much is to demonstrate that these manœuvres can be done in an effective, and in particular *via* approximations to recursive functions, in a Δ_1^{GJ}, manner.]

Using (i) and (ii) we have by induction:

(iii) *If* $\varphi(v_0, \ldots v_n)$ *is* $\psi(v_0, \ldots, v_m)$ *and* F_ψ *satisfies* (7) *then we have* F_φ *satisfying* (7).

(iv) *If* F_φ *satisfies* (7) *then we may define* $F_{\neg\varphi}(v_0, \ldots, v_n) = (v_0 \times \cdots \times v_n) \backslash F_\varphi(v_0, \ldots, v_n)$.

(v) *If* $\chi(v_0, \ldots v_n)$ *and* $\psi(v_0, \ldots v_n)$ *have* F_χ *and* F_ψ *satisfying* (7) *then we may define* F_φ *for* $\varphi(v_0, \ldots v_n) \equiv \psi \wedge \chi$ *satisfying* (7) *as:*

$F_\varphi(v_0, \ldots, v_n) =_{\mathrm{df}} F_\psi(v_0, \ldots, v_n) \cap F_\chi(v_0, \ldots, v_n) = F_\psi(v_0, \ldots, v_n) \backslash (F_\psi(v_0, \ldots, v_n) \backslash F_\chi(v_0, \ldots, v_n))$.

(vi) *If for* $\psi(v_0, \ldots, v_n)$ *we have* F_ψ *satisfying* (7) *and* $\varphi(v_0, \ldots v_{n+1})$ *arises from* $\psi(v_0, \ldots, v_n)$ *by substituting all free occurrences of* v_n *by* v_{n+1}, *then there is* F_φ *satisfying* (7).

If $n = 0$ then $\varphi(v_0, v_1) \leftrightarrow \psi(v_1)$ so take $F_\varphi(v_0, v_1) = v_0 \times F_\psi(v_1)$. Otherwise take

$F_\varphi(v_0, \ldots, v_{n+1}) = f_7(F_\psi(v_0, \ldots, v_{n-1}, v_{n+1}), v_n)$.

(vii) *If* $\psi(v_0, v_1)$ *has* F_ψ *satisfying* (7) *and* $\varphi(v_0, \ldots v_n)$ *(for* $n \geq 2$*) arises from* $\psi(v_0, \ldots, v_n)$ *by substituting all free occurrences of* v_0 *by* v_{n-1}, *and free* v_1 *by* v_n, *then there is* F_φ *satisfying* (7). $n = 2$ is trivial so assume $n > 2$. Then take

$F_\varphi(v_0, \ldots, v_n) =_{\mathrm{df}} f_6(v_0 \times \cdots \times v_{n-2}, F_\psi(v_{n-1}, v_n))$.

(viii) *There is* $F_{v_0 = v_1}$ *satisfying* (7)*:* this is just f_8.

(ix) *There is* $F_{v_n = v_{n+1}}(v_0, \ldots, v_{n+1})$ *satisfying* (7)*:* by (vii) and (viii).

(x) *For all* m, n *there is* $F_{v_m = v_n}$ *satisfying* (7).

If $m = n$ then let $F_{v_n = v_n}(v_0, \ldots, v_n) = v_0 \times \cdots \times v_n$; if $m < n$ this then follows by an induction on $n - m$; if $m > n$ then use $F_{v_n = v_m}$.

(xi) *For all* $m < n$ *there is* $F_{v_m \in v_n}$ *satisfying* (7). This follows by similar arguments to (x).

(xii) *For all* m, n *there is* $F_{v_m \in v_n}$ *satisfying* (7).

As $m < n$ has been dealt with, assume $\varphi(v_0, \ldots v_m)$ is $v_m \in v_n$. Let $\psi(v_0, \ldots, v_{m+1})$ be $v_n = v_{m+1} \wedge v_m \in v_{m+1}$; by (iii), (v), (x), and (xii) there is F_ψ satisfying (7). So let

$F_{v_m \in v_n}(v_0, \ldots, v_m) = \mathrm{dom}(F_\psi(v_0, \ldots, v_m, v_n))$.

