Minimization and NP multifunctions

N. Danner *

Department of Mathematics, University of California Los Angeles, Los Angeles, CA 90095-1555

.

C. Pollett

Department of Mathematics and Computer Science, San Jose State University, San Jose, CA 95192.

Abstract

The implicit characterizations of the polynomial-time computable functions **FP** given by Bellantoni-Cook and Leivant suggest that this class is the complexity-theoretic analog of the primitive recursive functions. Hence it is natural to add minimization operators to these characterizations and investigate the resulting class of partial functions as a candidate for the analog of the partial recursive functions. We do so in this paper for Cobham's definition of **FP** by bounded recursion and for Bellantoni-Cook's safe recursion and prove that the resulting classes capture exactly **NPMV**, the nondeterministic polynomial-time computable partial multifunctions. We also consider the relationship between our schemes and a notion of nondeterministic recursion defined by Leivant and show that the latter characterizes the total functions of **NPMV**. We view these results as giving evidence that **NPMV** is the appropriate analog of partial recursive. This view is reinforced by earlier results of Spreen and Stahl who show that for many of the relationships between partial recursive functions and r.e. sets, analogous relationships hold between **NPMV** and **NP** sets. Furthermore, since **NPMV** is obtained from **FP** in the same way as the recursive functions are obtained from the primitive recursive functions (when defined via function schemes), this also gives further evidence that **FP** is properly seen as playing the role of primitive recursion.

* Corresponding author.
Email addresses: ndanner@member.ams.org (N. Danner), pollett@mathcs.sjsu.edu (C. Pollett).

1 Introduction

When considering the analogy between the arithmetic and polynomial-time hierarchy, a standard view is that polynomial time plays the role of recursive (though not always—see, e.g., Selman [6], to which we will return). The polynomial-time sets are used as the base Δ_0^p of the polynomial hierarchy and Δ_{i+1}^p is defined as those sets polynomial-time in an oracle for some Σ_i^p set. However, this view is not unproblematic. The analogy of Σ_i^p with Σ_i indicates that $\mathbf{NP} \cap \mathrm{co}\mathbf{NP}$ should be identified with the recursive sets. Under the assumption that $\mathbf{P} \neq \mathbf{NP} \cap \mathrm{co}\mathbf{NP}$, this leaves the polynomial-time sets to be viewed in analogy with some smaller class.

A natural choice for this smaller class is the primitive recursive sets (or functions). This is backed up by Cobham's [3] characterization of **FP**, the functions computable in polynomial time, by a scheme of recursion on notation that is nothing more than an explicitly bounded version of primitive recursion itself (formulated for binary words). Further, the work of Bellantoni and Cook [2] and Leivant [4] provides primitive recursive schemes for defining **FP** with no explicit bounds of any sort, instead controlling the primitive recursion by semantically-inspired, syntactically implemented tiering notions. This approach to characterizing complexity classes without any explicit mention of resources or bounds is typically referred to as *implicit computational complexity*.

However, if we are to identify **FP** with the primitive recursive functions and view $\mathbf{NP} \cap \mathrm{co}\mathbf{NP}$ as the appropriate analogy for the recursive sets, we are left with the following question: what is the correct analog of the partial recursive functions? Of course, we expect some class of partial functions. However, from a machine-based view of complexity classes, this is bound to lead to some unpleasantness, as there does not seem to be a straightforward, "clean" way to define complexity of partial functions. This especially becomes clear if we insist upon some notion of time-bounded machines which may not halt on some inputs. It is here that implicit computational complexity truly shines. Since its characterizations make no mention of resources or bounds, it is natural to extend the existing implicit characterizations of **FP** with operators analogous to the classical case, and then analyze the resulting class of partial functions as a candidate for the analog of the partial recursive functions.

Since one passes from the primitive recursive functions to the partial recursive functions by adding the minimization operator to composition and primitive recursion, an obvious starting point is to add a version of minimization to Cobham's, Bellantoni and Cook's, and Leivant's characterizations. To understand the version of minimization that we use, consider the classical definition

used for the partial recursive functions:

$$(\mu z.\varphi(\bar{x}, z) = 0) = a \Leftrightarrow \varphi(\bar{x}, a) = 0 \wedge \forall b{<}a \exists t{>}0.\varphi(\bar{x}, b) = t.$$

To obtain a version of minimization appropriate to polynomial-time functions, we want to ensure that the verification that a is in fact the minimum value requires a search over a polynomial number of values. Thus we replace the bounded quantifier with a sharply bounded one:

$$(\mu z.\varphi(\bar{x}, z) = 0) = a \Leftrightarrow \varphi(\bar{x}, a) = 0 \wedge \forall b{<}|a| \exists t{>}0.\varphi(\bar{x}, b) = t.$$

This definition immediately raises a flag: minimization no longer defines a partial *function*, as there could be several a satisfying $\forall b{<}|a| \exists t{>}0.\varphi(\bar{x}, b) = t$ (e.g., suppose that φ is 1 on all inputs of length ≤ 3 and 0 on all inputs of length > 3; then this minimization would yield all a with $|a| = 4$). Thus our straightforward approach leads us to consider partial *multifunctions* which formally are maps from the natural numbers $\mathbf{N}$ to $\mathcal{P}^{<\omega}(\mathbf{N})$, the set of finite subsets of $\mathbf{N}$. In retrospect, this is not at all unexpected. A partial function satisfies the condition that for all x, $\varphi(x)$ has at most one element. A partial multifunction is the same, except that $\varphi(x)$ has at most finitely many elements. With the (essentially) unrestricted computational resources of recursive functions, collapsing many values into one is not a problem. But when we restrict the complexity of the objects under consideration, it is reasonable to assume that we cannot necessarily perform such a collapse (e.g., if $\varphi(x)$ has too many elements), and so our "functions" must be allowed to have many outputs.

Another view of this sharply bounded quantifier is that instead of having multiple output values, we should instead choose one of the values. This leads directly to the idea of a nondeterministic choice in the computation, and in fact such function classes have been considered. In particular, Spreen [7] defines the class **NPMV** to consist of those partial multivalued functions computable on a nondeterministic Turing machine in polynomial time (there called $\mathbf{NTIMEF}(\mathbf{Pol}_\Sigma)$). Selman [6] analyzes this class in detail as well. Here, "computed" means that y is a possible output of $\varphi(x)$ iff the TM has an accepting path on input x such that the contents of the output tape upon halting are y. In [7], Spreen has a number of characterizations of **NPMV** in terms of various function-definition schemes such as a "guess" operator and unordered search operator; here we extend this list using minimization notions and Bellantoni-Cook style safe recursion. Thus not only does the approach of implicit computational complexity provide a straightforward way to generalize polynomial-time computable functions to a broader class that ought to correspond to the partial recursive functions, it in fact gives a resource-free characterization of a previously-defined class that asserts to play the same role, thus justifying the naturality of (what turns out to be) the single class considered. Furthermore, in an earlier paper Spreen and Stahl [8] provide compelling evidence

that **NPMV** is the correct analog of partial recursive functions by showing that many of the results relating partial recursive functions and the r.e. sets transfer to the single-valued pmf's of **NPMV** and the **NP** sets; although we do not do so here, it is routine to extend these results to all of **NPMV**.

