MORE ABOUT UNIFORM UPPER BOUNDS ON IDEALS OF TURING DEGREES\textsuperscript{1}

HAROLD T. HODES

Abstract. Let $I$ be a countable jump ideal in $\mathcal{P} = \langle \text{The Turing degrees}, \leq \rangle$. The central theorem of this paper is:

$a$ is a uniform upper bound on $I$ iff $a$ computes the join of an $I$-exact pair whose double jump $a'^{(2)}$ computes.

We may replace "the join of an $I$-exact pair" in the above theorem by "a weak uniform upper bound on $I$".

We also answer two minimality questions: the class of uniform upper bounds on $I$ never has a minimal member; if $\bigcup I = L_\alpha[A] \cap \omega_1$ for $\alpha$ admissible or a limit of admissibles, the same holds for nice uniform upper bounds.

The central technique used in proving these theorems consists in this: by trial and error construct a generic sequence approximating the desired object; simultaneously settle definitely on finite pieces of that object; make sure that the guessing settles down to the object determined by the limit of these finite pieces.

Fix recursive pairing and unpairing functions on $\omega$, such that $x = \langle (x)_0, (x)_1 \rangle$. For $f: \omega \rightarrow \omega$, let $(f)_x(y) = f(\langle x, y \rangle)$. If $\mathcal{F} \subseteq \omega_1$, $f$ parametrizes $\mathcal{F}$ iff $\mathcal{F} = \{(f)_x|x \in \omega \}$. We depart from standard practice and view Turing degrees as equivalence classes on $\omega_1$, not $\mathcal{P}(\omega)$, under $\equiv_T$. This has no importance; the following definitions could be rephrased to apply to Turing degrees as usually defined. All degrees in this paper are Turing degrees.

A degree $a$ is a uniform upper bound (u.u.b.) on a class $I$ of degrees iff some $f \in a$ parametrizes $\bigcup I$; $a$ is a weak u.u.b. iff some $f \in a$ parametrizes $\bigcup I \cap \omega_2$. $I$ is an ideal iff $I$ is downward closed under $\leq$ and closed under join. $I$ is a jump ideal iff $I$ is an ideal closed under jump. Where $I$ is an ideal, the pair $(b, c)$ is $I$-exact iff $I = \{d|d < b \land d < c\}$. Recent results of Shore imply that there is a degree-theoretic definition of the relation: $a$ is a u.u.b. on $I$, where $I$ is a countable jump ideal; it is obtained by encoding the analytic definition of a u.u.b. into degree-theoretic terms. The central result of this paper provides a more natural degree-theoretic definition of this relation.

**Theorem 1.** Where $I$ is a countable jump ideal: $a$ is a u.u.b. on $I$ iff there is an $I$-exact pair $(b, c)$, $b \vee c \leq a$ and $(b \vee c)^{(2)} \leq a^{(1)}$.

The technique used in proving the hard direction $(\Rightarrow)$ is then extended to answer further questions about u.u.b.s, some of which were raised in [2].

For $\mathcal{F} \subseteq \{(f)_x|x \in \omega \}$, $f$ is a subparametrization of $\mathcal{F}$. Let $f = f_0 \oplus \cdots \oplus f_{n-1}$ iff for all $x$, $f(x) = f_i((x)_1)$ if $(x)_0 = i < n$, $f(x) = 0$ otherwise.

Received May 10, 1981.

\textsuperscript{1}I wish to thank David Posner for an illuminating discussion which led to all these theorems.

© 1983, Association for Symbolic Logic

0022-4812/83/4802-0021/$02.70
GUESSING LEMMA. Let $I$ be an ideal of degrees, $f$ subparametrizes $\bigcup I$. There are two and three-place partial $f$-recursive functions $G$ and $H$ such that:

1. if $(f)(x_0 \oplus \cdots \oplus (f)(x_{m-1}) \in \bigcup I$ then $\lim n H(m, \langle x_0, \ldots, x_{m-1} \rangle, n)$ exists and if it is $z$, $(f)(x) = (f)(x_0 \oplus \cdots \oplus (f)(x_{m-1})$;
2. if $(f)(x) \in \bigcup I$, then $\lim n G(x, n)$ exists and if it is $z$, $(f)(x) = (f)(x)$.

(Here $\langle x_0, \ldots, x_{m-1} \rangle$ is a recursive coding of finite sequences from $\omega_0$ into $\omega$.)

PROOF. We construct $G$.

Let $g(x, u) = \begin{cases} 
\text{the least } t & \text{such that } \{u\}(f)(x(u)) \text{ converges in } t \text{ steps} \\
0 & \text{otherwise.} 
\end{cases}$

$\lambda u. g(x, u) \equiv_T (f)(x)$. Thus if $(f)(x) \in \bigcup I$, $\lambda u. g(x, u) \in \bigcup I$. Let $h$ be a non-decreasing function which eventually dominates each member of $\bigcup I$, $h \leq_T f$: for example, $h(z) = \max_{u \leq z} (f)(u)$. We shall say that $z$ is a candidate for $x$ at step $n$ iff for every $u < n$:

- if $\{u\}(f)(x(u))$ converges in $h(u) + n$ steps,
- $\{u\}(f)(x(u))$ if not.

Given $x$, select $u_0$ such that for all $u \geq u_0$, $h(u) \geq g(x, u)$. Let $n_0 = \max\{g(x, u) : u < u_0\}$. For $n \geq n_0$, if $z$ is a candidate for $x$ at step $n$, $(f)(x) \uparrow n = (f)(x)$, since for all $u$, $g(x, u) < h(u) + n$. Let $G(x, n) = \lambda z. \{x\}(f)(x) = (f)(x)$ is a candidate for $x$ at step $n$. Suppose that $(f)(x) \in \bigcup I$, $z_0$ is the least $z$ such that $(f)(x) = (f)(x)$, and $n_1$ is the least $n$ such that for each $z < z_0$, $(f)(x)(n) \neq (f)(x)(n)$ for some $n < n_1$. Then for $n \geq \max(n_0, n_1)$ and any $z < z_0$, $z$ is not a candidate for $x$ at step $n$. But $z_0$ is one as of step $n$. So $G(x, n) = z_0$ for such $n$. The construction of $H$ is easier and we omit it. Q.E.D.

We note the following. Suppose $f$ parametrizes $\bigcup I \cap \omega_2$ and $0 \in I$. $\deg(f)$ is a u.u.b. on $I$ iff there is a $G \leq_T f$ as above which guesses at the location of jumps. This is easy to prove.

LEMMA 1. If $I$ is a set of degrees and $f$ is a function such that for every $g \in \bigcup I$ there is an $e$ such that $g = \ast (f)(e)$, and for every $e$, $(f)(e) \in \bigcup I$, then $\deg(f)$ is a u.u.b. on $I$.

PROOF. Let Seq be the set of sequence numbers, letting $s = \langle (s)_0, \ldots, (s)_{h(s)-1} \rangle$.

Let

$$(f)(s)_x = \begin{cases} 
(f)(x) & \text{if } s \notin \text{Seq}, \\
(s)_x & \text{if } s \notin \text{Seq} \& x < h(s), \\
(f)(x) & \text{if otherwise.} 
\end{cases}$$

$\hat{f} \leq_T f$ and $\hat{f}$ parametrizes $\bigcup I$. Since the class of u.u.b.’s on $I$ is closed upwards, $\deg(f)$ is a u.u.b. on $I$. Q.E.D.

PROOF OF THEOREM 1 (‘). Suppose $(b, c)$ is $I$-exact, $b \lor c \leq a$ and $(b \lor c)(2) \leq a$, $A \in a$, $B \in b$, $C \in c$. Since $(B \oplus C)(2) \leq_T A$ (1), recursively in $A$ we may guess
at the truth of $\forall \exists$ sentences about $B$ and $C$ so that in the limit these guesses are correct. Let $f$ be such that

$$f(x) = \begin{cases} 0 & \text{if for some } t \geq \max(x, n), \text{ the } t\text{th guess is} \\ \{e_1\}_{B}(y) \text{ is undefined} \\ \{e_2\}_{B}(y) \text{ is undefined} \\ \{e_1\}_{B}(y) \neq \{e_2\}_{B}(y) \text{, and either } t = \\ \max(x, n) \text{ or } \{e_1\}_{B}(x) \text{ is undefined}; \\ \{e_1\}_{B}(x) \text{ otherwise.} \\
\end{cases}$$

$f \leq_{f} A$. In the otherwise case, $\{e_1\}_{B}(x)$ is defined, since in the limit our guesses at whether $\forall (\forall y)(\forall z) \{e_1\}_{B}(y)$ is defined $\& \{e_1\}_{B}(y) = \{e_2\}_{B}(y)$ are right. If $\{e_1\}_{B}$ is total and $\{e_1\}_{B} = \{e_2\}_{B}$, then $f_{\langle e_1, e_2 \rangle, n} = * \{e_1\}_{B}$; otherwise $f_{\langle e_1, e_2 \rangle, n} = * \lambda x. 0$.

By Lemma 1, $a$ is a u.u.b. on $L (\equiv)$. Let $Str$ be the set of finite strings of 0’s and 1’s, coded into $\omega$. For $\sigma, \tau \in Str,$ $\sigma \tau$ is the concatenation of $\sigma$ and $\tau$; $\sigma \leq \tau$ iff $\sigma$ extends $\tau$; $\sigma < \tau$ iff $\sigma \leq \tau$ and $\sigma \neq \tau$. $P$ is a tree iff $P : Str \rightarrow Str$ and for all $\sigma, \tau \in Str$, if $\tau \leq \sigma$ then $P(\sigma) \leq P(\sigma)$. A tree $P$ is perfect iff for all $\sigma \in Str$, $P(\sigma - 1)$ is strictly left of $P(\sigma - 1)$ in the lexicographic ordering of $Str$. For $C \in \omega^2$, $C \leq \sigma$ if $\sigma$ codes an initial segment of $C$. Let $B \in [P]$ iff $B$ is a branch of $P$ iff for some $C \in \omega^2$, $B = \lim\{P(\sigma) \mid C \leq \sigma\}$. $P$ is uniformly recursively pointed iff for some $e$: for all $B \in [P]$, $P = \{e\}_{B}$. We code $B \in \omega^2$ into a tree $P$, yielding a tree $Code(P, B)$, as follows:

$Code(P, B)(\langle \rangle) = P(\langle \rangle),$$\quad Code(P, B)(\sigma) = P(\langle B(0), (\sigma)_0, \ldots, (\sigma)_{lh(\sigma) - 1} \rangle)$ for $lh(\sigma) \geq 1$.

