On Closure Properties of #P in the Context of PF ∘ #P

Mitsunori Ogiwara
Dept. Computer Science
Univ. Electro-Communications
Tokyo 182, JAPAN
(ogiwara@cs.uec.ac.jp)

Thomas Thierauf
Fakultät für Informatik
Universtät Ulm
D-7900 Ulm, Germany
(thierauf@informatik.uni-ulm.de)

Seinosuke Toda
Dept. Computer Science
Univ. Electro-Communications
Tokyo 182, JAPAN
(toda@cs.uec.ac.jp)

Osamu Watanabe
Dept. Computer Science
Tokyo Institute of Technology
Tokyo 152, JAPAN
(watanabe@cs.titech.ac.jp)

Abstract

For any operator τ, we say that #P is closed under τ in context PF ∘ #P, *if for every* $f \in$ #P, *and for every* $h \in$ PF, $h \circ \tau[f]$ *also belongs to* PF ∘ #P. *For several operators* τ *on* #P *functions, it is shown that the closure property of* #P *w.r.t.* τ *in context* PF ∘ #P *is closely related to the relation between* $P^{\#P[1]}$ *and higher classes such as* PH^{PP} *and* PP^{PP}.

1 Introduction

Counting is one of the key notions in computation, and recently, various counting problems have received considerable attention (see, e.g., [Sch90]). Motivated by those counting problems, many complexity classes of counting functions and their related complexity classes of languages have been introduced and extensively studied. Typical such *counting classes* are function classes #P [Val79], spanP [KST89], and GapP [FFK91], and languages classes PP [Gil77], ⊕P [PZ83], $C_=P$ [Sim75, Wag86a], CH [Tor88, Wag86b], and SPP [FFK91]. In most cases, questions concerning relationships between these classes have been left open. However, while absolute answers to such questions seem hard to get, it is still possible to obtain relative answers that help us to develop our intuition or understanding to these relationships. (*Cf.* Although the question whether P ≠? NP is still left open, through various research, we have now some understanding on how NP (seems) different from P.) In this paper, we propose a structural approach to extend our understanding on relationships between counting classes.

Our approach is to study closure properties of function classes under various settings. For example, it is easy to see that #P is *closed* under 1-increment, that is, for every #P function $f(x)$, the function $g(x) = f(x) + 1$ also belongs to #P. Such properties of #P —*closure properties* of #P— have played important roles, both explicitly and implicitly, for studying complexity classes related to #P [BHW91, BRS91, CH90, FR91]. We have found many closure properties that #P actually possesses, as well as many closure properties that #P does not seem to possess. Ogiwara and Hemachandra [OH91] have established a theory for closure properties for function classes and studied how likely it is that those function classes have basic closure properties. For example, they proved that #P is closed under subtraction by polynomial-time computable functions if and only if CH collapses to UP, and thereby showing why it is unlikely that #P is closed under subtraction by polynomial-time computable functions. On the other

*A part of the work was done while the first author was visiting Department of Computer Science, State University of New York at Buffalo, and the second, third, and fourth authors were visiting Department of Computer Science, Universtiy of Rochester. This research is supported in part by JSPS/NSF International Collaboration Grant JSPS-ENGR-207/NSF-INT-9116781. The first author is supported in part by NSF Grant grant CCR-9002292. The second author is supported in part by DFG Postdoctorial Stipend Th 472/1-1 and NSF grant CCR-8957604.

1063-6870/93 $03.00 © 1993 IEEE

hand, if, for some other reason, we do not believe that #P is closed under subtraction by polynomial-time computable functions, then this result can be interpreted as the explanation of why CH seems different from UP. In this paper, we take this approach and analyze our original questions on relationships of counting complexity classes in terms of closure properties.