(xiii) *For* $i \leq N$ *there is* F_{A_i} *satisfying* (7). Suppose $\varphi(v_0, \ldots v_n)$ is $A_i(v_m)$. Then

$F_{A_i}(v_0, \ldots v_n) = v_0 \times \cdots \times v_{m-1} \times f_{17+i}(v_m, v_m) \times \cdots \times v_n$.

At this stage we have shown all atomic formulae $\varphi(v_0, \ldots v_m)$ have appropriate F_φ and that propositional connectives can be dealt with. We only have now to show:

(xiv) *If* $\psi(v_0, \ldots, v_j)$ *has* F_ψ *satisfying* (7) *and* $\varphi(v_0, \ldots v_n)$ *is* $\exists v_j \in v_k \psi(v_0, \ldots, v_j)$ *where* $j > n$ *then there is* F_φ *satisfying* (7).

Let $\chi(v_0, \ldots v_j)$ be $v_j \in v_k$ (recall by our convention that $k < j$). So there is $F_{\chi \wedge \psi}$ satisfying (7), and

$F_{\chi \wedge \psi}(v_0, \ldots, v_{j-1}, \bigcup v_k) = \{\langle x_0, \ldots, x_j \rangle \mid x_j \in x_k, x_m \in v_m\ (m \leq j) \wedge \psi(x_0, \ldots, x_j)\}$.

Now take $F_\varphi(v_0, \ldots, v_n) = \mathrm{dom}^{j-n}(F_{\chi \wedge \psi}(v_0, \ldots, v_n, v_0, v_0, \ldots, v_0, \bigcup v_k))$.

This concludes the proof. QED Lemma 7

From the discussion concerning the function k_0 we assert:

Corollary 1. *There is a* Δ_1^{GJ} *effective function* $k_0 \colon \mathrm{Fml}_0 \times \omega \longrightarrow \mathrm{RudF}$ *so that* $k_0(\ulcorner \varphi \urcorner, n)$ *is a code of a tree for* $F_\varphi(v_n, v_0, \ldots, v_n)$ *satisfying* (7), *if the free variables of* φ *are amongst the* $v_0, \ldots, v_n$*; otherwise* $k_0(\ulcorner \varphi \urcorner, n) = \varnothing$.

Exercise 4. For each $1 \leq i, j < k$ show that
$F_k^{i,j}(a_1, \ldots a_k) =_{\mathrm{df}} a_1 \times \cdots a_{i-1} \times a_j \times a_{i+1} \times \cdots \times a_{j-1} \times a_i \times a_{j+1} \times \cdots \times a_k$ is rudimentary.

Lemma 8. *For any* $i \leq 16 + N$ R *proves the function* $f_i^*(v_0, v_1) =_{\mathrm{df}} f_i``v_0 \times v_1$ *is also rudimentary.*

Proof: For each such i we define g_i a composition of basis functions such that:

$g_i(v_0, v_1) = \{\langle \langle x_0, x_1 \rangle, y \rangle \mid x_0 \in v_0 \wedge x_1 \in v_1 \wedge y \in f_i(x_0, x_1)\}$.

[Then we may set $f_i^*(v_0, v_1) = f_6(g_i(v_0, v_1), v_0 \times v_1)$.]

(0)

[Routine details here]

QED

It follows by induction on the composition of rudimentary functions that:

Lemma 9. *For any rudimentary function* $f(x_0,\ldots,x_n)$, $f^*(x_0,\ldots,x_n) =_{\mathrm{df}} f``x_0 \times \cdots \times x_n$ *is also rudimentary.*

We may now conclude that $R \vdash (v)_\varphi$. QED (Theorem 2)

Definition 9. *A relation* R *is* rudimentary *if there is a rudimentary* F *so that:*

$$R(x_0,\ldots,x_n) \Longleftrightarrow F(x_0,\ldots,x_n) \neq \varnothing \quad (9).$$

We now may prove:

Lemma 10. *Every* Δ_0 *relation is rudimentary. Moreover if any* Δ_0 φ *defines the relation* $R(x_0,\ldots,x_n)$*, there is an effective way to construct a rudimentary function* $F_\varphi(x_0,\ldots,x_n)$ *so that (9) holds. In particular:*

There is a recursive function $k\colon \mathrm{Fml}_0 \times \omega \longrightarrow \mathrm{RudF}$ *so that* $m = k(\ulcorner\varphi\urcorner, n)$ *is a code of a tree for a rudimentary function* $g_m(v_0,\ldots,v_n)$ *satisfying* (9)*, if the free variables of* φ *are amongst the* $v_0,\ldots,v_n$*; otherwise* $k(\ulcorner\varphi\urcorner, n) = \varnothing$

Proof: Let φ be a Δ_0 formula; then by Lemma 7, there is a rudimentary function $G(a, x_0,\ldots,x_n) = \{x_{n+1} \in a \mid \varphi(x_0,\ldots,x_n) \wedge x_{n+1} = x_{n+1}\}$. Then set $F_\varphi(x_0,\ldots,x_n) = G(\{\varnothing\}, x_0,\ldots,x_n)$.

The proof of Lemma 7 indicated at the same time how to inductively define on code sets for trees in Fml_0, a code in RudF for the function f, this is built up inductively in a manner that step-by-step mirrors the inductive construction of G, then finally $F_\varphi(x_0,\ldots,x_n)$. QED

Suppose $\varphi(v_0,\ldots v_m)$ defines the rudimentary relation $R(v_0,\ldots,v_m)$. By Lemma 10 there is a recursive map k that returns for us a code for a rudimentary function F_φ satisfying 9.

This allows a significant result on satisfaction and so truth definitions over transitive models of R.

Definition 10. *(i)* For Δ_0 formulae φ, the Δ_0-satisfaction relation *is the relation*

$$P(\ulcorner\varphi\urcorner, n, \langle v_0,\ldots,v_n\rangle) \longleftrightarrow_{\mathrm{df}} f_{k(\ulcorner\varphi\urcorner,n)}(v_0,\ldots,v_n) \neq \varnothing.$$

(ii) The Σ_m-satisfaction relation *is the relation*

$$\exists y_1 \forall y_2 \cdots Q y_m\, P(\ulcorner\varphi\urcorner, m+n, \langle y_1,\ldots y_m, v_0,\ldots,v_n\rangle)\ ;$$

For the former we shall write: $\models_{\Delta_0} \ulcorner\varphi\urcorner(v_0,\ldots,v_n)$, and the latter $\models_{\Sigma_m} \ulcorner\varphi\urcorner(v_0,\ldots,v_n)$.

Lemma 11. *(i) The* Δ_0*-satisfaction relation* $\models_{\Delta_0}\varphi(v_0,\ldots,v_n)$ *is* Δ_1*-definable over any transitive model of* $\mathrm{GJ}\Pi_1$.

(ii) The Σ_m*-satisfaction relation* $\models_{\Sigma_m}\varphi(x_1,\ldots,x_n)$ *is* Σ_m*-definable over any transitive model of* $\mathrm{GJ}\Pi_1$ *for* $m > 0$.

Proof: (i) Let $M = \langle M, A_0,\ldots,A_{N-1}\rangle$ be a transitive model of R. Then M is closed under the rudimentary functions $f_0,\ldots,f_{16+N}$. Further it contains all elements of $\mathrm{FTree}(L)$ (for every $K \in \omega$, GJ proves the class of trees q of length $\|q\| \leq K$ in $\mathrm{FTree}(4 \times K)$ is Δ_0 definable and a set in V ([5] 10.10), and M is transitive). It similarly contains all elements of RudF. The Lemma is easier to prove if we additionally assume that $\mathrm{HF} \in M$, for then as the graph of any recursive function is Δ_1-definable over $\langle \mathrm{HF}, \in\rangle$ we may write,

$$\begin{aligned}P(\ulcorner\varphi\urcorner, n, \langle v_0, ..., v_n\rangle) \longleftrightarrow\; & f_{k(\ulcorner\varphi\urcorner,n)}(v_0, ..., v_n) \neq \varnothing\\ \longleftrightarrow\; & \exists\bar{k}(\bar{k} = k(\ulcorner\varphi\urcorner, n) \wedge E_{\bar{k}}(h) \neq \varnothing)\\ \longleftrightarrow\; & \forall\bar{k}(\bar{k} = k(\ulcorner\varphi\urcorner, n) \longrightarrow E_{\bar{k}}(h) \neq \varnothing) \text{ where } h(17 + N + i) = v_i\ (i \leq n).\end{aligned}$$