The plan of this paper is as follows. In Section 2 we provide the basic definitions, including a notion of bounded minimization, resulting in an extension of the Cobham class. Section 3 contains the proof that the extension captures exactly **NPMV**. In Section 4 we introduce a resource-free notion of minimization within the framework of Bellantoni and Cook's *safe recursion* [2]. The only other work we know of that directly addresses the issue of adding a minimization operator in a similar way is Bellantoni's [1], where he adds an operator that is single-valued and total; we compare our approach to his in this section as well. In Section 5 we consider the scheme of *nondeterministic recursion* defined by Leivant in [4] and show that the (total) functions definable with this scheme are exactly the total functions of **NPMV** (and hence are the total functions definable using safe recursion and our notion of safe minimization).

2 Definitions

We use lowercase Roman letters a, b, x, y for numeric variables. The *length* of a number, $|x|$, is defined as $\lceil \log_2(x+1) \rceil$. We abbreviate a sequence $x_1, \ldots, x_k$ as $\bar{x}$ when k is not relevant and write $|\bar{x}|$ for $|x_1|, \ldots, |x_k|$. The binary successors are $s_i(x) =_{\text{df}} 2x+i$ for $i = 0, 1$ and the binary predecessor is given by $p(x) =_{\text{df}} \lfloor x/2 \rfloor$.

A *partial multifunction* (*pmf*) is a map $\varphi : \mathbf{N}^k \to \mathcal{P}^{<\omega}(\mathbf{N})$ for some k. Alternatively, φ can be viewed as a relation on $\mathbf{N}^{k+1}$ satisfying the constraint that for all $\bar{x}$, $\{y \mid \langle \bar{x}, y \rangle \in \varphi\}$ is finite. We use φ, ψ, ρ, and θ to range over pmf's. Any total or partial function is identified in the natural way as a pmf, and we assume this identification whenever we mention any functions or function classes. We write $\varphi(\bar{x}) \mapsto y$ when y is a (possible) output of $\varphi(\bar{x})$ and $y < \varphi(\bar{x})$ if there is some z such that $\varphi(\bar{x}) \mapsto z$ and $y < z$.

Our computation model is the register machine (RM) which consists of a finite set of states $d_1, \ldots, d_\ell$, a finite set of registers $\pi_0, \ldots, \pi_m$, and a finite set of instructions of the following form:

Increment $d_i I_0 \pi_j \pi_{j'} d_k$ and $d_i I_1 \pi_j \pi_{j'} d_k$: When in state d_i, if π_j holds x, store $s_0 x$ or $s_1 x$ in $\pi_{j'}$ and change to state d_k.

Decrement $d_i D \pi_j \pi_{j'} d_k$: When in state d_i, if π_j holds x, store $p(x)$ in $\pi_{j'}$ and change to state d_k.

Transfer $d_i T \pi_j d_{k_0} d_{k_1}$: When in state d_i, if the contents of π_j are either 0 or $s_0 x$, change to state d_{k_0} and otherwise change to state d_{k_1}.

Let M be a register machine. States of M for which there is at most one instruction are *deterministic* states; the others are *nondeterministic*. M is deterministic if all of its states are deterministic. For uniformity, we assume that all RM's have two distinguished states d_{acc} and d_{rej} for which there are no instructions (as well as possibly others). If M enters d_{acc} or d_{rej}, it is said to *accept* or *reject* its input accordingly; if it enters any state for which there is no instruction, it is said to *halt*. A computation of a RM M on input $\bar{x} = x_1, \ldots, x_n$ is started by storing x_i in π_i, entering state d_1, and then executing instructions in the natural (possibly non-deterministic) manner, halting if M enters a state for which there is no instruction. Define $M(\bar{x}) =_{\text{df}} \{y \mid \pi_0 \text{ contains } y \text{ when } M \text{ accepts } \bar{x}\}$. If M is deterministic, then $M(\bar{x})$ has cardinality 0 or 1, and we say that $M(\bar{x})$ computes the partial function φ defined as follows: if $M(\bar{x})$ is empty, then $\varphi(\bar{x})$ is undefined; otherwise, $M(\bar{x}) = \{\varphi(\bar{x})\}$.[1] If M is non-deterministic and $M(\bar{x})$ is finite for all $\bar{x}$, then we say that M computes the partial multifunction φ given by $\varphi(\bar{x}) = M(x)$. A *configuration* of M is a list $d_i, x_0, \ldots, x_m$ specifying a state and contents of the registers. A *computation sequence* for M is a sequence of configurations $c_1, \ldots, c_n$ such that

(1) For all $i < n$ there is a transition of M from c_i to c_{i+1}; or
(2) There is some $d \leq n$ such that c_d is an accepting configuration, there is a transition of M from c_i to c_{i+1} for all $i < d$, and $c_j = c_d$ for all $d < j \leq n$ (in other words, we may repeat an accepting configuration c_d, even though there is no transition from c_d to itself).

We assume a coding of register machines and configurations satisfying the condition that if e is the code of M and z is the code of a computation sequence of length n for which every register in every configuration stores a number of length $\leq n$, then $n \leq |z| \leq \underline{\text{sqbd}}_e(n)$, where $\underline{\text{sqbd}}_e$ is polynomial-time computable in n. We also define a polynomial-time function $\underline{\text{result}}$ so that if z is the code of a computation sequence, $\underline{\text{result}}(z)$ is the contents of π_0 in the last configuration of z. We shall consistently conflate state, configurations, etc. with their corresponding codes.

We will consider the following classes of functions in this paper:

- **FP**: the class of total single-valued functions computable on a deterministic RM in time polynomial in the length of the input.
- **NPMV**: the class of pmf's computable on a nondeterministic RM in time

[1] Of course, this is an insignificant variation on the usual definition for a partial function to be computable on a deterministic RM.

polynomial in the length of the input. [2]

- **PR**: the primitive recursive functions.
- **REC**: the partial recursive functions.

As noted by Leivant [4], Turing Machines and register machines are polynomial-time reducible to each other, so our definitions of **FP** and **NPMV** correspond to the usual ones.