For any operator (or, functor) τ and any function class $\mathcal{CF}$, let $\tau[\mathcal{CF}]$ denote the class of functions obtained by applying τ to some function in $\mathcal{CF}$. Then we ask whether $\tau[\mathcal{CF}]$ is larger than $\mathcal{CF}$, i.e., whether τ adds some computational power to $\mathcal{CF}$. Closure properties studied in [OH91] are essentially questions of this type. However, requirements such as $\tau[\mathcal{CF}] = \mathcal{CF}$ are generally too strong; even very primitive operators cannot satisfy such requirements. For example, as explained above, subtraction by polynomial-time computable function seems to create new functions when applying to #P functions, though it is a kind of "trivial" operation. Here we propose to consider closure properties in "higher levels", and introduce the notion called "closure properties in context". By "context", we mean a complexity class $\mathcal{C}(\mathcal{CF})$ (either a function class or a language class) defined by applying some operation to $\mathcal{CF}$. Now, for a context $\mathcal{C}(\mathcal{CF})$, we say that $\mathcal{CF}$ is *closed under τ in context $\mathcal{C}(\mathcal{CF})$*, if $\mathcal{C}(\mathcal{CF}) = \mathcal{C}(\tau[\mathcal{CF}])$, that is, applying τ to $\mathcal{CF}$ never creates a new class as long as we are concerned with the context $\mathcal{C}(\mathcal{CF})$. By considering "context", we may be able to absorb minor oscillation, thereby investigating more drastic change caused by operators. It turned out that this generalized notion of closure properties is meaningful and useful (in some cases) for analyzing relationships between complexity classes. We demonstrate it by showing results on closure properties of #P in context PF $\circ$ #P.

Among many possible contexts, in the present paper, we study closure properties of #P in context PF $\circ$ #P. One reason is that PF $\circ$ #P is perhaps the simplest and the smallest context next to "no context". But the main reason is that closure properties in context PF $\circ$ #P are closely related to the relation between $\mathrm{P}^{\#\mathrm{P}[1]}$ and its higher classes. The class $\mathrm{P}^{\#\mathrm{P}[1]}$ is equivalent to $\mathrm{P}_{tt}^{\#\mathrm{P}}$. That is, in the counting hierarchy, $\mathrm{P}^{\#\mathrm{P}[1]}$ corresponds to Θ_2^{P} of the polynomial-time hierarchy. It is known that $\mathrm{P}^{\#\mathrm{P}[1]}$ contains the polynomial-time hierarchy PH; indeed, $\mathrm{P}^{\#\mathrm{P}[1]}$ is strong enough to contain $\mathrm{PP}^{\mathrm{PH}}$ [Tod91]. On the other hand, we believe that $\mathrm{P}^{\#\mathrm{P}[1]}$ is not so strong to contain $\mathrm{PH}^{\mathrm{PP}}$ or $\mathrm{PP}^{\mathrm{PP}}$. Our study provides some explanations on why (or how) $\mathrm{P}^{\#\mathrm{P}[1]}$ seems less powerful than $\mathrm{PH}^{\mathrm{PP}}$ or $\mathrm{PP}^{\mathrm{PP}}$.

Consider the following two majority operators[1] $\mathrm{maj}_{\mathrm{w}}$ and maj. For any function $f : \Sigma^* \to \mathbf{N}$, $\mathrm{maj}_{\mathrm{w}}[f]$ and maj$[f]$ are functions that respectively takes the following value for each string x in Σ^*.

$\mathrm{maj}_{\mathrm{w}}[f](x)$: If more than half of $f(\langle 1, x\rangle)$, ..., $f(\langle 2^{|x|}, x\rangle)$ is y, then $\mathrm{maj}_{\mathrm{w}}[f](x) = y$; otherwise $\mathrm{maj}_{\mathrm{w}}[f](x)$ is *some value*. (Precisely speaking, "some value" should be appropriately chosen so that the following main result (1) holds.)

maj$[f](x)$: If more than half of $f(\langle 1, x\rangle)$, ..., $f(\langle 2^{|x|}, x\rangle)$ is y, then $\mathrm{maj}_{\mathrm{w}}[f](x) = y$; otherwise $\mathrm{maj}_{\mathrm{w}}[f](x) = ?$, where "?" is a symbol not in $\mathbf{N}$.

Intuitively, both $\mathrm{maj}_{\mathrm{w}}[f]$ and maj$[f]$ take the majority of the values of f on exponentially many different inputs if it exists. The only difference between these operators is in the treatment of the case where no majority exist, and from the following result, we know that the difference is crucial:

(1) #P is closed under $\mathrm{maj}_{\mathrm{w}}$ in context PF $\circ$ #P, and
(2) #P is closed under maj in context PF $\circ$ #P if and only if $\mathrm{P}^{\#\mathrm{P}[1]} = \mathrm{PP}^{\mathrm{PP}}$ (or equivalently, CH collapses to $\mathrm{P}^{\#\mathrm{P}[1]}$).