By the comment after Def. 8 concerning $E_k(h)$, the final lines here demonstrate Δ_1^M. For our applications to the hierarchies to come, this will be quite sufficient, as the extra assumption on HF will trivially hold.

If $\mathrm{HF} \notin M$ the argument is somewhat more delicate. One way to do this is to argue that we have codes of a sufficiently long initial segment of the recursive computation so that we may, in $\mathrm{GJ\Pi_1}$ alone, verify that "$y = k(u, n)$" for $u \in \mathrm{Fml}_0$, $n \in \omega$. See the Exercise for one method.

(ii) This then is straightforward, assuming (as we may by predicate calculus) that φ is written in prenex normal form, and quantifiers have been contracted by the use of pairing functions. The latter is unproblematic as M is closed under the rudimentary functions. QED

Exercise 5. Give some details of the formalisation of a Turing computation of a total recursive function k in GJ: Consider the usual Turing machine program as a finite list of quintuples with states actions *etc.* Represent an input n as a string of 1's as usual, and arrange the output, if any of a halting program as the number i of the cell C_i on which the machine halts. Let $\mathrm{Tur}(n, K, q)$ hold if $q \in \mathrm{Finseq}$ has domain K and $q_i \in (\mathrm{States} \times 2) \times \omega$, with $(q_i)_0$ representing the current state and character being read, and $(q_i)_1$ the current cell number; the transition from q_i to q_{i+1} is then a simple operation and can be written easily in a Δ_0 way, using a disjunction of the quintuples in the transition table. Then "$k(n) = j$" iff $\exists K, q($ q codes course of computation for K steps and halts in $C_j)$ is a $\Sigma_1^{\mathrm{GJ\,\Pi_1}}$ formulation; a $\Pi_1^{\mathrm{GJ\,\Pi_1}}$ is similar.]

We may thus assume that we have a recursive enumeration of the rudimentary functions, RudF, $\langle g_i |\, i < \omega\rangle$ such that if g_i occurs as a subfunction of g_j then $i < j$. Likewise if needed we can assume an enumeration $\langle r_i |\, i < \omega\rangle$ of Fml so that if r_i is a subformula of r_j then $i < j$.

Bibliography

[1] K. Devlin. *Constructibility*. Perspectives in Mathematical Logic. Springer Verlag, Berlin, Heidelberg, 1984.

[2] A. J. Dodd. *The Core Model*, volume 61 of *London Mathematical Society Lecture Notes in Mathematics*. Cambridge University Press, Cambridge, 1982.

[3] R.O. Gandy. Set theoretic principles for elementary syntax. In T. Jech, editor, *Axiomatic Set Theory*, volume 13 II of *Proceedings of Symposia in Pure Mathematics*, pages 103–126, Providence, Rhode Island, 1974. American Mathematical Society.

[4] R.B. Jensen and C. Karp. Primitive recursive set functions. In D. Scott, editor, *Axiomatic Set Theory*, volume 13 I of *Proceedings of Symposia in Pure Mathematics*, pages 143–167, Providence, Rhode Island, 1971. American Mathematical Society.

[5] A.R.D. Mathias. Weak systems of Gandy, Jensen and Devlin. In Joan Bagaria and Stevo Todorcevic, editors, *Set theory*, Trends in mathematics, pages 149–224. Birkh user Verlag, Basel, 2006. Centre de Recerca Matemàtica, Barcelona, 2003–2004.

[6] L. Stanley. Review of "Constructibility" by K. Devlin. *Journal of Symbolic Logic*, 52:864–868, 1987.