We will make use of the following operators on pmf's:

Composition φ is defined by *composition* from ψ, $\rho_1, \ldots, \rho_k$, written $\varphi(\bar{x}) = \psi(\bar{x}, \rho_1(\bar{x}), \ldots, \rho_k(\bar{x}))$, when

$$\varphi(\bar{x}) \mapsto y \Leftrightarrow \exists \bar{z}.\rho_1(\bar{x}) \mapsto z_1 \wedge \cdots \wedge \rho_k(\bar{x}) \mapsto z_k \wedge \psi(\bar{x}, \bar{z}) \mapsto y.$$

Bounded Recursion on Notation φ is defined by *bounded recursion on notation* (*brn*) from ψ, ρ_0, ρ_1, and θ, when φ satisfies

$$\begin{aligned} \varphi(\bar{x}, 0) \mapsto y &\Leftrightarrow \psi(\bar{x}) \mapsto y \\ \varphi(\bar{x}, s_i z) \mapsto y &\Leftrightarrow \exists u.\varphi(\bar{x}, z) \mapsto u \wedge \rho_i(\bar{x}, z, u) \mapsto y \\ \varphi(\bar{x}, z) &\leq \theta(\bar{x}, z) \qquad (\text{all } \bar{x},\ z) \end{aligned}$$

when

$$\varphi(\bar{x}, s_i z) \mapsto y \Leftrightarrow \exists u.\varphi(\bar{x}, z) \mapsto u \wedge \rho(\bar{x}, z, u) \mapsto y$$

and for all $\bar{x}$ and z, one has that $\varphi(\bar{x}, z) \leq \theta(\bar{x}, z)$.

Bounded Weak Minimization φ is defined by *bounded weak minimization* (*bwm*) from ψ and θ, written $\varphi(\bar{x}) = \mu z{<}\theta(\bar{x}).\psi(\bar{x}, z) = 0$, when

$$\varphi(\bar{x}) \mapsto y \Leftrightarrow y < \theta(\bar{x}) \wedge \psi(\bar{x}, y) \mapsto 0 \wedge \forall z{<}|y| \exists t{>}0.\psi(\bar{x}, z) \mapsto t.$$

If $\varphi(\bar{x}) \mapsto y$, we say that $y < \theta(\bar{x})$ is *weak minimal* such that $\psi(\bar{x}, y) = 0$.

Bounded Witnessing φ is defined by *bounded witnessing* from ψ and θ, written $\varphi(\bar{x}) = Wz{<}\theta(\bar{x}).\psi(\bar{x}, z) = 0$, when

$$\varphi(\bar{x}) \mapsto y \Leftrightarrow y < \theta(\bar{x}) \wedge \psi(\bar{x}, y) \mapsto 0.$$

Definition 1

(1) The class C is the smallest class of pmf's that contains the projections, zero, binary successors s_0 and s_1, and smash function $(x, y) \mapsto 2^{|x| \cdot |y|}$ and is closed under composition, brn, and bwm.

(2) The class D is the smallest class of pmf's that contains the projections, zero, binary successors s_0 and s_1, and smash function $(x, y) \mapsto 2^{|x| \cdot |y|}$ and is closed under composition, brn, and bounded witnessing.

[2] Following Papadimitriou [5], we assume that if M is a non-deterministic RM that runs in time $p(n)$, then on any input of length n, every computation path of M halts in $\leq p(n)$ steps.

Proposition 2 $\mathbf{FP} \subseteq C$ *and* $\mathbf{FP} \subseteq D$.

PROOF. Without bwm or bounded witnessing, both classes are defined as Cobham's characterization of **FP**. □

We will freely make use of polynomial-time computable predicates in the definitions of our functions; in such cases, we understand that formally we are referring to their characteristic functions, where $\chi_P(\bar{x}) = 1$ if $P(\bar{x})$ holds and $\chi_P(\bar{x}) = 0$ if not. Specifically, in a minimization of the form $\mu z < \theta(\bar{x}).P(\bar{x}, z)$, we mean $\mu z < \theta(\bar{x})[1 - \chi_P(\bar{x}, z) = 0]$.

Definition 3 *A pmf* φ *is* polynomially-bounded *(*poly-bounded *for short) if there is a polynomial* p *such that for all* $\bar{x}$*, if* $\varphi(\bar{x}) \mapsto y$*, then* $|y| \leq p(|\bar{x}|)$.

Proposition 4 *Every pmf in* C *and* D *is poly-bounded.*

PROOF. This is a straightforward proof by induction on the definitions of C and D. □

By this proposition, we may assume that the bounding functions used in bounded recursion on notation and bounded weak minimization are in fact polynomials.

3 Machine Characterization

Theorem 5 *Let* φ *be a pmf. The following are equivalent:*

(1) $\varphi \in C$*;*
(2) $\varphi \in D$*;*
(3) $\varphi \in \mathbf{NPMV}$.

PROOF. (1) $\Rightarrow$ (3): This is proved by induction on the definition of $\varphi \in C$. The initial functions of C are p-time computable, and hence in **NPMV**. Otherwise, assume that φ is defined from pmf's we already know to be in **NPMV**. The proof breaks into cases depending on how φ is defined from these pmf's. When a pmf ψ is given by the induction hypothesis, we assume it is computed by the nondeterministic RM M_ψ in time p_ψ, and similarly for other such pmf's.

Suppose $\varphi(x) = \psi(x, \rho(x))$. Then φ is computed by the machine M as follows. On input x, run M_ρ on x; if it rejects, M also rejects. Otherwise, if M_ρ accepts with output z, run M_ψ on input x, z; if it rejects, M rejects, and otherwise M accepts with output that of M_ψ. Clearly M computes φ and runs in time $p_\rho(n) + p_\psi(n, p_\rho(n))$.

Suppose that $\varphi(x, z)$ is defined by bounded recursion on notation from ψ, ρ_i, and $2^{q(|\cdot|,|\cdot|)}$, where q is a polynomial that is increasing in both arguments. Then for any z, we have $\varphi(x, z) \mapsto y$ iff there is a sequence $t_0, \ldots, t_{|z|}$ such that $\psi(x) \mapsto t_{|z|}$, for all $i < |z|$ we have $\rho_{z(i)}(x, \lfloor z/2^{i+1} \rfloor, t_{i+1}) \mapsto t_i$, and $t_0 = y$, where $z(i)$ is the i^{th} bit of z. Furthermore, for any such sequence and all i, we have $\varphi(x, \lfloor z/2^i \rfloor) \mapsto t_i$, so $t_i < 2^{q(|x|,|z|)}$. Thus to compute $\varphi(x, z)$, guess a sequence $t_0, \ldots, t_{|z|}$ with each $t_i < 2^{q(|x|,|z|)}$ and verify that the above condition holds for each element of the sequence by running M_ψ or M_{ρ_i} and comparing the output to the previous element in the sequence. If none of the verifications fails and $t_0 = y$, then accept and output y. Guessing the sequence takes $|z|q(|x|, |z|)$ time and the verification takes $|z|p(|x|, q(|x|, |z|))$ time.

Finally, suppose that $\varphi(x) = \mu y < \theta(x).\psi(x, y) = 0$. We compute $\varphi(x)$ as follows. Run M_θ on input x. If it rejects, then reject, and otherwise guess y smaller than its output. Next verify that $\psi(x, y) \mapsto 0$ and reject if not. If the verification is successful, run M_ψ on input x, i for every $i < |y|$. If any of the computations accepts with output 0, reject, and otherwise accept with output y.