By these results, we conclude that the crucial factor that (possibly) separates $\mathrm{P}^{\#\mathrm{P}[1]}$ and $\mathrm{PP}^{\mathrm{PP}}$ is that one question to #P does not help to detect whether majority exists in exponentially many values of #P functions.n

2 Preliminaries

In this paper, we follow standard definitions and notations in computational complexity theory (see, e.g., [BDG88, BDG91]).

Throughout this paper, we fix our alphabet to $\Sigma = \{0, 1\}$; by a *string* we mean an element of Σ^*, and by a *language* we mean a subset of Σ^*. Natural numbers are encoded in Σ^* in an ordinary way, and let $\mathbf{N}$ denote the set of (encoded) natural numbers. For any string x, let $|x|$ denote the length of x, and for any set X, let $\|X\|$ denote the cardinality of X. For any language L, let $L^{\leq n}$ be the set $\{ x \in L : |x| \leq n \}$. The standard lexicographical ordering of Σ^* is used; that is, for strings $x, y \in \Sigma^*$, x is *lexicographically smaller* than y (denoted by $x < y$) if either (i) $|x| < |y|$, or (ii) $|x| = |y|$ and there exists $z \in \Sigma^*$ such that $x = z0u$ and $y = z1v$. We consider a standard one-to-one pairing function from $\Sigma^* \times \Sigma^*$ to Σ^* that is computable

[1] In our formal discussion in the next sections, we will consider a class of operators instead of one fixed operator.

and invertible in polynomial time. For inputs x and y, we denote the output of the pairing function by $\langle x, y\rangle$; this notation is extended to denote every n tuple. In the following, we simply write, e.g., $f(\langle i, x\rangle)$ as $f(i, x)$. In order to simplify discussion, we sometimes use another type of paring mechanism. For any string x and y, let $x\#y$ denote a string representing the ordered pair of (x, y) under some fixed rule. As above, we assume that $x\#y$ is polynomial-time computable from x and y, and polynomial-time invertible. Furthermore, we assume that for all (x, y) and (x', y') such that $|x| = |x'|$ and $|y| = |y'|$, we have $|x\#y| = |x'\#y'|$.

Throughout this paper we assume that functions are *total*.

For our computation model, we consider a standard Turing machine model. A machine is either deterministic or nondeterministic, and a deterministic machine is either an *acceptor* or a *transducer*, while a nondeterministic Turing machine is always an acceptor. We also consider a *query machine*, i.e., a machine that can ask queries to a given oracle. In this paper, an oracle is either a set or a function; for each oracle type, we adopt the standard query mechanism for our query machines. We assume that the nondeterministic branching degree at each guessing state is always two. For a nondeterministic machine M and any string x, let $acc_M(x)$ (resp., $rej_M(x)$, $total_M(x)$) denote the number of accepting paths (resp., the number of rejecting paths, the total number of paths) of M on input x.

In what follows, we define complexity classes used in the paper. (Let $\mathcal{C}$ be any class of either languages or functions, and we define classes relative to $\mathcal{C}$. Nonrelativized classes are defined similarly, or they can be defined as the ones with the empty oracle set.)

(1) $\mathrm{P}^{\mathcal{C}}$ is the class of languages L for which there exist some polynomial-time bounded deterministic query acceptor M and some oracle X in $\mathcal{C}$ such that $\forall x \in \Sigma^*\ [\ x \in L \leftrightarrow M^X \text{ accepts } x\ ]$.

(2) $\mathrm{NP}^{\mathcal{C}}$ is the class of languages L for which there exist some polynomial-time bounded nondeterministic query acceptor M and some oracle X in $\mathcal{C}$ such that $\forall x \in \Sigma^*\ [\ x \in L \leftrightarrow acc_{M^X}(x) > 0\ ]$.

(3) $\mathrm{PP}^{\mathcal{C}}$ is the class of languages L for which there exist some polynomial-time bounded nondeterministic acceptor M and some oracle X in $\mathcal{C}$ such that $\forall x \in \Sigma^*\ [\ x \in L \leftrightarrow acc_{M^X}(x) > total_{M^X}(x)/2\ ]$[2].

(4) $\mathrm{PF}^{\mathcal{C}}$ is the class of functions that are computable by some polynomial-time bounded query transducer with some oracle in $\mathcal{C}$.

[2]Throughout this paper, we use n/m to denote $\lfloor n/m \rfloor$.