Abusing notation, we write $Code(P, f)$ for $Code(P, graph(f))$.

A condition is a pair $\langle P, Q \rangle$ of uniformly recursively pointed perfect trees belonging to $\bigcup I$ such that $P \equiv_{f} Q$. $P$ is a subtree of $Q$ iff for all $\sigma \in Str$, $P(\sigma) \leq Q(\sigma)$. Where $\langle P, Q \rangle$ and $\langle R, S \rangle$ are conditions, $\langle P, Q \rangle$ extends $\langle R, S \rangle$ iff $P$ and $Q$ are subtrees of $R$ and $S$, respectively. $Code(\langle P, Q \rangle, f) = \langle Code(P, f), Code(Q, f) \rangle$.

For $f \in \bigcup I$, this is a condition.

Let $Str(l) = \{\sigma \mid \sigma \in Str \& lh(\sigma) \leq l\}$. A function $P : Str(l) \rightarrow Str$ is a pretree iff $P$ fulfills the definition of a perfect tree, except with domain restricted to $Str(l)$; $l$ is the height of $P = ht(P)$. If $P$ is a perfect tree, $P \upharpoonright Str(l)$ is a pretree of height $l$. If for each $l < \omega$, $P_l$ is a pretree of height $l$ and $P_l \subseteq P_{l + 1}$, $\bigcup P_l$ is a perfect tree. A precondition of height $l$ is a pair of pretrees of height $l$. Since pretrees and preconditions are finite objects, we code them into $\omega$. A pretree $P$ is a subpretree of a tree or pretree $R$ iff for each $\sigma \in dom(P)$ there is a $\tau \in dom(R)$, $\tau \leq \sigma$ and $P(\sigma) = R(\tau)$. If $P$ is a subpretree of $R$ and $\sigma \in dom(P)$, $\sigma \in dom(R)$ and $P(\sigma) \leq R(\sigma)$; if, furthermore, $R$ is a pretree, $ht(P) \leq ht(R)$. $\langle P, Q \rangle$ is a subprecondition of a condition or precondition $\langle R, S \rangle$ iff $P$ and $Q$ are subtrees of $R$ and $S$, respectively. Suppose that for each $l < \omega$, $\langle P_l, Q_l \rangle$ is a subprecondition of a condition or precondition $\langle R, S \rangle$, $l = ht(\langle P_l, Q_l \rangle)$, $\langle P_{l + 1}, Q_{l + 1} \rangle$ is a subprecondition of $\langle P_l, Q_l \rangle$, and $\langle P_l, Q_l \rangle_{\upsilon < \omega}$ is recursive in $R \uplus S$; then $lim P_l, Q_l \rangle = \bigcup P_l, Q_l \rangle$ is a condition extending $\langle R, S \rangle$.

For $P$ a pretree and $B \in \omega^2$, we may code as much of $B$ as possible into $P$, letting:

$Code(P, B)(\langle \rangle) = P(\langle \rangle),$$\quad Code(P, B)(\sigma) = P(\langle B(0), (\sigma)_0, \ldots, (\sigma)_{lh(\sigma) - 1} \rangle)$, for $lh(\sigma) \geq 1$.

Note that if $ht(P) = 2l$ or $2l + 1$, $Code(P, B)$ has height $l$. We define
"Code(P, f)" and Code(⟨P, Q⟩, f) where ⟨P, Q⟩ is a precondition, as one would expect.

For P a tree or pretree and σ ∈ Str, we shall say that σ is on P iff for some τ ∈ dom(P), P(τ) ⊑ σ. Full is the tree id ↑ Str. Where P is a tree or pretree, Full(P, σ) is the tree or pretree determined by Full(P, σ)(τ) = P(σ−τ). Note that if P is a pretree of height l, Full(P, σ) is totally undefined, and so technically not a pretree, if l < lh(σ).

Fix a listing ⟨ψj⟩j<ω of all primitive recursive relations on w2 × w2 × w × w. Introducing "B" and "C" as uninterpreted predicate constants, let ϕj be "(∃x)¬(∃y)ψj(B, C, x, y)." We now define forcing, for ⟨P, Q⟩ a condition.

⟨P, Q⟩ ⊩ ¬ϕj iff for all ⟨B, C⟩ ∈ [P] × [Q], ⟨B, C⟩ ⊨ ¬ϕj;
⟨P, Q⟩ ⊩ ϕj iff for some n for all ⟨B, C⟩ ∈ [P] × [Q],
⟨B, C⟩ ⊩ ¬(∃y)ϕj(B, C, n, y).

[3] contains a proof of the crucial density theorem: any condition extends to a condition deciding ϕj. Implicit in that proof is the construction of a function force(j, ⟨P, Q⟩) with domain ≤ ω such that, letting force(j, ⟨P, Q⟩)(l) = ⟨Pl⟩ ⊩ QYD):

(1) force(j, ⟨P, Q⟩)(l) is, if defined, a subprecondition of ⟨P, Q⟩ of height l;
(2) if l + 1 ∈ dom(force(j, ⟨P, Q⟩)),
force(j, ⟨P, Q⟩)(l) = ⟨Pl⟩ ⊩ Str(l), Q(l + 1) ⊩ Str(l);
(3) for l ∈ dom(force(j, ⟨P, Q⟩)), σ, τ strings of length l, there is a yσ,τ such that
ϕj(Pl)(σ), Q(l)(τ, l, yσ,τ). (Following a standard convention, "ϕj(σ, τ, x, y)" means "For all B ⊑ σ, C ⊑ τ, ϕj(B, C, x, y).") To compute force(j, ⟨P, Q⟩)(0), we search for strings σ and τ of the same length and for a yσ,τ so that ϕj(P(σ), Q(τ), 0, yσ,τ), and let P(0)(⟨ ⟩) = P(σ), Q(0)(⟨ ⟩) = Q(τ). Call these chosen σ and τ, if they exist, ⟨ ⟨ ⟩ and ⟨ ⟩, respectively. Now suppose that force(j, ⟨P, Q⟩)(l) = ⟨Pl⟩ ⊩ Q(l) has been computed; for ρ ∈ Str(l), we suppose that ρ’ and ρ” have been defined, P(l)(ρ) = P(ρ’), Q(l)(ρ) = Q(ρ”). We now try to compute P(l + 1) and Q(l + 1) on all of Str(l + 1). By our computation of P(l) and Q(l) (2), it suffices to do this for strings of length l + 1. Let σ1, ..., σ2l+1, τ1, ..., τ2l+1 be two lists of all strings of length l + 1. We search for strings σ’1, ..., σ’2l+1, τ’1, ..., τ’2l+1 all of the same length, and for witnesses yσ’i,τ’k, i, k ∈ {1, ..., 2l+1}, such that for σi = σ̄−⟨m⟩ and τk = τ̄−⟨n⟩, σi ≤ σ̄−⟨m⟩ and τk ≤ τ̄−⟨n⟩, and ϕj(P(σi), Q(τk), l + 1, yσ’i,τ’k); we let P(l + 1)(σi) = P(σi), Q(l + 1)(τk) = Q(τk). For details on this search, see [3]. This search is recursive in P ⊗ Q. So force(j, ⟨P, Q⟩) is partial recursive in P ⊗ Q, uniformly in j and ⟨P, Q⟩, by the procedure outlined. "Force(j, ⟨P, Q⟩)(l) is defined in q steps" means that according to the procedure just outlined, that computation converges in q steps. If force(j, ⟨P, Q⟩) is total, limjforce(j, ⟨P, Q⟩)(l) = ⟨Pl⟩ ⊩ Q(l), l ∈ dom(force(j, ⟨P, Q⟩)) is a condition forcing ¬ϕj.

On the other hand, suppose force(j, ⟨P, Q⟩) is not total. Call ⟨l, σ, τ⟩ a j-witness for ⟨P, Q⟩ iff σ, τ ∈ Str, lh(σ) = lh(τ), and ⟨Full(P, σ), Full(Q, τ)⟩ ⊩ ¬(∃y)ψj(B, C, l, y). We now find a j-witness for ⟨P, Q⟩. Let l be the least l ∈ dom(force(j, ⟨P, Q⟩)). If l = 0, let σ = τ = ⟨ ⟩. If l = x + 1, let ⟨σi, τk⟩ be the least pair selected from the lists σi, ..., σ2i, τ1, ..., τ2k, for which we cannot find
appropriate $\sigma_i'$, $\tau_k'$ and $\nu_{z_i}$ and $\nu_{x_i}$. Letting $\sigma_i = \sigma^{0\sim} <n>$, $\tau_k = \tau^{0\sim} <m>$, let $\sigma = \sigma^{0\sim} <n>$, $\tau = \tau^{0\sim} <m>$. $\langle l, \sigma, \tau \rangle$ is easily seen to be a $j$-witness for $\langle P, Q \rangle$. Notice that $lh(\sigma) = lh(\tau)$, since in defining $P(x)$ and $Q(x)$ we required that $lh((\sigma^0)^\sim) = lh((\tau^0)^\sim)$. We have just described a procedure recursive in $(P \oplus Q)^{(1)}$ which halts iff $force(j, \langle P, Q \rangle)$ is partial, and, if it halts, delivers a $j$-witness for $\langle P, Q \rangle$. Call this procedure $Wit(j, \langle P, Q \rangle)$.

The construction of $force(j, \langle P, Q \rangle)(0)$, and then of $force(j, \langle P, Q \rangle)(l + 1)$ given $force(j, \langle P, Q \rangle)(l)$, proceeds by working down $P$ and $Q$, thinking of trees as growing downwards. Thus we may extend our definition of $force(j, \langle P, Q \rangle)$ to apply to the case in which $\langle P, Q \rangle$ is a precondition. In this case, $\text{dom}(force(j, \langle P, Q \rangle))$ is finite, and in fact, $\leq \text{ht}(\langle P, Q \rangle)$.

Fix $f \in a$, parametrizing $\bigcup I$. We wish to construct $B, C \in \omega^2$, $\langle \text{deg}(B), \text{deg}(C) \rangle$ $I$-exact, $(B \oplus C)^{(2)} \leq_T f^{(1)}$ and $B \oplus C \leq_T f$.