(3) $\Rightarrow$ (1): Suppose that φ is computed by the nondeterministic register machine M with code e that runs in time $p(n)$. Define the polynomial-time predicate $T(e, \bar{x}, z)$ to hold iff z codes an accepting computation sequence of M_e on input $\bar{x}$ of length $p(|\bar{x}|)$; the point here is to accept only sufficiently long computation sequences, so that any accepted sequence is automatically weak-minimal. We claim that $\varphi(\bar{x}) \mapsto a$ iff there is a weak-minimal $z < 2^{\underline{\text{sqbd}}_e(p(|\bar{x}|))}$ such that $T(e, \bar{x}, z)$ and $\underline{\text{result}}\, z = a$. The reverse direction is obvious. For the forward direction, suppose that z codes an accepting computation sequence of M on $\bar{x}$. By repeating the final configuration as many times as necessary, we can assume that the computation sequence has length $p(|\bar{x}|)$ so that $T(e, \bar{x}, z)$ holds. We need to show that z is weak-minimal, so suppose that $w < |z|$. Since $z \leq 2^{\underline{\text{sqbd}}_e(p(|\bar{x}|))}$, this implies that $w < \underline{\text{sqbd}}_e(p(|\bar{x}|))$. For sufficiently large $\bar{x}$, we have that $\underline{\text{sqbd}}_e(p(|\bar{x}|)) < 2^{p(|\bar{x}|)}$. By our assumption on codes of computation sequences (that the code of a sequence of length n must be at least 2^n), w cannot code a sequence of length $p(|\bar{x}|)$, and so $T(e, \bar{x}, w)$ cannot hold. Thus z must be weak-minimal, and therefore we can define φ by

$$\varphi(\bar{x}) = \underline{\text{result}}(\mu z < 2^{\underline{\text{sqbd}}_e(p(|\bar{x}|))}.T(e, \bar{x}, z)).$$

The proof that (2) and (3) are equivalent is essentially the same as the pre-

ceding, but without having to compensate for short computation sequences for the reverse direction. This was originally proved by Spreen [7]. □

Corollary 6 (Normal Form Theorem) *Every pmf $\varphi(\bar{x}) \in C$ can be written in the form $\rho(\mu z < p(\bar{x}).\psi(\bar{x}, z) = 0)$ for some $\rho, \psi \in \mathbf{FP}$.*

As an application of this equivalence, consider the following "intersection" and "union" operations on pmf's:

$$(\varphi \oplus \psi)(x) \mapsto y \Leftrightarrow \varphi(x) \mapsto y \vee \psi(x) \mapsto y$$
$$(\varphi \otimes \psi)(x) \mapsto y \Leftrightarrow \varphi(x) \mapsto y \wedge \psi(x) \mapsto y$$

We see that **NPMV** is closed under both operations as follows. If φ and ψ are computed by the RM's M_φ and M_ψ, respectively, then $(\varphi \oplus \psi)(x)$ is computed by nondeterministically choosing to run one of M_φ or M_ψ on x, and $(\varphi \otimes \psi)(x)$ is computed by running both M_φ and M_ψ and accepting exactly when both accept with the same output. By the Theorem, C and D must also be closed under these operations. Note that this latter fact is not so obvious, because composition is not such a simple operation when applied to partial functions. In particular, consider the composition $\varphi(\bar{x}) = \psi(\rho_0(\bar{x}), \rho_1(\bar{x}))$. To evaluate $\varphi(\bar{x})$, intuitively we must find some z_1 and z_2 such that $\rho_i(\bar{x}) \mapsto z_i$. This involves evaluating both ρ_i. But if, for example, $\rho_1(\bar{x})$ is undefined, then the entire composition is undefined. On the other hand, these intersection and union operators are well-defined even when $\varphi(x)$ or $\psi(x)$ is not defined.

4 Resource-free Characterization

We now recall Bellantoni and Cook's definition of safe recursion [1] in the context of pmf's. Arguments are separated into two types: normal (for which we use x, y, z) and safe (a, b, and c). One view of this distinction is that the normal arguments can be used to clock iterations, whereas the safe arguments can be used simply as bit stores, for which a polynomial number of the bits can be examined in a computation [1]. Another view is that the safe positions are safe for "large" input values; i.e., one can increase the size of these arguments without a significant increase in the computation time [2]. Yet another, more philosophically justified, view is given by Leivant [4]. He views the safe argument positions as being able to take data that has been somehow "impredicatively" defined. That is to say, if the definition of the data somehow assumes the totality of the domain on which the function is being defined, it is impredicative, and can only be used in a safe position. In particular, when a function is defined by recurrence as $f(s_i x, y) = g(x, y, f(x, y))$, the definition only makes sense when g is defined no matter what its third argument is—in

other words, the definition assumes the totality of f, and hence the domain of definition—and so this argument position must be safe. We use a semicolon to delimit normal from safe arguments: $\varphi(x;a)$ takes a single normal argument x and a single safe argument a.

Now consider the following operations on pmf's, where $x \bmod 2$ is the value of the low-order bit of x:

Safe Composition φ is defined by *safe composition* from ψ, $\theta_1, \ldots, \theta_k$, and $\rho_1, \ldots, \rho_\ell$, written $\varphi(\bar{x};\bar{a}) = \psi(\theta_1(\bar{x};), \ldots, \theta_k(\bar{x};); \rho_1(\bar{x};\bar{a}), \ldots, \rho_\ell(\bar{x};\bar{a}))$, when

$$\varphi(\bar{x};\bar{a}) \mapsto c \Leftrightarrow \exists \bar{w}\bar{b}. \bigwedge_{i=1}^{k} [\theta_i(\bar{x};) \mapsto w_i] \wedge \bigwedge_{j=1}^{\ell} [\rho_j(\bar{x};\bar{a}) \mapsto b_j] \wedge \psi(\bar{w};\bar{b}) \mapsto c.$$

Safe Recursion on Notation φ is defined by *safe recursion on notation* (*srn*) from ψ and ρ_i, written

$$\begin{aligned} \varphi(\bar{x}, 0;\bar{a}) &= \psi(\bar{x};\bar{a}) \\ \varphi(\bar{x}, s_i z;\bar{a}) &= \rho_i(\bar{x}, z;\bar{a}, \varphi(\bar{x}, z;\bar{a})) \end{aligned}$$

when

$$\varphi(\bar{x}, s_i z;\bar{a}) \mapsto c \Leftrightarrow \exists b.\varphi(\bar{x}, z;\bar{a}) \mapsto b \wedge \rho_i(\bar{x}, z;\bar{a}, b) \mapsto c.$$

Safe Weak Minimization φ is defined by *safe weak minimization* (*swm*) from ψ, written $\varphi(\bar{x};\bar{a}) = \mu b.\psi(\bar{x};\bar{a},b) \bmod 2 = 0$, when

$$\varphi(\bar{x};\bar{a}) \mapsto c \Leftrightarrow \psi(\bar{x};\bar{a},c) \bmod 2 \mapsto 0 \wedge \forall d{<}|c|.\psi(\bar{x};\bar{a},d) \bmod 2 \mapsto 1$$

(we write $\psi(\bar{x};\bar{a},c) \bmod 2 \mapsto 0$ if there is z such that $\psi(\bar{x};\bar{a},c) \mapsto z$ and $z \bmod 2 = 0$; this is just the usual notion of pmf composition of φ and $\lambda z.z \bmod 2$).