(5) $\#\mathrm{P}^{\mathcal{C}}$ is the class of total functions $f : \Sigma^* \to \mathbf{N}$ for which there exist some polynomial-time bounded nondeterministic query acceptor M and some oracle X in $\mathcal{C}$ such that $\forall x \in \Sigma^*\ [\ f(x) = acc_{M^X}(x)\ ]$.

By restricting the way of asking queries, we can define various subclasses of the above classes. Here we define those that are used in our discussion.

(6) $\mathrm{P}^{\mathcal{C}[1]}$ (resp., $\mathrm{PF}^{\mathcal{C}[1]}$) is the class of languages accepted (resp., computed) by some polynomial-time deterministic query machine relative to some oracle in $\mathcal{C}$, where the query machine asks at most one query during a computation. (Such query machines are called *one-query machines*.)

Classes $\mathrm{P}^{\mathcal{C}[1]}$ and $\mathrm{PF}^{\mathcal{C}[1]}$ are sometimes denoted as $\mathrm{P}^{\mathcal{C}}_{1\text{-T}}$ and $\mathrm{PF}^{\mathcal{C}}_{1\text{-T}}$, respectively.

The polynomial-time hierarchy and the counting hierarchy are defined as follows.

(7) $\mathrm{PH}^{\mathcal{C}}$ is the class $\mathrm{NP}^{\mathcal{C}} \cup \mathrm{NP}^{\mathrm{NP}^{\mathcal{C}}} \cup \mathrm{NP}^{\mathrm{NP}^{\mathrm{NP}^{\mathcal{C}}}} \cup \ldots$ where classes $\mathrm{NP}^{\mathrm{NP}^{\mathcal{C}}}$, $\mathrm{NP}^{\mathrm{NP}^{\mathrm{NP}^{\mathcal{C}}}}$, ... are defined inductively.

(8) $\mathrm{CH}^{\mathcal{C}}$ is the class $\mathrm{PP}^{\mathcal{C}} \cup \mathrm{PP}^{\mathrm{PP}^{\mathcal{C}}} \cup \mathrm{PP}^{\mathrm{PP}^{\mathrm{PP}^{\mathcal{C}}}} \cup \ldots$ where classes $\mathrm{PP}^{\mathrm{PP}^{\mathcal{C}}}$, $\mathrm{PP}^{\mathrm{PP}^{\mathrm{PP}^{\mathcal{C}}}}$, ... are defined inductively.

Among various combination of query classes and oracle classes, we mainly consider the following language classes: $\mathrm{P}^{\#\mathrm{P}[1]}$, $\mathrm{P}^{\#\mathrm{P}}$, PH, $\mathrm{PH}^{\mathrm{PP}}$, $\mathrm{PP}^{\mathrm{PP}}$, and CH. For these classes, we know that $\mathrm{PP}^{\mathrm{PH}} \subseteq \mathrm{P}^{\#\mathrm{P}[1]}$ [Tod91] and $\mathrm{P}^{\#\mathrm{P}[1]} \subseteq \mathrm{P}^{\#\mathrm{P}} \subseteq \mathrm{PH}^{\mathrm{PP}} \subseteq \mathrm{P}^{\mathrm{PP}^{\mathrm{PP}}} \subseteq \mathrm{CH}$, and we conjecture that all inclusions are proper. Also it is easy to show that $\mathrm{P}^{\#\mathrm{P}[1]} = \mathrm{P}^{\#\mathrm{P}}_{\mathrm{tt}}$, where $\mathrm{P}^{\#\mathrm{P}}_{\mathrm{tt}}$ is the class of languages accepted by some polynomial-time bounded *nonadaptive* query machine with some oracle in #P.

In this paper, closure properties are discussed in the context of $\mathrm{PF} \circ \#\mathrm{P}$, where $\mathrm{PF} \circ \#\mathrm{P}$ is defined by $\{\, g : g = h \circ f \text{ for some } h \in \mathrm{PF} \text{ and } f \in \#\mathrm{P} \,\}$, where $h \circ f$ is the ordinary function composition of h and f. It is easy to show that $\mathrm{PF} \circ \#\mathrm{P} = \mathrm{PF}^{\#\mathrm{P}[1]}$.

2.1 Operator, Context, and Closure Property

An *operator* is formally defined as a functor mapping one function to another. (Recall that we assume functions are total.) For discussing closure properties, we consider a class of operators instead of a single operator. In the following, such operator classes are defined.