A natural strategy suggests that we try to construct a sequence of conditions $\{\langle P_j, Q_j \rangle\}_{j < \omega}$ and an auxiliary sequence $\{\langle x_j, \sigma_j, \tau_j \rangle\}_{j < \omega}$ such that:

1. $P_0 = Q_0 = \text{Full}$;
2. for all $j$:
   a. if $x_j \geq 0$ then $\langle x_j, \sigma_j, \tau_j \rangle = Wit(j, \langle P_{2j}, Q_{2j} \rangle)$ and $\langle P_{2j+1}, Q_{2j+1} \rangle = \langle \text{Full}(P_{2j}, \sigma_j), \text{Full}(Q_{2j}, \tau_j) \rangle$;
   b. if $x_j = -1$, $\sigma_j = \tau_j = \langle \rangle$ and $force(j, \langle P_{2j}, Q_{2j} \rangle)$ is total and $\langle P_{2j+1}, Q_{2j+1} \rangle = \lim_I force(j, \langle P_{2j}, Q_{2j} \rangle)(l)$;
3. for all $j$, $\langle P_{2j+2}, Q_{2j+2} \rangle = \text{Code}(\langle P_{2j+1}, Q_{2j+1} \rangle, (f)_j)$.

Then we shall let $\{B\} = \bigcap_I P_j$, $\{C\} = \bigcap_I Q_j$. Choice of $\langle P_{2j+2}, Q_{2j+2} \rangle$ insures that $(f)_j \leq_T B$ and $(f)_j \leq_T C$. The genericity of the sequence of conditions insures that if $g \leq_T B$ and $g \leq_T C$, $g \in \bigcup I$.

We also want our construction to be recursive in $f$. But choice of $\langle P_{2j+1}, Q_{2j+1} \rangle$ or, equivalently, of $\langle x_j, \sigma_j, \tau_j \rangle$, depends on facts about $\langle P_{2j} \oplus Q_{2j} \rangle^{(2)}$ which cannot be decided uniformly in $j$ and recursively in $f$. A further difficulty appears when we specify the sense in which we would like $\{\langle P_j, Q_j \rangle\}_{j < \omega}$ to be recursive in $f$. We want an $f$-recursive function $j \mapsto \langle n_j, m_j \rangle$ such that $P_j = (f)_n$, $Q_j = (f)_m$, and such a function may not exist. Instead we proceed by guessing, recursively in $f$ at the previously described construction.

For $x \geq 1$, let $d(x) = y$ iff $x = 2y + 1$ or $x = 2y + 2$. At stage $i$ of our construction we will have a number $z_i \geq 1$ and, for each $j \leq z_i$, a guess $\langle P_j, Q_j \rangle$ at $\langle P_j, Q_j \rangle$, and, for each $j \leq d(z_i)$, guesses $x_j, \sigma_j$ and $\tau_j$ at $x_j, \sigma_j$ and $\tau_j$. $P_j$ and $Q_j$ are functions, $\text{dom}(P_j) = \text{dom}(Q_j) \leq \omega$ such that, letting $\langle P_j, Q_j \rangle(l) = \langle P_j(l), Q_j(l) \rangle$, $\langle P_j, Q_j \rangle(l)$ is, if defined, a precondition of $\text{ht} l$ such that:

1. $\langle P_0, Q_0 \rangle(l) = \langle \text{Full} \uparrow \text{Str}(l), \text{Full} \uparrow \text{Str}(l) \rangle$;
2. for all $j \leq d(z_i)$, if $x_j \geq 0$, 

This content downloaded from 132.174.252.179 on Wed, 16 Feb 2022 02:29:07 UTC
All use subject to https://about.jstor.org/terms
\[ \langle P_{2j+1}, Q_{2j+1} \rangle \approx \langle \text{Full}(P_{2j}(k + l), \sigma_j), \text{Full}(Q_{2j}(k + l), \tau_j) \rangle, \]

where \( \text{lh}(\sigma_j) = \text{lh}(\tau_j) = k; \)

if \( x_j^1 = -1, \sigma_j^1 = \tau_j^1 = \varnothing \) and

\[ \langle P_{2j+1}, Q_{2j+1} \rangle \approx \text{force}(j, \langle P_{2j}, Q_{2j} \rangle(1'))(l) \]

for an \( l' \in \text{dom}(\langle P_{2j}, Q_{2j} \rangle), \) but large enough for the right-hand side to be defined, if such there be;

\( (3') \) for all \( 2j + 2 \leq z_i, \)

\[ \langle P_{2j+2}, Q_{2j+2} \rangle(l) \approx \text{Code}(\langle P_{2j+1}, Q_{2j+1} \rangle(2l), (f)_j). \]

For reasons to appear shortly, we need to modify this outline in one respect.

In the sequence described by (1)–(3) we shall add, between consecutive conditions \( \langle P_j, Q_j \rangle \) and \( \langle P_j+1, Q_j+1 \rangle, \) an intermediate condition \( \langle P^*_j, Q^*_j \rangle, \) determined by strings \( \delta_j \) and \( \varepsilon_j \) of equal length, so that:

\( (4^*) \) for all \( j, \)

\[ \langle P^*_j, Q^*_j \rangle = \langle \text{Full}(P_j, \delta_j), \text{Full}(Q_j, \varepsilon_j) \rangle, \]

with (2) and (3) revised to (2*) and (3*), (2*) saying that \( \langle P_{2j+1}, Q_{2j+1} \rangle \) is formed from \( \langle P^*_j, Q^*_j \rangle \) in the way in which (2) says it is formed from \( \langle P_{2j}, Q_{2j} \rangle, \) and

(3*) saying that \( \langle P_{2j+2}, Q_{2j+2} \rangle \) is formed from \( \langle P^*_j, Q^*_j \rangle \) in the way in which (3) says it is formed from \( \langle P_{2j}, Q_{2j} \rangle. \)

In our guessing construction, at stage \( i \) for all \( j < z_i, \) we shall have guesses \( \delta_j^i \) and \( \varepsilon_j^i \) at \( \delta_j \) and \( \varepsilon_j \) and guesses \( \langle P^*_j, Q^*_j \rangle \) at \( \langle P^*_j, Q^*_j \rangle \) given by:

\( (4^*) \) for \( j < z_i, \)

\[ \langle P^*_j, Q^*_j \rangle(l) \approx \langle \text{Full}(P_j(k + l), \delta_j^i), \text{Full}(Q_j(k + l), \varepsilon_j^i) \rangle, \]

for \( k = \text{lh}(\delta_j^i) = \text{lh}(\varepsilon_j^i). \)

(2') and (3') are now revised to (2*) and (3*), following the obvious analogy with (2*) and (3*).

If our guess converges appropriately, we shall have \( (B \oplus C)^{(2)} \leq_T f^{(1)}. \)

To insure that \( B \oplus C \leq_T f, \) we must supplement the guessing procedure just described with a nonguessing process such that for each \( n \) we can \( f \)-recursively find a stage \( i \) which definitely settles the questions "\( n \in B?\)" and "\( n \in C?\)".

To this end we construct sequences \( \{\beta_i\}_{i=\omega} \) and \( \{\gamma_i\}_{i=\omega} \) of strings \( \beta_{i+1} \leq \beta_i, \gamma_{i+1} \leq \gamma_i, \) and we make sure that \( B = \lim \beta_i, \) \( C = \lim \gamma_i. \) \( \beta_i \) and \( \gamma_i \) will be fixed at stage \( i \) on the basis of our guesses as of stage \( i. \) But thereafter any further guesses, including revisions of guesses on the basis of which \( \beta_i \) and \( \gamma_i \) were fixed, must honor the commitments that \( B < \beta_i \) and \( C < \gamma_i. \) This is where \( \delta_j^i \) and \( \varepsilon_j^i \) come in; when we make a decision at stage \( i \) about what \( \langle P_{j+1}, Q_{j+1} \rangle \) looks like, we shall choose \( \delta_j^i, \varepsilon_j^i \) to "protect" \( \beta_i \) and \( \gamma_i; \) that is, we shall try to make sure that \( \beta_{j+1} \leq P^j(< >) \leq P_j(1)(< >) \leq P_{j+1}(1)(< >) \leq Q_{j+1}(1)(< >) \leq \gamma_i. \) To carry all this out, at stage \( i \) we shall actually have to compute, for each \( j = z_i, \)

\[ \langle P_j, Q_j \rangle(k_j^i) \]

for a certain \( k_j^i. \)

To this end, we introduce functions \( l_j^i \) defined by \( l_j^i \leq z_i \) and \( l_j^i < z_r. \) Intuitively, \( l_j^i(q) \) is the largest \( l \) such that we can compute \( \langle P_j, Q_j \rangle(l) \) in \( \leq q \) steps; \( l_j^i* \) is the largest \( l \) such that we can compute \( \langle P_j, Q_j* \rangle(l) \) in \( \leq q \) steps. \( l_j^i \) or \( l_j^i* \) may be undefined on
an initial segment of ω, since it can take a while even to compute \(<P_j^*, Q_j^*> (0)\) or \(<P_j^*, Q_j^*> (0)\). But if defined, \(l_j^q(q)\) is defined and \(\leq l_j^q(q)\); similarly for \(l_j^q(q)\). If \(l_j^q(q)\) is defined, \(<P_j^*, Q_j^*> (l_j^q(q))\) is a subprecondition of \(<P_j^*, Q_j^*> (l_j^q(q))\) with \(l_j^q(q)\) defined; if \(l_j^q(q)\) is defined, \(<P_j^*, Q_j^*> (l_j^q(q))\) is a subprecondition of \(<P_j^*, Q_j^*> (l_j^q(q))\), with \(l_j^q(q)\) defined. Furthermore, for \(j < z_i\), if \(\lim_{q \to \omega} l_j^q(q) = \omega\) then \(\lim_{q \to \omega} l_j^q(q) = \omega\); for \(2j + 1 < z_i\), if \(\lim_{q \to \omega} l_j^q(q) = \omega\) then \(\lim_{q \to \omega} l_j^q(q) = \omega\); for \(2j < z_i\), if \(\lim_{q \to \omega} l_j^q(q) = \omega\) then: if \(x_j^q < 0\), \(\lim_{q \to \omega} l_j^q(q) = \omega\); if \(x_j^q = -1\), \(\lim_{q \to \omega} l_j^q(q) = \omega\) iff force\((j, <P, Q>)\) is total, for \(<P, Q> = \lim_{q \to \omega} <P_{j*}, Q_{j*}> (l_j^q(q))\).

Our informal description of \(l_j^q\) and \(l_j^q\) could serve as a definition of these functions, but we offer definitions anyway:

\[ l_j^q(q) = q; \]
\[ l_j^q(q) \approx l_j^q(q) - \gamma(l_j^q(q)); \]
if \(x_j^q = -1\), \(l_j^q(q) = \omega\) if \(x_j^q = -1\), \(l_j^q(q) = \omega\).