It would be natural to say that φ is defined by *safe witnessing* from ψ when

$$\varphi(\bar{x};\bar{a}) \mapsto c \Leftrightarrow \psi(\bar{x};\bar{a},c) \bmod 2 \mapsto 0.$$

But with no minimization or bounding requirements, this would lead to pmf's with an infinite number of outputs for a fixed input (e.g., if ψ is constantly 0), so we do not consider such a scheme.

As we mentioned in the introduction, Bellantoni [1] defines the minimization scheme $\mu^t b.f(\bar{x};\bar{a},b) \bmod 2 = 0$ to be the least b such that $f(\bar{x};\bar{a},b) \bmod 2 = 0$ if such a b exists and 0 otherwise.[3] Note that if f is total, then so is $\mu^t b.f(\bar{x};\bar{a},b) \bmod 2 = 0$, and so Bellantoni can add this scheme to his characterization of **FP** (base functions plus safe composition and safe recursion

[3] Actually, if there is such a b, Bellantoni's minimization outputs $s_1 b$.

on notation) to define a new class of total functions. He then proves that under the natural definition of "i applications of minimization," the functions definable with $i+1$ applications of minimization are exactly those that are computable in polynomial time from a Σ_i^p oracle. He also shows that the same characterization holds for a bounded version of this total minimization scheme (no notion of safety). Furthermore, we note that if we add an unbounded, total minimization operator to primitive recursion, then it is easy to show that the functions definable with $i+1$ applications are exactly those recursive in a Σ_i oracle (for a proof, one can simply remove the bounds from Bellantoni's proof of the bounded version). Thus this total minimization operator (bounded or safe) behaves in a manner exactly analogous to the corresponding operator for recursive functions. However, for exactly this reason, it does not capture the appropriate minimization that is analogous to generating the partial recursive functions, which is our aim in this paper.

However, safe weak minimization as stated also does not allow us to reach this goal. Intuitively, the reason is as follows. Define the (single-valued) pmf φ by

$$\varphi(x;a) = \begin{cases} 0, & |x| \leq |a| \\ 1, & |x| > |a| \end{cases}$$

Now define $\psi(x;) = \mu b.\varphi(x;b) = 0$. Fix any x; for simplicity, assume x has the form 2^z for some z. The least b such that $\varphi(x;b) = 0$ is x. The crucial point is that *any* $x \leq a \leq 2^x$ satisfies $\varphi(x;a) = 0$ and is weak-minimal, since $a \leq 2^x$ and $w < |a|$ implies $|w| < |x|$. In particular, the set of outputs of $\varphi(x;)$ has cardinality approximately 2^x. But any **NPMV** pmf has a set of outputs of cardinality $\leq 2^{p(|x|)}$ for some polynomial p, so ψ cannot be in **NPMV**. [4]

The solution is to use only the "already computable" low-order bits of the result of minimization. First define $a \underline{\bmod} v$ to be the $|v|$ low-order bits of a; Bellantoni gives a definition with v normal and a safe in [1]. Now consider the following scheme:

Limited Safe Weak Minimization φ is defined by *limited safe weak minimization* (*lswm*) from ψ and ρ when

$$\varphi(\bar{x};\bar{a}) = (\mu b.\psi(\bar{x};\bar{a},b) \bmod 2 = 0) \underline{\bmod} \rho(\bar{x};).$$

Note that we are not actually limiting the scope of the minimization operator here; we are simply throwing away data from its results. We return to this idea momentarily.

We now show that limited safe weak minimization is the appropriate operator

[4] We would like to thank one of the referees for pointing out (essentially) this example.

for this goal.

Definition 7 *B_0 is the following set of functions:*

(1) The constant 0 (nullary) function.
(2) Projections: $\pi_i^{n;m}(w_1, \ldots, w_n; w_{n+1}, \ldots, w_{n+m}) = w_i$.
(3) Binary successors: $s_0(;a) = 2a$, $s_1(;a) = 2a+1$.
(4) Predecessor: $\underline{\text{pred}}(;0) = 0$, $\underline{\text{pred}}(;s_i a) = a$.
(5) Conditional: $\underline{\text{cond}}(;a,b,c) = \underline{\text{if}}\ a \bmod 2 = 0\ \underline{\text{then}}\ b\ \underline{\text{else}}\ c$.

Definition 8 *The class C_{safe} is the smallest class of pmf's containing B_0 that is closed under safe composition, srn, and lswm. The class $C_{\text{safe}}^{\text{nml}}$ consists of those functions of C_{safe} with only normal arguments.*

The following definition, lemma, and proposition are taken essentially from [1]. Taken together, they formalize the intuition that for any pmf φ there is a polynomial q such that the $|w|$ bits of the output of $\varphi(\bar{x};\bar{a})$ depend only on the $q(|w|)$ bits of the safe arguments $\bar{a}$. For a sequence $\bar{a} = a_1, \ldots, a_k$, we write $\bar{a}\ \underline{\text{mod}}\ v$ for $a_1\ \underline{\text{mod}}\ v, \ldots, a_k\ \underline{\text{mod}}\ v$. We only use this for vector notation; in particular, $\bar{a}, b\ \underline{\text{mod}}\ v = a_1, \ldots, a_k, b\ \underline{\text{mod}}\ v$.

Definition 9 *Let $\varphi(\bar{x},\bar{a})$ be a pmf (note that we do not separate the arguments into normal and safe here) and let q be a polynomial.*

(1) φ is poly-checking on $\bar{x}$ with threshold q *if for all w and all v satisfying $|v| \geq q(|\bar{x}|)+|w|$ we have $\varphi(\bar{x}, \bar{a}\ \underline{\text{mod}}\ v) \mapsto y \underline{\text{mod}} w \Leftrightarrow \varphi(\bar{x},\bar{a}) \mapsto y \underline{\text{mod}} w$.*
(2) A pmf $\varphi(\bar{x},\bar{a})$ is polymax bounded on $\bar{x}$ by *q if for all $\bar{x}$ and $\bar{a}$, if $\varphi(\bar{x},\bar{a})$ is defined, then $|\varphi(\bar{x},\bar{a})| \leq q(|\bar{x}|) + \max\{|\bar{a}|\}$.*