We shall arrange our construction so that at each stage \(i:\)

(1.ii) \(l_j^q(q)\) is defined, with \(\beta_i\) on \(P_j^*(l_j^q(q))\) and \(\gamma_i\) on \(Q_j^*(l_j^q(q))\).

In addition to the sequences so far described, we also need a sequence \(\{<nj, mj>\}_{j<\omega}\) such that:

\[ \text{(5) for all } j, \quad <P_j, Q_j> = <(f)_{nj}, (f)_{mj}>. \]

We shall also need guess \(<nj, mj>\) at \(<nj, mj>\) for \(j \leq z\). Let \([n, m, \delta, \varepsilon]\) abbreviate \(<\text{Full}(f)_n, \delta, \text{Full}(f)_m, \varepsilon>\). For \(2j + 1 \leq z\), let \(2j + 1\) have property 1 at stage \(i\) iff \([nj, mj, \delta_j, \varepsilon_j]\) is a condition, and: if \(x_j^q > 0\),

\[ <x_j^q, \delta_j^q, \varepsilon_j^q> = \text{Wit}(j, [nj, mj, \delta_j, \varepsilon_j]); \]
if \(x_j^q = -1\), \(\text{Wit}(j, [nj, mj, \delta_j, \varepsilon_j])\) is undefined. Note that “\(2j + 1\) has property 1 at stage \(i\)” is \(\Sigma_3^0\) in \((f_{nj}, \delta_j^q, (f)_{mj})\). It would be nice at stage \(i\) to have all \(2j + 1 \leq z\), with property 1. But to keep the construction recursive in \(f\) we can only guess at whether a given \(2j + 1\) has property 1. We do this by asking the question of our \(g(i)\)th guess at \((f)_{nj} \oplus (f)_{mj} (3)\) namely

\[ (f)_{G(\text{G}(\text{H}(m_{nj}^q, m_{mj}^q, g(i)), g(i)), g(i)), g(i))}. \]

We content ourselves with insuring that at each stage \(i:\)

(2.1) for each \(2j + 1 \leq z\) our \(g(i)\)th guess at \((f)_{nj} \oplus (f)_{mj} (3)\) says that \(2j + 1\) has property 1 at stage \(i\).

This can be checked recursively in \(f\).

For \(j \leq z\), let \(j\) have property 2 at stage \(i\) iff \(\lim_{q \to \omega} <P_j^*, Q_j> (l) = <(f)_{nj}, (f)_{mj}>\), which is a condition. Again, “\(j\) has property 2 at stage \(i\)” is \(\Sigma_3^0\) in \((f)_{nj} \oplus (f)_{mj})\).
It would be nice to have all \( j \leq z_i \) with property 2 at stage \( i \), so that our guesses at \( \langle n_j, m_j \rangle \) accurately reflect our guesses at \( \langle P_j, Q_j \rangle \). But, to keep the construction recursive in \( f \), the best we can do is to insure that at each stage \( i \):

\[(3.i) \text{ for each } j \leq z_i,
\]

\[\langle P_j, Q_j \rangle (l(j(g(i)))) = \langle (f)n_j \uparrow \text{Str}(l(j(g(i)))), (f)m_j \uparrow \text{Str}(l(j(g(i)))) \rangle.\]

Checking this will be recursive in \( f \).

After such extensive previewing, the presentation of the construction may, at least, be brief.

**Stage 0.** \( z_0 = 0, g(0) = 0, \beta_0 = \gamma_0 = \langle \rangle; \) for all \( l, P_0(l) = Q_0(l) = \text{Full} \uparrow \text{Str}(l); \)

select \( \langle n_0, m_0 \rangle \) so that \( \langle f \rangle n_0 = \langle f \rangle m_0 = \text{Full}. \)

**Stage \( i + 1 \).** Suppose we already have \( z_i, g(i), \{\langle P_j, Q_j \rangle\}_{j < z_i}, \{\langle \delta_j, \varepsilon_j \rangle\}_{j < d(z_i)}, \{\langle x_j, \sigma_j, \tau_j \rangle\}_{j < d(z_i)} \), \( \beta_i \) and \( \gamma_i \), with \( (1.i)-(3.i) \) all true. For \( 2j + 1 \leq z_i \), let \( 2j + 1 \) be 1-bad at \( (i, q) \) iff our \( (g(i) + q + 1) \)st guess at \( \langle (f)n_{2j}, (f)m_{2j} \rangle \) says that \( 2j + 1 \) lacks property 1 at stage \( i \). For \( j < z_i \), \( j \) is 2-bad at \( (i, q) \) iff

\[\langle P_j, Q_j \rangle (l_1(g(i) + q + 1)) \neq \langle (f)n_j \uparrow \text{Str}(l(j(g(i) + q + 1))), (f)m_j \uparrow \text{Str}(l(j(g(i) + q + 1))) \rangle.\]

Let \( \langle \delta, \varepsilon \rangle \) be a \( q \)-combination for \( 2j \leq z_i \) iff \( \text{lh}(\delta) = \text{lh}(\varepsilon) \leq l_{2j} (g(i) + q + 1) = l \)

and

(6) either:

(a) our \( (g(i) + q + 1) \)st guess at \( \langle (f)n_{2j}, (f)m_{2j} \rangle \) says that

\[\text{Wit}(j, [n_{2j}, m_{2j}] \delta, \varepsilon) = \langle x, \sigma, \tau \rangle \]

in \( l \leq g(i) + q + 1 \) steps for some \( \langle x, \sigma, \tau \rangle \) with \( \text{lh}(\delta) + \text{lh}(\sigma) \leq l \); or

(b) our \( (g(i) + q + 1) \)st guess at \( \langle (f)n_{2j}, (f)m_{2j} \rangle \) is undefined, and force \( (j, \langle P, Q \rangle)(0) \) is defined in \( l \leq g(i) + q + 1 \) steps for

\[\langle P, Q \rangle = \langle \text{Full}(P_{2j}(l), \delta), \text{Full}(Q_{2j}(l), \varepsilon) \rangle.\]

Whether \( \langle \delta, \varepsilon \rangle \) is a \( q \)-combination, in fact whether there is a \( q \)-combination for a given \( 2j \), is decidable recursively in \( f \). We shall say that \( q \) changes the primary guess at \( 2j + 1 \leq z_i \) iff: for \( k < 2j + 1 \), \( k \) is neither 1-bad nor 2-bad at \( (i, q) \); \( 2j + 1 \) is 1-bad at \( (i, q) \); and

(7) there is a \( q \)-combination \( \langle \delta, \varepsilon \rangle \) such that

\[P_{2j}(l_{2j}(g(i) + q + 1)) \delta \leq \beta_i\]

and

\[Q_{2j}(l_{2j}(g(i) + q + 1)) \varepsilon \leq \gamma_i.\]

We shall say that \( q \) changes the secondary guess at \( j \leq z_i \) iff: for all \( k < j, k \) is neither 1-bad nor 2-bad at \( (i, q) \); \( j \) is 2-bad but not 1-bad at \( (i, q) \); and

(8) there are strings \( \beta \) and \( \gamma \) on \( P_j(l_j(g(i) + q + 1)) \) and \( Q_j(l_j(g(i) + q + 1)) \), respectively, \( \beta \leq \beta_i \) and \( \gamma \leq \gamma_i \). We shall say that \( q \) creates a guess at \( z_i + 1 = z \) iff for all \( j \leq z_i, j \) is neither 1-bad nor 2-bad at \( (i, q) \), and

This content downloaded from 132.174.252.179 on Wed, 16 Feb 2022 02:29:07 UTC
All use subject to https://about.jstor.org/terms
(9) there are strings $\delta$ and $\varepsilon$ such that

\begin{equation}
 P_{\delta}(l_{\varepsilon}(g(i) + q + 1))(\delta) \leq \beta_i
\end{equation}

and

\begin{equation}
 Q_{\delta}(l_{\varepsilon}(g(i) + q + 1))(\varepsilon) \leq \gamma_i;
\end{equation}

(9.2) if $z = 2j + 1$, $(\delta, \varepsilon)$ is a $q$-combination for $2j$.

**Lemma 2.** There is a $q$ which either changes or creates a guess.

**Proof.** Let $j = \text{the least } j \leq z_i \text{ which lacks either property 1 or 2, if there is one}; j = z_i + 1 \text{ otherwise}. \text{ If } j \text{ lacks property 1, we find a } q \text{ changing the primary guess at } j; \text{ if } j \text{ has property 1 but not property 2, we find a } q \text{ changing the secondary guess at } j; \text{ if } j = z_i + 1, \text{ we find a } q \text{ creating a guess at } j. \text{ Consider the first situation. Suppose that for } q \geq q_0, \text{ our } (g(i) + q + 1)\text{st guess at } (f_{m_{2j}}(g(i) + q + 1))^{(3)} \text{ is correct for all } 2j \leq j. \text{ So for } q \geq q_0, \text{ all } k < j \text{ are neither 1-bad nor 2-bad at } (i, q), \text{ and } j \text{ is 1-bad at } (i, q). \text{ For } j < j, \text{ } \lim_{i \to j} f_i(l_i) = \omega. \text{ If not, let } j \text{ be the least counterexample; by remarks preceding the definition of } l_j, \text{ } j = 2j' + 1, \text{ and } force(j', \lim_{i < j} (f_{m_{2j'}}(g(i)))) \text{ is partial; so } \text{ Wit}(j', [n_{2j}, m_{2j}/\delta_{2j'}, \varepsilon_{2j}]) \text{ is defined, and } j \text{ lacks property 1}; \text{ contradiction with } j < j. \text{ Now let } j = 2j + 1. \text{ For sufficiently large } q \text{ we may increase } l = l_{2j}(g(i) + q + 1) \text{ large enough to find } (\delta, \varepsilon), \text{ } P_{2j}(l)(\delta) \leq \beta_i \text{ and } Q_{2j}(l)(\varepsilon) \leq \gamma_i, \text{ } lh(\delta) = lh(\varepsilon) = l. \text{ Note that}

\[ [n_{2j}, m_{2j}/\delta_{2j}, \varepsilon] = \lim_{i \to \infty} \langle \text{Full}(P_{2j}(l), \delta), \text{Full}(Q_{2j}(l), \varepsilon) \rangle. \]

If \text{ Wit}(j, [n_{2j}, m_{2j}/\delta, \varepsilon]) \text{ is defined, then for } q \geq q_0 \text{ our } (g(i) + q + 1)\text{st guess at } (f_{m_{2j}}(g(i) + q + 1))^{(3)} \text{ says it is}; \text{ so for sufficiently large } q \geq q_0, \text{ it truthfully says that } \text{ Wit}(j, [n_{2j}, m_{2j}/\delta, \varepsilon]) = \langle x, \sigma, \tau \rangle \text{ in } \leq g(i) + q + 1 \text{ steps, and } lh(\sigma) + lh(\delta) \leq l_{2j}(g(i) + q + 1). \text{ On the other hand, if } \text{ Wit}(j, [n_{2j}, m_{2j}/\delta, \varepsilon]) \text{ is undefined, our } (g(i) + q + 1)\text{st guess at } (f_{m_{2j}}(g(i) + q + 1))^{(3)} \text{ says so. For sufficiently large } q, \text{ force}(j, \langle P, Q \rangle)(0) \text{ is defined in } \leq g(i) + q + 1 \text{ steps, for}

\[ \langle P, Q \rangle = \langle \text{Full}(P_{2j}(l_{2j}(g(i) + q + 1)), \delta), \text{Full}(Q_{2j}(l_{2j}(g(i) + q + 1)), \varepsilon) \rangle. \]

So a sufficiently large $q \geq q_0$ is as desired. Similar arguments apply in the other two situations. Q.E.D.