It might be argued that by automatically limiting the minimization operator, we enter a "grey area" of implicit computational complexity, as the definition is reminiscent of limited primitive recursion. However, this limitation is of a different character. For any sort of minimization operator, when looking at potential witnesses b for which $\psi(\bar{x};\bar{a},b) = 0 \bmod 2$, we can examine only an amount of information that is polynomial in $|b|$; otherwise the verification that b is acceptable could not be done in polynomial time.[5] Suppose that b is the least such witness, and let c satisfy $b < c < 2^b$. Examining information that is polynomial in $|c|$ probably means examining values $< b$ (as is the case in the counterexample above). By minimality of b, we see that c is also possibly a witness, which implies that there are possibly an exponential number of witnesses. However, we can view the pmf's of C_{safe} as being stratified by the number of uses of lswm. Consider a function that uses lwsm exactly once, say $\varphi(\bar{x};\bar{a}) = (\mu b.\psi(\bar{x};\bar{a},b) = 0 \bmod 2)\ \underline{\text{mod}}\ \rho(\bar{x};)$. In [1], Bellantoni shows that the functions definable from the base functions B_0 using safe composition

[5] Note that Spreen's operators in [7] do not attempt a notion of minimization.

and srn are poly-checking on their normal arguments. Of course, ψ is such a function. So even though there may be exponentially large witnesses b that $\psi(\bar{x};\bar{a},b) = 0 \bmod 2$, only a polynomial number of any of them are actually used in the computation. Thus intuitively our limiting operator only returns the portion of the witness actually used. Of course, this rationale can only be extended if we continue to generate pmf's that are poly-checking, which is what we now prove.

Proposition 10 *If $\varphi(\bar{x};\bar{a}) \in C_{\text{safe}}$, then φ is poly-checking and polymax bounded on $\bar{x}$.*

PROOF. By induction on the definition of C_{safe}. The cases other than lswm are proved as in [1]. Suppose that $\varphi(\bar{x};\bar{a}) = (\mu b.\psi(\bar{x};\bar{a},b) = 0 \bmod 2)\underline{\text{mod}}\rho(\bar{x};)$ and that ψ and ρ are both polymax bounded by $q(|\bar{x}|)$, which also acts as a threshold witnessing that they are poly-checking. For any $\bar{x}$ and $\bar{a}$ we have that if $\varphi(\bar{x};\bar{a}) \mapsto b$, then $|b| \leq |\rho(\bar{x};)| \leq q(|\bar{x}|)$, so φ is polymax bounded by q. Now let $p(n) = q(n) + 2$, fix any $\bar{x}$, $\bar{a}$, and w and choose v with $|v| \geq p(|\bar{x}|) + |w|$. Define the sets

$$A =_{\text{df}} \{b \bmod \rho(\bar{x};) \mid b \text{ weak-minimal s.t. } \psi(\bar{x};\bar{a} \bmod v, b) = 0 \bmod 2\}$$
$$B =_{\text{df}} \{b \bmod \rho(\bar{x};) \mid b \text{ weak-minimal s.t. } \psi(\bar{x};\bar{a} \bmod v, b \bmod v) = 0 \bmod 2\}.$$

It suffices to show that $A = B$, for then we have

$$\begin{aligned}
&\varphi(\bar{x};\bar{a}\underline{\text{mod}}v) \bmod w \\
&\qquad = ((\mu b.\psi(\bar{x};\bar{a} \bmod v, b) = 0 \bmod 2) \bmod \rho(\bar{x};)) \bmod w \\
&\qquad = ((\mu b.\psi(\bar{x};\bar{a} \bmod v, b \bmod v) = 0 \bmod 2) \bmod \rho(\bar{x};)) \bmod w \\
&\qquad = \varphi(\bar{x};\bar{a} \bmod v, b \bmod v) \bmod w.
\end{aligned}$$

Since q is a threshold for ψ we have that

$$\psi(\bar{x};\bar{a},b) \bmod 1 = \psi(\bar{x};\bar{a} \bmod v, b \bmod v) \bmod 1.$$

Now suppose that $c = b \bmod \rho(\bar{x};) \in A$; we must show that $c \in B$. By the previous comment, we have that $\psi(\bar{x};\bar{a} \bmod v, b \bmod v) = 0 \bmod 2$. Furthermore, if $d < |b|$, then $d \bmod v < d < |b|$ so by weak minimality of b we have that $\psi(\bar{x};\bar{a} \bmod v, d \bmod v) \neq 0 \bmod 2$, so $c \in B$. Now suppose that $c = b \bmod \rho(\bar{x};) \in B$. Set $b' =_{\text{df}} b \bmod v$; we first show that $b' \bmod \rho(\bar{x};) \in A$. By definition, $\psi(\bar{x};\bar{a} \bmod v, b') = 0 \bmod 2$. If $d < |b'|$, then $d < |b|$, so $\psi(\bar{x};\bar{a} \bmod v, d \bmod v) \neq 0 \bmod 2$ by weak-minimality of b. But if $d < |b'|$, then we also have that $|d| < |v|$, so $d \bmod v = d$ and therefore $\psi(\bar{x};\bar{a} \bmod v, d) \neq 0 \bmod 2$. Thus we conclude that $b' \bmod \rho(\bar{x};) \in A$. Since $|v| \geq |\rho(\bar{x};)|$ we have $b' \bmod \rho(\bar{x};) = (b \bmod v) \bmod \rho(\bar{x};) = b \bmod \rho(\bar{x};) = c$, and therefore we conclude that $c \in A$, completing the proof. $\square$

Theorem 11 *Let φ be a pmf. Then $\varphi \in$ **NPMV** iff $\varphi \in C^{\text{nml}}_{\text{safe}}$.*

PROOF. The proof that $\varphi \in$ **NPMV** implies $\varphi \in C^{\text{nml}}_{\text{safe}}$ follows the same lines as the proof of Thm. 5 that if $\varphi \in$ **NPMV** then $\varphi \in C$. What we must do is define a version of the Kleene T-predicate using safe recursion on notation that allows us to extract accepting computation sequences of a nondeterministic RM. The main idea is that we use srn to define functions $f(x; a)$ which "examine" $p(|x|)$ bits of a and return some value. In other words, normal arguments are used as clocks to examine safe values, which are used as "bit stores." With this in mind, define (characteristic functions of) the following predicates:

- $\underline{\text{mach}}(m; e) \Leftrightarrow$ the $|m|$ low-order bits of e code a description of a nondeterministic RM (given the coding, $\underline{\text{mach}}(m; e)$ actually examines $p(|m|)$ bits of e for some fixed polynomial p, a detail which we do not specify from this point onward).
- $\underline{\text{cfg}}(n; e, c) \Leftrightarrow \underline{\text{mach}}(|e|; e) \wedge$ the $|n|$ low-order bits of c code a configuration of the RM described by e.
- $\underline{\text{trans}}(n; e, c_1, c_2) \Leftrightarrow \underline{\text{mach}}(|e|; e) \wedge \underline{\text{cfg}}(n; e, c_1) \wedge \underline{\text{cfg}}(n; e, c_2) \wedge$ there is a transition from c_1 to c_2 according to the description e.
- $\underline{\text{cs}}(n, t; e, z) \Leftrightarrow \underline{\text{mach}}(|e|; e) \wedge |t| \leq |z| \wedge$ the $|t|$ low-order bits of z code a sequence of configurations $c_1, \ldots, c_k, \ldots, c_k$ of a RM such that for all $i < k$ we have $\underline{\text{trans}}(n; e, c_i, c_{i+1})$ and if c_k is repeated, then it codes an accepting configuration.
- $T^*(n, t; e, \bar{x}, z) \Leftrightarrow \underline{\text{cs}}(n, t; e, z) \wedge$ the first configuration of z is the initial configuration of the RM described by e on input $\bar{x}$ and the last configuration is an accepting state.