Notice that we can $f$-recursively decide whether $q$ is as described in Lemma 2. We proceed as follows, recursively in $f$. Search for the least $q$ as described in Lemma 2. Let $g(i + 1) = g(i) + q + 1$. If $q$ changes the primary or secondary guess at $j$, let $j = z_{i+1} = z$. Otherwise let $z_{i+1} = z_i + 1$. Now we preserve some earlier guesses: for $j < z$, let

\[ \langle P_{j+1}, Q_{j+1} \rangle = \langle P_j, Q_j \rangle, \quad \langle n_{j+1}^{(1)}, m_{j+1}^{(1)} \rangle = \langle n_j, m_j \rangle; \]

for $2j + 1 < z$, let

\[ x_{j+1} = x_j, \quad \sigma_{j+1} = \sigma_j, \quad \tau_{j+1} = \tau_j; \]

for $j < z - 1$, let

\[ \delta_{j+1} = \delta_j, \quad \varepsilon_{j+1} = \varepsilon_j. \]

The situation in which $q$ changes the secondary guess at $z$ is easiest to handle.
Here our guesses $\langle n'_i, m'_i \rangle$ have been found to be wrong relative for $\langle P'_i, Q'_i \rangle$. We let $\delta'_{i-1} = \delta'_{i-1}, e'_{i-1} = e'_{i-1}, \langle P'^{i+1}, Q'^{i+1} \rangle = \langle P'_i, Q'_i \rangle$ and, if $z = 2j + 1, x'_i = x_j, \sigma'_{i+1} = \sigma_j, \tau'_{i+1} = \tau_j$. Select $\beta$ and $\gamma$ as in (8) and let $\beta_{i+1} = \beta, \gamma_{i+1} = \gamma$. Note that $l'_z = l'^{i+1}$. Now find the least $\langle n, m \rangle$ such that

$$\langle P'^{i+1}, Q'^{i+1} \rangle (l'^{i+1}(g(i + 1))) = \langle (f)_n \uparrow \text{Str}(l'^{i+1}(g(i + 1))), (f)_m \uparrow \text{Str}(l'^{i+1}(g(i + 1))) \rangle,$$

and let $\langle n'^{i+1}, m'^{i+1} \rangle = \langle n, m \rangle$.

Next easiest is the case in which $q$ creates a new guess at $z = 2j + 2$. Select strings $\delta$ and $\epsilon$ as described in (9.1) to be $\delta'_{i-1}$ and $e'_{i-1}$, respectively. Let $P_{i-1} = P^{i-1}(l_{i-1}(g(i + 1)))(\delta)$ and $\gamma_i = Q_{i-1}(l_{i-1}(g(i + 1)))(\epsilon)$. So $\langle P^{i-1}, Q^{i-1} \rangle$ and $\langle P'_i, Q'_i \rangle$ are defined as described before the construction began. Now select $\langle n^{i-1}, m^{i-1} \rangle$ as in the previous case.

The cases in which $q$ changes the primary guess at $z$ and in which $q$ creates a new condition at $z = 2j + 1$ are similar. Select $a$ and $e$ as described in (7) or in (9), and let $P^{i+1}_j = P^{i+1}_j(l^{i+1}_j(g(i + 1)))(\delta)$ and $\gamma_{i+1} = Q^{i+1}_j(l^{i+1}_j(g(i + 1)))(\epsilon)$. Our $\langle P'_i, Q'_i \rangle$ is now determined. If $\langle \delta, \epsilon \rangle$ is a q-combination by virtue of (6)(a), let $\langle x'_i, \sigma'_i, \tau'_i \rangle = \langle x, \sigma, \tau \rangle$ described in (6)(a). If $\langle \delta, \epsilon \rangle$ is a q-combination by (6)(b), let $x'_i = -1, \sigma'_i = \tau'_i = \langle . \rangle$. Form $\langle P'_i, Q'_i \rangle$ as indicated in the preparatory remarks. We now select $\langle n'^{i-1}, m'^{i-1} \rangle$ as in the previous two cases.

Notice that $\langle \delta'_{i-1}, e'_{i-1} \rangle$ is changed from $\langle \delta'_{i-1}, e'_{i-1} \rangle$ only if we changed a primary guess; $\langle \delta'_{i-1}, e'_{i-1} \rangle$ is defined while $\langle \delta'_{i-1}, e'_{i-1} \rangle$ was undefined iff we created a new guess at $z$. It is easy to verify that (1.1 + 1), (2.1 + 1) and (3.i + 1) are true. We now show that all our guesses settle down to sequences as described in (1), (2*), (3*), and (4*) and (5).

**Lemma 3.** There are sequences $\{\langle P_j, Q_j \rangle\}_{j<\omega}, \{\langle \delta_j, \epsilon_j \rangle\}_{j<\omega}, \{\langle x_j, \sigma_j, \tau_j \rangle\}_{j<\omega}, \{\langle n_j, m_j \rangle\}_{j<\omega}$ making (1), (2*), (3*), (4*), and (5) true; and for any $k$ there is an $i_k$ such that for all $i \geq i_k$

1. for $j \leq k, j$ has properties 1 and 2 at $i; k < z_i$;
2. for $j \leq k, \langle n_j, m_j \rangle = \langle n_j, m_j \rangle$;
3. for $j < k, \lim_{j \rightarrow \omega} P'_j = P_j_Q_j$;
4. for $j < k, \langle \delta_j, \epsilon_j \rangle = \langle \delta_j, \epsilon_j \rangle$;
5. for $2j + 1 < k, \langle x_j, \sigma_j, \tau_j \rangle = \langle x_j, \sigma_j, \tau_j \rangle$.

**Proof.** The crucial fact here is that $g$ is increasing. For $k = 0, i_k = 0$. Assume for $k$. Select $i \geq i_k$ such that for all $q \geq g(l)$ and all $2j \leq k, w$ our qth guess at $(q)_n \uparrow (q)_m$ is correct. For all $i \geq i$, if $k$ is even, $k + 1$ has property 1 at $i$, is not 1-bad at any $(i, q')$, and is not selected for a primary change. We may let $\langle P_{k+1}, Q_{k+1} \rangle$ be least $\langle n, m \rangle$ such that $\langle (f)_n, (f)_m \rangle = \langle P_{k+1}, Q_{k+1} \rangle$. For each $\langle n', m' \rangle < \langle n_{k+1}, m_{k+1} \rangle$ there is an $l_{(n', m')} \uparrow l$ such that

$$\langle (f)_n \uparrow \text{Str}(l), (f)_m \uparrow \text{Str}(l) \rangle \neq \langle P_{k+1} \uparrow \text{Str}(l), Q_{k+1} \uparrow \text{Str}(l) \rangle.$$

Let $i_{k+1}$ be an $i \geq i$ such that $l_{i_{k+1}}(g(i)) \geq l_{(n', m')}$, for all such $\langle n', m' \rangle$. For $i \geq i_{k+1}$, we have $\langle n_{i_{k+1}}, m_{i_{k+1}} \rangle = \langle n_{k+1}, m_{k+1} \rangle$. $k + 1$ has property 2 at such a stage $i$, so is not 1-bad at any $(i, q')$, and is not selected for a secondary change. So
Unifom Upper Bounds on Ideals of Turing Degrees

$k + 1 < z_i$. (13) and (14) are obviously true, letting $\delta_k = \delta_k^{j+1}$, $\varepsilon_k = \varepsilon_k^{j+1}$, and $x_j = x_j^{j+1}$, $\delta_j = \delta_j^{j+1}$, $\tau_j = \tau_j^{j+1}$ if $k + 1 = 2j + 1$. Q.E.D.

We finally must check that $B = \lim_{i \to \infty} B_i$, $C = \lim_{i \to \infty} C_i$. For any $j$ at which either we create a new guess at $i$ or make a primary change at $i$. For such an $i$, we have arranged that $P(i)(g(i)) \leq \beta_i$, $Q_i(i)(g(i)) \leq \tau_i$. But for sufficiently large $j$, these $P(i)(g(i))$ and $Q(i)(g(i))$ may be made arbitrarily long. This insures the desired limits. Q.E.D.

**Corollary.** Where $I$ is a countable jump ideal and $a$ is an u.u.b. on $I$ then there is an $I$ exact $(b, e)$ with $(b \lor c) < a$.

**Proof.** With $a$, $b$, $c$ as above, if $b \lor c = a$, $(b \lor c)^{(2)} \leq a^{(1)} = (b \lor c)^{(1)}$, a contradiction. Thus $(b \lor c) < a$.

The construction of Theorem 1 may be altered, using Sacks’ technique for constructing minimal upper bounds, to insure that $b$ and $c$ are both minimal.

Recall that $a$ is high over $b$ iff $b \leq a \leq b^{(1)} \leq b^{(2)} \leq a^{(1)}$. Can Theorem 1 be improved to: $a$ is an u.u.b. on $I$ iff $a$ is high over the join of an $I$-exact pair? Perhaps. But we see no way to modify the previous construction to make $f \leq \tau(B \oplus C)^{(1)}$.

Furthermore, for all we know now Theorem 1 may be strengthened to: $a$ is an u.u.b. on $I$ iff for some $I$-exact $(b, e)$, $(b \lor c)^{(1)} = a$; this is equivalent to: if $a$ is an u.u.b. on $I$, for some $I$-exact $(b, e)$, $(b \lor c)^{(1)} \leq a$.