Now suppose that $\psi \in C$. By Thm. 5, ψ is computed by a nondeterministic RM with code e with time bounded by some polynomial $p(n)$. Since no more than $p(n)$ registers are accessed during a computation on input of length n, every configuration can be coded by a string whose length is also polynomial in n, say $q(n)$. Furthermore we have that the code of any computation sequence of length $\leq n$ is itself $\leq \underline{\text{sqbd}}_e(n;)$. Thus, arguing as in Thm. 5, if we define

$$\varphi(\bar{x};) =_{\text{df}} \underline{\text{result}}(\mu z.T^*(2^{q(|\bar{x}|)}, 2^{\underline{\text{sqbd}}_e(p(|\bar{x}|);)}; e, \bar{x}, z)),$$

then $\varphi \in C^{\text{nml}}_{\text{safe}}$ and $\psi(\bar{x}) = \varphi(\bar{x};)$.

For the reverse direction, one proves by induction that if $\varphi(\bar{x}; \bar{a}) \in C_{\text{safe}}$, then $\varphi \in C$ using the polymax bound on φ given by Prop. 10. The only cases of real interest are when φ is defined by safe recursion on notation or safe weak minimization. If φ is defined by srn from ψ and ρ_i, then the same recursion on notation defines φ in C. Furthermore, if φ is polymax bounded by q on $\bar{x}$, then for all $\bar{x}$ and $\bar{a}$ for which $\varphi(\bar{x}; \bar{a})$ is defined we have

$\varphi(\bar{x};\bar{a}) \leq 2^{q(|\bar{x}|)+\sum_i |a_i|}$. Since the bound is polynomial-time computable, the recursion on notation is bounded by a function definable in C. Suppose that $\varphi(\bar{x};\bar{a}) = (\mu b.\psi(\bar{x};\bar{a},b) = 0 \bmod 2) \underline{\bmod}\, \rho(\bar{x};)$. Let $q(n)$ serve as a polymax bound and threshold function for φ, ψ, and ρ, and let ψ' and ρ' be given by the induction hypothesis applied to ψ and ρ. Define the sets

$$A = \{b \,\underline{\bmod}\, \rho(\bar{x};) \mid b \text{ weak-minimal s.t. } \psi(\bar{x};\bar{a},b) = 0 \bmod 2\}$$
$$B = \{b \,\underline{\bmod}\, \rho'(\bar{x}) \mid b < 2^{q(|\bar{x}|)+\max\{|\bar{a}|\}+2} \text{ weak-minimal s.t. } \psi'(\bar{x},\bar{a},b) = 0 \bmod 2\}$$

It suffices to show that $A = B$, for then we can take

$$\varphi'(\bar{x},\bar{a}) = (\mu b < 2^{q(|\bar{x}|)+\max\{|\bar{a}|\}+3}.\psi'(\bar{x},\bar{a},b) = 0 \bmod 2) \,\underline{\bmod}\, \rho'(\bar{x})$$

where we note that the $\underline{\bmod}$ function is polynomial-time computable and therefore definable in C. Clearly $B \subseteq A$. Suppose that $b \,\underline{\bmod}\, \rho(\bar{x};) \in A$. Fix any v with $|v| = q(|\bar{x}|) + \max\{|\bar{a}|\} + 2$ and set $b' =_{\mathrm{df}} b \,\underline{\bmod}\, v$. Then

$$\psi(\bar{x};\bar{a},b \,\underline{\bmod}\, v) = \psi(\bar{x};\bar{a} \,\underline{\bmod}\, v, b \,\underline{\bmod}\, v) \,\underline{\bmod}\, 1 = \psi(\bar{x};\bar{a},b) \,\underline{\bmod}\, 1 = 0$$

because $\bar{a}\underline{\bmod}v = \bar{a}$ and q is a polychecking threshold for ψ. Thus we have that $\psi'(\bar{x},\bar{a},b') = \psi(\bar{x};\bar{a},b') = 0 \bmod 2$ by the induction hypothesis. Furthermore $|b'| \leq |v| = q(|\bar{x}|) + \max\{|\bar{a}|\} + 2$. Finally, suppose that $d < |b'|$; then $d < |b|$, so $\psi'(\bar{x},\bar{a},d) = \psi(\bar{x};\bar{a},d) \neq 0 \bmod 2$ by weak minimality of b. Thus since $|v| \geq |\rho(\bar{x})|$, we have that $b\underline{\bmod}\rho(\bar{x};) = (b\underline{\bmod}v)\underline{\bmod}\rho(\bar{x};) = b'\underline{\bmod}\rho'(\bar{x}) \in B$, completing the proof. $\square$

5 Nondeterministic Recursion

In [4, §5], Leivant discusses a notion of nondeterministic recursion and its safe (ramified) variant. We say that $f(\bar{x},z;\bar{a})$ is defined from the finite set Φ of multifunctions by nondeterministic safe recursion if

$$f(\bar{x},0;\bar{a}) \mapsto c \Leftrightarrow \exists g \in \Phi.g(\bar{x};\bar{a}) \mapsto c$$
$$f(\bar{x},s_i z;\bar{a}) \mapsto c \Leftrightarrow \exists g \in \Phi.\exists b.f(\bar{x},z;\bar{a}) \mapsto b \wedge g(\bar{x},z;\bar{a},b) \mapsto c$$

In other words, at each step of the computation of $f(\bar{x},z;\bar{a})$ we are allowed to "choose" a different recursion function from Φ. Note that if Φ consists of total multifunctions, then f is also a total multifunction. Let E be the smallest class of (total) multifunctions containing B_0 that is closed under safe composition and nondeterministic safe recursion; E^{nml} consists of those functions of E that have only normal arguments. In this section we prove that E^{nml} consists of exactly the total multifunctions of **NPMV**. [6]

[6] Because of a technical subtlety, Leivant's proof sketch of this result does not carry through.