We now characterize u.u.b.s in terms of weak u.u.b.s.

**Theorem 2.** For a countable jump ideal $I$, $a$ is an u.u.b. on $I$ iff for some $b \leq a$, $b$ is a weak u.u.b. on $I$ and $b^{(2)} \leq a^{(1)}$.

**Proof.** $(\Rightarrow)$. Let $B \in b$ parametrize $\bigcup I \cap o2$. Fix $A \in a$. $X \subseteq \omega$ is total iff for every $x$ there is a $y$ such that $\langle x, y \rangle \in X$. Since $B^{(2)} \leq \tau A^{(1)}$, we may guess recursively in $A$ at whether $(B)_e$ is total and in the limit we are correct. Fix such a guessing procedure. Let $h(x, e, n) = \langle (B)_e \rangle$ be the least $y$ such that either $\langle x, y \rangle \in (B)_e$ or the $(n + y)$th guess is that $(B)_e$ is not total. Define $f$ by:

$$(f)_{g(x, e, n)} = \begin{cases} 0 & \text{if the } (n + h(x, e, n))\text{th guess is that } (B)_e \text{ is not total;} \\ h(x, e, n) & \text{otherwise.} \end{cases}$$

If $(B)_e$ is total, $(B)_e = * \text{ graph } (f)_{g(x, e, n)}$; if $(B)_e$ is not total, $(f)_{g(x, e, n)} = * \land x. 0$. By Lemma 1, $\deg(f)$ is an u.u.b. on $I$. Since $f \leq \tau A$, so is $a$.

$(\Rightarrow)$ Let $f \in a$ parametrize $\bigcup I \cap \omega 2$. Let $\langle \psi_j \rangle_{j \in \omega}$ be a recursive enumeration of primitive recursive relations on $\omega 2 \times \omega \times \omega$. Introducing “$B$” as an uninterpreted one place predicate constant, let $\phi_j$ be “$(\exists x) \neg (\exists y) \psi_j(B, x, y)$.” Let a condition be a finite sequence of members of $\bigcup I \cap \omega 2$. Where $\langle f_0, \ldots, f_{k-1} \rangle = K$ is a condition, let

$$K \models B(m) \text{ iff } (m)_0 < k \text{ and } f_{(m)_0}(m)_1 = 1.$$ 

Other clauses in the definition of forcing run as usual. Note that

$$K \models \neg B(m) \text{ iff } (m)_0 < k \text{ and } f_{(m)_0}(m)_1 = 0.$$ 

Conditions may be coded as sequence numbers:

$$\langle n_0, \ldots, n_{k-1} \rangle \text{ codes } \langle \text{sg}((f)_0), \ldots, \text{sg}((f)_n) \rangle.$$
where for any \( x \in \omega \) and \( h \in \omega^\omega \),

\[
\text{sg}(h)(x) = \begin{cases} 0 & \text{if } h(x) = 0, \\ 1 & \text{otherwise.} \end{cases}
\]

We abuse terminology and call sequence numbers conditions.

For \( X \subseteq \omega \), \( X^{(<\omega)} = \{ \langle x, y \rangle \in X | x < k \} \). For a condition \( K = \langle f_0, \ldots, f_{k-1} \rangle \), \( \hat{K} = f_0 \oplus \cdots \oplus f_{k-1} \). If \( B \) is generic and extends \( K \), we shall have \( B^{(<\omega)} = \hat{K} \). For \( \sigma \in \text{Str} \), \( \sigma \) is consistent with \( \hat{K} \) iff for all \( x < \text{lh}(\sigma) \), if \( (x)_0 < k \), \( (\sigma)_x = f_{(x)_0} ((\sigma)_1) \); \( K \) includes \( \sigma \) iff \( \sigma \) is consistent with \( K \) and for all \( x < \text{lh}(\sigma) \), \( (x)_0 < k \). All these definitions carry over to where \( K \) is a sequence number via the encoding previously described. From now on, conditions are sequence numbers.

The use of \( \text{sg} \) in this encoding leads to another abuse of terminology. For \( K = \langle n_0, \ldots, n_{k-1} \rangle \), our \( q \)th guess at \( X = (f_0 \oplus \cdots \oplus f_{k-1})^{(2)} \) is \( Y = (f)_G(G(H(k, K, q), q), q) \). Since \( \hat{K}^{(2)} \) is clearly \( 1 \)-reducible to \( X \), we shall call \( Y \) our \( q \)th guess at \( \hat{K}^{(2)} \).

**Lemma 4.** \( "K \vdash \varphi_j" \) and \( "K \vdash \neg \varphi_j" \) are \( \Sigma_2^0 \) and \( \Pi_0^0 \) in \( \hat{K} \), respectively.

**Proof.** \( K \vdash (\langle y \rangle) \psi_h(n, y) \) iff for any \( \sigma \in \text{Str} \) and any \( y \), if \( \sigma \) is consistent with \( K \), \( "\neg \psi_h(\sigma, n, y)" \) is true. Thus \( "K \vdash \varphi_j" \) is \( \Sigma_2^0 \) in \( \hat{K} \). For \( X \subseteq \omega \) and \( \text{lh}(K) = k \), let \( \Phi(K, X, m) = \hat{K} \cup \{ \langle x + k, y \rangle \mid \langle x, y \rangle \in X^{(<m)} \} \). Notice that \( K' \) extends \( K \) iff for some \( X \subseteq \text{U}(\text{U}) \) and some \( m \), \( K' = \Phi(K, X, m) \). Using this fact we can show that \( K \vdash \neg \varphi_j \) iff for every \( x, m \in \omega \) and \( X \subseteq \text{U}(\text{U}) \):

(\( \dagger \)) there are \( \sigma \in \text{Str} \) and \( y \) such that \( \sigma \) is consistent with \( \Phi(K, X, m) \) and \( \psi_h(\sigma, x, y) \).

(\( \dagger \)) has the form \( "(\exists \sigma)(\exists y)P(\hat{K}, X, m, \sigma, x, y)" \), with \( P \) recursive. So \( K \vdash \neg \varphi_j \) iff for all \( x \) and \( m \):

(\( \dagger \dagger \)) for all \( X \subseteq \text{U}(\text{U}) \), \( \langle y \rangle P(\hat{K}, X, m, \sigma, x, y) \).

\( \dagger \dagger \) is equivalent to a \( \Sigma_2^0 \) in \( \hat{K} \) formula by the Kreisel basis theorem and the fact that \( \hat{K}^{(1)} \subseteq \text{U} \). Notice that here is where the difference between \( \text{U} \) and \( \text{U}(\text{U}) \) appears. We now have \( "K \vdash \neg \varphi_j" \) in a \( \Pi_0^0 \) in \( \hat{K} \) form. Q.E.D.

Our goal is to construct sequences \( \{K_j\}_{j<\omega} \), \( \{x_j\}_{j<\omega} \) and \( \{\beta_i\}_{i<\omega} \) such that:

(1) for all \( j \), \( K_j \) is a condition and \( K_{j+1} \) extends \( K_j \);

(2) for all \( j \),

\[
\begin{align*}
\text{if } x_j & \geq 0, \quad K_{2j+1} \vdash \neg (\exists y) \psi_h(x_j, y); \\
\text{if } x_j & = -1, \quad K_{2j+1} \vdash \neg \varphi_j;
\end{align*}
\]

(3) for all \( j \), \( K_{2j+2} = K_{2j+1} \cap \langle f_j \rangle \);

(4) for all \( i \) and \( j \), \( \beta_i \in \text{Str} \), \( \beta_{i+1} \leq \beta_i \) and \( \beta_i \) is consistent with \( K_j \).

Notice that (2) implies \( \lim \text{lh}(K_j) = \omega \), which with (4) implies that \( \lim \beta_i = \bigcup K_j \).

Of course, such a construction cannot be carried out recursively in \( f \). We resort to guessing at the sequences \( \langle K_j \rangle_{j<\omega} \) and \( \langle x_j \rangle_{j<\omega} \). At stage \( i \) we shall have \( z_i \), for \( j \leq 2z_i \) guesses \( K_j \) at \( K_j \), and for \( j < z_i \) guesses \( x_j \) at \( x_j \). Revising previous terminology, let \( (K', x) \) be a \( j \)-witness for \( K \) iff \( K' \) extends \( K \) and forces \( "(\exists y) \psi_h(B, x, y)" \). \( (K', x) \) is a \( j \)-witness for \( "K" \) and \( "K has a j\)-witness" are \( \Pi_0^0 \) and \( \Sigma_2^0 \) in \( \hat{K} \), respectively. Clearly if \( \hat{K}' \) extends \( K' \) and \( (K', x) \) is a \( j \)-witness for \( K' \), \( (K', x) \) is also a \( j \)-witness for \( K \). We shall say that \( (K, x) \) is consistent with a string \( \beta \) iff \( K \) is...
consistent with \( \beta \). Notice that if \( K \) has no \( j \)-witness consistent with \( \beta \), any condition extending \( K \) and including \( \beta \) forces \( \varphi_j \). Fix an \( f \)-recursive function \( \text{Incl} \) such that: for \( \beta \) consistent with \( K \), \( \text{Incl}(K, \beta) \) extends \( K \) and includes \( \beta \). For example, where \( \text{lh}(K) = k \), and \( \beta \) is consistent with \( K \), let

\[
\text{Incl}(K, \beta) = \begin{cases} 
K & \text{if } K \text{ includes } \beta, \\
K \prec \langle n_h, \ldots, n_i \rangle & \text{otherwise},
\end{cases}
\]

where for \( k \leq i \leq l \), \( n_i \) is the least \( n \) such that for all \( x < \text{lh}(\beta) \) with \( (x)_0 = i \), \((\beta)_x = \text{sg}((f)_n)((x)_0)\). For \( j < z_i \), we shall say that \( 2j + 1 \) has property 1 at stage \( i \) iff:

- if \( x_j > 0 \) then \( (K_{i+1}, x_j) \) is a \( j \)-witness for \( K_{i+1} \);
- if \( x_j = -1 \), then there is no \( j \)-witness for \( K_{i+1} \) consistent with \( \beta_i \).