Lemma 12 *If $f(\bar{x};\bar{a}) \in E$, then f is polymax bounded on $\bar{x}$.*

PROOF. This is proved by induction on the definition of f and is essentially the same as the proof that the functions defined by safe recursion (no minimization) are all polymax bounded. $\square$

Theorem 13 *Let f be a total multifunction. Then $f \in E^{\mathrm{nml}}$ iff $f \in$ **NPMV**.*

PROOF. For the forward direction, prove by induction that if $f(\bar{x};\bar{a}) \in E$, then $f(\bar{x},\bar{a}) \in C$, and therefore $f \in$ **NPMV** by Thm. 5. This is essentially the same as the proof of the forward direction of Thm. 11. Of course, the only case that needs consideration is when f is defined by nondeterministic safe recursion from Φ. Say that $\Phi = \Phi_0 \cup \Phi_1$, where the multifunctions of Φ_0 are used to compute $f(\bar{x},0;\bar{a})$ and the multifunctions of Φ_1 are used to compute $f(\bar{x},s_i z;\bar{a})$. Then we define f in C by

$$f(\bar{x},0,\bar{a}) = \left(\bigoplus_{g\in\Phi_0} g\right)(\bar{x},\bar{a})$$
$$f(\bar{x},s_i z,\bar{a}) = \left(\bigoplus_{g\in\Phi_1} g\right)(\bar{x},z,\bar{a},f(\bar{x},z,\bar{a}))$$

where $\oplus$ is the "union" operator defined after Cor. 6 (recall that we have only defined nondeterministic safe recursion from finite sets Φ, so Φ_0 and Φ_1 are finite). The polymax bound on f given by Lemma 12 gives a polynomial-time computable bound on the recursion, and so the recursion can be defined in C.

For the reverse direction, suppose that f is computed by a nondeterministic RM M that runs in time $q(n)$, uses states $d_1,\ldots,d_\ell$ and registers $\pi_0,\ldots,\pi_m$. For simplicity, assume that f is unary. We can assume that M has exactly two instructions for each nondeterministic state. We define a new register machine N that will behave similarly to M; the difference is that N will first guess its nondeterministic choices, then behave deterministically, referring to its original guesses whenever it encounters a nondeterministic state. Formally, we define N as follows:

(1) N uses registers $\pi_0,\ldots,\pi_{m+1}$, states $d_1,\ldots,d_\ell$, and new states $\bar{t}$ and d_{a0}, d'_{a0}, d_{a1}, and d'_{a1} for every nondeterministic state d_a of M.
(2) From the initial state, N nondeterministically constructs y in π_{m+1} with length $\leq q(|\pi_1|+\cdots+|\pi_n|)$ using the states $\bar{t}$.
(3) Otherwise, N is exactly the same as M, except that for every nondeterministic state with instructions $d_a\bar{D}_0$ and $d_a\bar{D}_1$ we replace these instruc-

tions with the following:

$$d_a T \pi_{m+1} d_{a0} d_{a1}$$
$$d_{a0} P \pi_{m+1} \pi_{m+1} d'_{a0}$$
$$d_{a1} P \pi_{m+1} \pi_{m+1} d'_{a1}$$
$$d'_{a0} \bar{D}_0$$
$$d'_{a1} \bar{D}_1$$

Thus, when N enters state d_a, it transitions to state d_{ai}, where i corresponds to the low-order bit of the contents of π_{m+1}, deletes that bit, and then executes either the first or second instruction that M would execute from state d_a.

Note that the only nondeterministic instructions are now those that start with one of the states from $\bar{t}$. Then N also computes $f(\bar{x})$ by executing exactly $q(|x_1| + \cdots + |x_n|)$ nondeterministic steps that change the contents of only π_{m+1}, then acting deterministically. To simulate N using nondeterministic safe recursion:

(1) Define transition functions $\tau_j(; s, a_0, \ldots, a_{m+1})$ for $j = -1, \ldots, m+1$ as in [4], but do not define them for the states $\bar{t}$ (more accurately, the definition for the states $\bar{t}$ is irrelevant). For $j \geq 0$, τ_j gives the contents of the j^{th} register after the transition from state (with code) s (for which there is only one possibility); τ_{-1} gives the next state. The τ_j can be defined with just the use of conditionals, successors, and predecessors, and hence can take all safe arguments.

(2) Define the functions σ_j by simultaneous safe recursion so that $\sigma_j(y_1, \ldots, y_j, x; d, a_0, \ldots, a_{m+1})$ is the contents of π_0 after executing $|y_1| \times \cdots \times |y_j| + |x|$ steps starting at state d and with a_i in register π_i; see [4] for details. Leivant also shows there that simultaneous safe recursion is reducible to safe recursion, so we can without loss of generality assume we use the latter.

(3) Define the function $\underline{\text{guess}}(z;)$ using nondeterministic safe recursion by

$$\underline{\text{guess}}(0;) = \begin{cases} 0 \\ 1 \end{cases} \qquad \underline{\text{guess}}(s_i z;) = \begin{cases} s_0(\underline{\text{guess}}(z;)) \\ s_1(\underline{\text{guess}}(z;)) \end{cases}$$

In other words, $\underline{\text{guess}}$ is defined from the set of functions containing the constant 0 and 1 functions and s_0 and s_1.

Just as in [4], if $q(n) \leq n^k$ (where $q(n)$ is the running time of M), then f is represented in E^{nml} as

$$f(x;) = \sigma_{m+1,k}(x, \ldots, x, 0, s_1, x, 0, \ldots, 0, \underline{\text{guess}}(2^{k \cdot |x|};)).$$

□

Examining the proof of Thm. 13, we actually proved that if $f \in E^{\mathrm{nml}}$, then f can be defined in C using only the schemes of composition and bounded recursion on notation, along with the $\oplus$ operator. So if we let C' be the smallest class of (total) multifunctions containing the same base functions as C that is closed under composition, brn, and $\oplus$, then we have proved:

Theorem 14 *Let f be a total multifunction. Then $f \in C'$ iff $f \in$* **NPMV**.

References

[1] S. Bellantoni. Predicative recursion and the polytime hierarchy. In *Feasible Mathematics II (Ithaca, NY, 1992)*, pages 15–29. Birkhäuser Boston, Boston, MA, 1995.

[2] S. Bellantoni and S. Cook. A new recursion-theoretic characterization of the polytime functions. *Comput. Complexity*, 2(2):97–110, 1992.

[3] A. Cobham. The intrinsic computational difficulty of functions. In *Logic, Methodology and Philos. Sci. (Proc. 1964 Internat. Congr.)*, pages 24–30. North-Holland, Amsterdam, 1965.

[4] D. Leivant. Ramified recurrence and computational complexity I: Word recurrence and poly-time. In *Feasible Mathematics II (Ithaca, NY, 1992)*, pages 320–343. Birkhäuser Boston, Boston, MA, 1995.

[5] C. H. Papadimitriou. *Computational Complexity*. Addison-Wesley Publishing Company, Reading, MA, 1994.

[6] A. L. Selman. Much ado about functions. In *Eleventh IEEE Conference on Computational Complexity*, pages 198–212. 1996.

[7] D. Spreen. On functions computable in nondeterministic polynomial time: Some characterizations. In *Computer Science Logic (Karlsruhe, 1987)*, pages 289–303. Springer-Verlag, Berlin, 1988.

[8] D. Spreen and H. Stahl. On the power of single-valued nondeterministic polynomial time computations. In *Computation Theory and Logic*, pages 403–414. Springer-Verlag, Berlin, 1987.