We would like to have all \( 2j + 1 \) with property 1 at stage \( i \) for \( j < z_i \). But to keep our construction recursive in \( f \), we cannot be so straightforward. Instead we insure that for all stages \( i \):

1. For all \( j < z_i \), our \( g(i) \)-th guess at \( (K_{i+1})^{(2)} \) says that \( 2j + 1 \) has property 1.
2. If \( z_i > 0 \), \( \beta_i \) is included in \( K_{i+1} \). (This permits us to have \( K_{i+1} = K_{i+1} \prec z_i \) without fear of destroying consistency with \( \beta_i \).)

We now sketch the construction.

Stage 0. \( z_0 = 0 \), \( K_0^0 = \langle \rangle \); \( \beta_0 = \langle \rangle \); \( g(0) = 0 \). (1.0) and (2.0) are vacuously true.

Stage \( i + 1 \). Assume that \( z_i \), \( g(i) \), \( \beta_i \prec \langle K_i^j \rangle_{j \leq z_i} \) and \( \langle x_i^j \rangle_{j < z_i} \) are defined with (1.0) and (2.0) true. For \( j < z_i \), \( 2j + 1 \) is bad at \( (i, q) \) iff our \( (g(i) + q + 1) \)-th guess at \( (K_i^j)^{(2)} \) says that \( 2j + 1 \) lacks property 1. Call \( \beta \) a \( q \)-combination for \( 2j \) at stage \( i \), where \( j \leq z_i \) iff \( \beta \leq \beta_i \), \( \beta \leq g(i) + q + 1 \), \( \beta \) is consistent with \( K_i^j \), and: if our \( (g(i) + q + 1) \)-th guess at \( (K_{i+1}^j)^{(2)} \) says that \( K_{i+1}^j \) has a \( j \)-witness consistent with \( \beta \), it identifies one in \( \leq g(i) + q + 1 \) steps. This property is decidable in \( f \). We shall say that \( q \) changes the guess at \( 2j + 1 \), for \( j \leq z_i \), iff for all \( k < j \), \( 2k + 1 \) is not bad at \( (i, q) \), \( 2j + 1 \) is bad at \( (i, q) \), and there is a \( q \)-combination for \( 2j \). We shall say that \( q \) creates a guess at \( 2z_i + 1 \) iff for all \( k \leq z_i \), \( 2k + 1 \) is not bad at \( (i, q) \) and there is a \( q \)-combination for \( 2z_i \).

**Lemma 5.** There is a \( q \) such that for some \( j \leq z_i \), \( q \) either changes or creates a guess at \( 2j + 1 \).

**Proof.** Fix \( j^* = \text{the least } j < z_i \) for which \( 2j + 1 \) lacks property 1, if there is one; \( j^* = z_i \) otherwise. Suppose that for all \( q \geq q_0 \), our \( (g(i) + q + 1) \)-th guess at \( (K_i^j)^{(2)} \) for any \( k \leq j^* \) is correct. Thus for \( q \geq q_0 \) if \( k < j^* \) \( 2k + 1 \) is not bad at \( (i, q) \); if \( j^* < z_i \), \( 2j^* + 1 \) is bad at \( (i, q) \). Select a \( \beta \leq \beta_i \) which is consistent with \( K_i^j \). Thus for \( k \leq 2j^* \), \( \beta \) is consistent with \( K_i^j \). If there is a \( j^*-\)witness for \( K_i^j \) consistent with \( \beta \), let \( q \geq q_0 \) be large enough so that \( (K_i^j)^{(2)} \) identifies one in \( \leq g(i) + q + 1 \) steps. \( \beta \) is a \( q \)-combination for \( 2j^* \). If \( j^* < z_i \), \( q \) indicates a change at \( 2j^* + 1 \); if \( j^* = z_i \), \( q \) creates a guess at \( 2j^* + 1 \). Q.E.D.

Notice that whether \( q \) is as described in Lemma 5 is decidable in \( f \). So we may search, recursively in \( f \), for the least such \( q \). Let \( g(i + 1) = g(i) + q + 1 \); where \( j \) corresponds to \( q \) as required by Lemma 5, let \( z_{i+1} = j + 1 \). We abbreviate \( "z_{i+1}" \) as \( "z" \). Select \( \beta_{i+1} \) to be a \( q \)-combination for \( 2z - 2 \). We preserve previous guesses
as follows: $K_{i+1}^j = K_i^j$ for $k \leq 2z - 2$; $x_{k+1}^j = x_k^j$ for $k < z - 1$. We now define $x_{i+1}^{i+1}$ and $K_{i+1}^{i+1}$.

If our $g(i + 1)$st guess at $(K_{2z}^{(2)})$ says that $K_{i+1}^{i+1}$ has a $(z - 1)$-witness consistent with $\beta_{i+1}$, it actually identifies some $(K, x)$ as such a witness in $\leq g(i + 1)$ steps. Select the least such $\langle K, x \rangle$ and let $x_{i+1}^{i+1} = x$, $K_{i+1}^{i+1} = \text{Incl}(K_{i+1}^{i+1}, \beta_{i+1})$. Otherwise our guess says that $K_{2z}^{(2)}$ has no $(z - 1)$-witness consistent with $\beta_{i+1}$. Let $x_{i+1}^{i+1} = -1$ and $K_{i+1}^{i+1} = \text{Incl}(K_{i+1}^{i+1}, \beta_{i+1})$. Notice that $(1.i + 1)$ and $(2.i + 1)$ are true. Let $K_{i+1}^{i+1} = K_{i+1}^{i+1} \cup \{z\}$. This construction settles down.

**Lemma 6.** There are sequences $\{K_j\}_{j<\omega}$ and $\{x_j\}_{j<\omega}$ with $\{\beta_j\}_{j<\omega}$ as just constructed, such that (1)-(4) are true; furthermore for any $k$ there is an $i_k$ such that for all $i \geq i_k$:

1. $z_i > k$;
2. for all $j \leq 2k$, $K_j = K_j$;
3. for all $j < k$, $x_j = x_j$.

The proof is very much like that of Lemma 3, except easier, so we omit it.

Letting $B = \bigcup_j K_j$, $B$ is a parametrization of $\bigcup \exists^o$. Since $B = \lim_i \beta_i$, $B \leq_T f$. Since $f^{(1)}$ can tell us when our guesses at $(K_{2z}^{(2)})$ are correct, $B^{(2)} \leq_T f^{(1)}$.

Q.E.D.

We do not know whether this theorem may be improved to: $a$ is an u.u.b. on $I$ iff for some weak u.u.b. $b$ on $I$, $a = b^{(1)}$.

Combining this construction with the exact-pair construction we may obtain $b$ and $c$ in Theorem 1 which are both weak u.u.b.s on $I$.

Clearly the $b$ constructed in Theorem 2 ($\Rightarrow$) is strictly below $a$. This observation is strengthened by the following.

**Theorem 3.** For a countable jump ideal $I$, $\{a \mid a$ is an u.u.b. on $I\}$ has no minimal member.

**Proof.** Let $f \in a$ parametrize $\bigcup I$. We construct $h <_T f$, $h$ parametrizing $\bigcup I$. Let $\langle \phi_f \rangle_{f<\omega}$ be as in the previous proof; we introduce an uninterpreted binary predicate letter “$H$” intended to denote the graph of a generic function. Let a condition be a sequence $K = \langle f_0, \ldots, f_{k-1} \rangle$ of members of $\bigcup I$. Let

$$K \models H(n, m) \iff (n)_0 < k \text{ and } f((n)_1)((n)_1) = m.$$ 

The other clauses in the definition of forcing are as usual. Again we note that

$$K \models \neg H(n, m) \iff (n)_0 < k \text{ and } f((n)_1)((n)_1) \neq m.$$ 

Let $K$ be the partial function with domain $\omega^{<\omega}$ such that $K(\langle i, x \rangle) = f_i(x)$. Since $K$ is partial, $K^{(1)}$ is undefined; therefore we shall abuse notation and write “$K^{(1)}$” for “$(f_0 \oplus \cdots \oplus f_{k-1})^{(1)}$”.

Notice that Lemma 1 provides a fixed $f$-recursive way of guessing at an $f$-index for that set, uniformly in a code for $K$. A finite function shall be one from a member of $\omega$ into $\omega$. A finite function $h$ is consistent with $K$ iff for all $x \in \text{dom}(h)$ with $(x)_0 < k$, $K(x) = h(x)$; $K$ includes $h$ iff $\text{dom}(h) \subseteq \omega^{<\omega}$ and $h$ is consistent with $K$. $R_j$ is the requirement $\{j\}^H \neq f$. $K$ meets $R_j$ with $x$ in $t$ steps iff for some $y$, $K \models \{j\}^H(y)$ converges to $y$ in $t$ steps” and $f(x) \neq y$. Where $h$ is a partial function, we understand a computation in $\text{graph}(h)$ to halt as soon as the oracle for $\text{graph}(h)$ is asked: “Is $\langle x, y \rangle \in \text{graph}(h)$?” for $x \notin \text{dom}(h)$. With this understanding, observe
that $K$ has an extension meeting $R_j$ with $x$ in $t$ steps iff there is a finite function consistent with $K$ and a $y \neq f(x)$ such that $\{j\}_{\text{graph}(h)}(x)$ converges to $y$ in $t$ steps; we may search for such an $h$ recursively in $K$, since finite functions code as sequence numbers.

Let sequence numbers encode conditions by $\langle n_0, \ldots, n_{k-1} \rangle \leftrightarrow \langle (f)_{n_0}, \ldots, (f)_{n_{k-1}} \rangle$. So we freely abuse our terminology and treat sequence numbers as conditions.

Fix an $f$-recursive function Incl such that for a finite $h$ consistent with $K$, Incl$(K, h)$ extends $K$ and includes $h$. (For example, vary the corresponding definition in the previous proof.)

Let $(K', x)$ be a $j$-witness for $K$ iff $K'$ extends $K$ and meets $R_j$ with $(x)_{0}$ in $\leq (x)_{1}$ steps. Call $h$ consistent with $(K, x)$ iff consistent with $K$. Suppose $K$ has no $j$-witness consistent with a finite function $h'$, $K'$ extends $K$ and includes $h$. Then for some $x$, $K \vdash \{j\}_{\text{graph}(h')(x)}$ is undefined. Suppose not. We may define $f$ by $f(x) = y$ iff

\[(\ast)\text{ some finite function } h' \text{ is consistent with } K' \text{ and } \{j\}_{\text{graph}(h')(x)} = y.\]

Here is why. By our assumption, for any $x$, $K'$ has an extension $K''$ forcing $\{j\}_{\text{graph}(h')(x)}$ is defined." Since $K''$ includes $h$, $(K'', x)$ is not a $j$-witness for $K$. So if $K'' \vdash \{j\}_{\text{graph}(h')(x)} = y'$, $y = f(x)$. The existence of such a $K''$ is equivalent with $(\ast)$. We would like to define sequences $\{K_x\}_{x < \omega}$, $\{x_j\}_{j < \omega}$ and $\{h_j\}_{j < \omega}$ such that:

1. for each $j$, $K_j$ is a condition;
2. for each $j$,
   
   if $x_j \geq 0$, $(K_{2j+1}, x_j)$ is a $j$-witness for $K_{2j}$; if $x_j = -1$, $K_{2j+1} \vdash \{j\}_{\text{graph}(h_j)(x)}$ is undefined" for some $x$;
3. for each $j$, $K_{2j+2} = K_{2j+1} < j$;
4. for each $i$ and $j$, $h_i$ is a finite function, $h_{i+1}$ properly extends $h_i$, and $h_i$ is consistent with $K_i$.

(3) implies that $h = \lim_j K_j$ is total;
(4) implies that $h = \lim_j h_i$. By (3), $h$ parametrizes $\bigcup I$. By (2) $f \neq T \upharpoonright h$.

To make this construction recursive in $f$, we resort to guessing. At stage $i$, we shall have $z_i$, $h_i$, $g(i)$, for $j \leq 2z$, a guess $K_j$ at $K_j$, and for $j \leq z_i$ a guess $x_j$ at $x_j$.

We make sure that at each stage $i$:

1. for $j < z_i$, if $x_j \geq 0$, $(K_{2j+1}, x_j)$ is a $j$-witness for $K_{2j}$;
2. for $j < z_i$, if $x_j = -1$, our $g(i)$th guess at $(K_{2j+1}, x_j)$ says $(\ast, i, j)$ is false.

We now describe the construction.

\textit{Stage 0.} $z_0 = 0$, $h_0 = \text{the null function}$, $K_0 = \langle \rangle$, $g(0) = 0$.

\textit{Stage} $i + 1$. Suppose we have $t_i$, $h_i$, $g(i)$, $(K_j)_{j \leq 2z_i}$, $\langle x_j \rangle_{j \leq z_i}$, with (1.1)–(3.1) true. For $j < z_i$, $2j + 1$ is bad at $(i, q)$ iff $x_j = -1$ and our $(g(i) + q + 1)$st guess at $(K_{2j+1})_{(i, q)}$ says $(\ast, i, j)$ is false. For a finite function $h$, $(h, x)$ is a $q$-combination for $2j$ at $i$ iff $h$ properly extends $h_i$, $\langle h, x \rangle \leq g(i) + q + 1$, and $\{j\}_{\text{graph}(h)(x)}$ is defined in $(x)_{1}$ steps and has value $\neq f((x)_{0})$.\]
We shall say that \( q \) changes the guess for \( 2j + 1 \) at stage \( i \) iff: for all \( k < j \), \( 2k + 1 \) is not bad at \((i, q)\), \( 2j + 1 \) is, and there is a \( q \)-combination for \( 2j \). We shall say that \( q \) creates a guess for \( 2z_i = 1 \) iff: for all \( k < z_i \), \( 2k + 1 \) is not bad at \((i, q)\), and either there is a \( q \)-combination for \( 2z_i \) or else \( q = 0 \) and our \((g(i) + 1)\)st guess at \((K_i^{(1)}, i, z_i)\) is true.

**Lemma 7.** Some \( q \) either changes or creates a guess.

Proof is very much like that of Lemma 5.

Whether \( q \) changes or creates a guess is decidable in \( f \). So recursively in \( f \) we search for the least such \( q \). Let \( g(i + 1) = g(i) + q + 1 \). If \( q \) changes or creates a guess at \( 2j + 1 \), let \( j + 1 = z_{i+1} \). Letting \( z = z_{i+1} \), we preserve earlier guesses:

\[
\text{for } j \leq 2z - 2, \ K_j^{i+1} = K_j; \quad \text{for } j < z - 1, \ x_j^{i+1} = x_j.
\]

If there is a \( q \)-combination for \( 2z - 2 \), let \( (h_{i+1}, x_{i+1}^{+1}) \) be the least such. Otherwise let \( x_{i+1}^{ji-1} = -1 \) and \( h_{i+1} = h_i \cup \{\langle \text{dom}(h_i), 0\rangle\} \). Let \( K_{zi+1}^{1} = \text{Incl}(K_{zi+1}^{ji-2}, h_{i+1}) \).

Notice that \((1.i + 1)-(3.i + 1)\) are true. Now let \( K_{zi+1} = K_{zi+1}^{-}\cdot \langle z\rangle \).

**Lemma 8.** With \( \langle h_i \rangle_{i<\omega} \) as just constructed, there are sequences \( \langle K_j \rangle_{j<\omega} \) and \( \langle x_j \rangle_{j<\omega} \) of which (1)-(4) are true; furthermore for each \( k \) there is an \( i_k \) such that for all \( i \geq i_k \):

1. for \( j \leq 2k \), \( K_j = K_j^i \);
2. for \( j < k \), \( x_j = x_j^i \).

The proof of this lemma should now be routine. Because this entire construction is recursive in \( f \), and \( h = \lim_i h_i \), \( h \leq_T f \). So by preliminary remarks, we are done.

Q.E.D.

Where \( I \) is a countable jump ideal \( a \) is a nice u.u.b. on \( I \) iff \( a \) is the degree of a nice parametrization of \( \bigcup f \); a parametrization \( f \) of \( \bigcup f \) is nice iff for some \( G \leq_T f \), \( H \leq_T f \), for all \( x \) and \( y \): \( (f)_x \cdot (f)_y = (f)_x \oplus (f)_y \). This notion is introduced in [1]; in [2] it is shown that \( a \) is a nice u.u.b. on \( I \) iff for some u.u.b. \( b \) on \( I \), \( a = b^{(1)} \). In [2] the following notions are defined. \( I \) is a hierarchy ideal iff for some \( A \subseteq \omega \) and some \( \alpha \), \( \bigcup f = L_{\alpha}[A] \inter \omega_\alpha \). \( I \) is a case 1 hierarchy ideal iff for some \( B \in L_{\alpha}[A] \), \( \alpha < \omega_\beta \) and \( \bigcup f = L_{\alpha}[A] \inter \omega_\omega \); \( I \) is a case 2 hierarchy ideal iff for some \( B \in L_{\alpha}[A] \), \( \alpha = \omega_\beta \) and \( \bigcup f = L_{\alpha}[A] \inter \omega_\omega \); \( I \) is a case 3 hierarchy ideal if it is a hierarchy ideal not falling under cases 1 or 2. Any case 1 hierarchy ideal has a least nice u.u.b.; for example, if \( \bigcup f = \{\text{f is arithmetic}\} \), that nice u.u.b. is \( 0^{(\omega)} \). In [2] it is asked whether any case 2 or case 3 hierarchy ideals have a minimal nice u.u.b. The technique of Theorem 3 may be modified to provide a negative answer.

**Theorem 4.** For \( I \) a case 2 or case 3 hierarchy ideal, \( \{a \mid a \text{ is a nice u.u.b. on } I\} \) has no minimal member.

**Proof.** Let \( f \in a \) be a nice parametrization of \( \bigcup f \). It suffices to construct a parametrization \( h \) of \( \bigcup f \) with \( h^{(1)} \leq_T f \). Let conditions and forcing be as in the previous proofs except that “\( H \)” is monadic, and:

\[
K \Vdash H(x) \iff \langle m, n_0 \rangle < k \text{ and } K = \langle f_0, \ldots, f_{k-1} \rangle, f_{(\omega_\alpha)}(n_0) = m.
\]

This way “\( x \in H^{(1)} \)” makes sense. Let \( R_f \) be the requirement \( \{j \mid f^{(1)}_j \neq f \} \). \( K \) meets \( R_f \) with \( x \) iff for some \( y \neq f(x), K \Vdash \langle j \mid f^{(1)}_j(x) = y \rangle \). Because \( f \) is nice, whether...
LEMMA 9. Suppose $K$ is consistent with a finite function $h$. If there is no $j$-witness for $K$ consistent with $h$, and $K'$ extends $K$ and includes $h$, then for some $x$, $K' \not\models \{j\}^{U(1)}(x)$ is undefined.

PROOF. If not, we may define $f$ by $f(x) = y$ iff some extension of $K'$ forces $\{j\}^{U(1)}(x) = y$. Let $f_0, \ldots, f_{k-1} \models \{j\}^{U(1)}(x) = y$ is $\Sigma^0_2$ in $f_kuture$. So $f$ is $\Sigma^1_1$ over $\bigcup I$ with graph($K'$) as a parameter. Since $f$ is a function, $f$ is even $\Delta^1_1$ over $\bigcup I$ in that parameter.

By familiar facts about hyperarithmeticity, in case 2, $f \leq_{\text{HYP}} \text{graph}(K')$; in case 3, $f$ is recursive in the hyperjump of graph($K'$) which belongs to $\bigcup I$. Either way, $f \in \bigcup I$, contradiction. Q.E.D.

The construction of $h$ is much like that used for Theorem 3, with $\{j\}^{U(1)}$ replacing $\{j\}^{U}$. But (2.i) must be changed to: if $j < z_i$, if $x^j_i = -1$ then there is no $j$-witness for $K^j$ which is consistent with $h_i$ and $\leq g(i)$.

The notion of being bad at $(i, q)$ is correspondingly changed. (We are forcing $\Sigma^0_2$ and $\Pi^0_2$ sentences; so $K^j_{2j+1}$ cannot tell us how to select $K^j_{2j+1}$. Since $f$ is nice, "$K$ has a $j$-witness consistent with $h_i$" is $\Sigma^0_1$ in $f$; thus guessing at $K^j_{2j+1}$ is replaced by a search recursive in $f$.) The rest is routine. Q.E.D.

In conclusion, we note that weak u.u.b.s remain shrouded in mystery. For example: are any weak u.u.b.s also minimal u.b.s? The technique of Theorem 3 does not yield a negative answer, for it cannot construct objects recursive in weak u.u.b.s which are not also u.u.b.s. It essentially involves guessing at jumps as described in the guessing lemma; thus by the remark immediately following the proof of the guessing lemma, the previous claim follows. Hopefully the techniques involved in answering questions like the one just posed will suggest a degree-theoretic definition of a weak u.u.b. in some way analogous to that of Theorem 1.

BIBLIOGRAPHY


DEPARTMENT OF PHILOSOPHY
CORNELL UNIVERSITY
ITHACA, NEW YORK 14853