First, we define some functions on $\mathbf{N}^*$, where $\mathbf{N}^*$ is the set of tuples of $\mathbf{N}$. For any $(x_1, ..., x_m) \in \mathbf{N}^*$, the following functions return the following values.

$\max(x_1, \ldots, x_m)$
$=$ the largest number in $\{x_1, \ldots, x_m\}$,
$\mathrm{plu}(x_1, \ldots, x_m)$
$=$ the *set* of most commonly occurring numbers in $(x_1, \ldots, x_m)$,
$\mathrm{plu}^*(x_1, \ldots, x_m)$
$=$ the smallest value in $\mathrm{plu}(x_1, \ldots, x_m)$,
$\mathrm{mid}(x_1, \ldots, x_m)$
$=$ the $m/2$-th smallest value $x_{i_{m/2}}$ in the ordering $x_{i_1} \leq \cdots \leq x_{i_m}$ of $x_1, \cdots, x_m$,

$\mathrm{maj}(x_1, \ldots, x_m)$
$= \begin{cases} y, & \text{if more than half of } x_1, \ldots, x_m \text{ is } y, \\ ?, & \text{otherwise,} \end{cases}$
where ? is some symbol not in $\mathbf{N}$; and
$\mathrm{maj}^{\phi}_{\mathrm{w}}(x_1, \ldots, x_m)$
$= \begin{cases} y, & \text{if more than half of } x_1, \ldots, x_m \text{ is } y, \\ \phi(x_1, ..., x_m), & \text{otherwise,} \end{cases}$
where ϕ is any given function from $\mathbf{N}^*$ to $\mathbf{N}$.

Now, we define operator classes that we are interested in. Let us consider any function f on Σ^*, and define operator classes in terms of f. First we define basic operator classes.

$$\textbf{poly-pre}[f] = \{ f \circ h : h \in \mathrm{PF} \},$$
$$\textbf{poly-post}[f] = \{ h \circ f : h \in \mathrm{PF} \}, \text{ and}$$
$$\textbf{poly}[f] = \textbf{poly-pre}[f] \cup \textbf{poly-post}[f],$$

Next define operator classes **max**, **plu**, **mid**, **maj**, and $\mathbf{maj}^{\phi}_{\mathrm{w}}$ (where ϕ is any given function $\mathbf{N}^*$ to $\mathbf{N}$). Class $\mathbf{max}[f]$ is defined as the class of functions g of the form

$$g(x) = \max(f(1, x), \ldots, f(e(|x|), x))$$

for some function $e : \mathbf{N} \to \mathbf{N}$ such that $e(n)$ is computable within polynomial time w.r.t. n. (Note that from the time bound for computing e, we can assume that $e(n) \leq 2^{p(n)}$ for some polynomial p. Also recall that $f(i, x)$ means $f(\langle i, x \rangle)$.) Classes $\mathbf{plu}[f]$, $\mathbf{mid}[f]$, $\mathbf{maj}[f]$, and $\mathbf{maj}^{\phi}_{\mathrm{w}}[f]$ are defined similarly by using plu^*, mid, maj, and $\mathrm{maj}^{\phi}_{\mathrm{w}}$ respectively. We often omit ϕ from $\mathbf{maj}^{\phi}_{\mathrm{w}}$ when it is fixed from the context. In general, for any class of functions $\mathcal{CF}$ and any operator class τ, let $\tau[\mathcal{CF}]$ be the class $\cup\{ \tau[f] : f \in \mathcal{CF} \}$.

The concept concerning "closure properties" is defined formally as follows.

Definition 2.1. Let τ be any operator class (or, operator) on Σ^*, $\mathcal{CF}$ be any class of functions on Σ^*, and $\mathcal{C}(\mathcal{CF})$ be any complexity class (of functions or of languages). Then, $\mathcal{CF}$ is closed under τ in context $\mathcal{C}(\mathcal{CF})$ if $\mathcal{C}(\mathcal{CF}) = \mathcal{C}(\tau[\mathcal{CF}])$.

The notion of "being closed under τ without context" is a special case, where $\mathcal{CF}$ itself is used for $\mathcal{C}(\mathcal{CF})$.

Although, we define the notion in the most general way, in the following discussion, we only consider operator classes on $\mathbf{N}$ for τ, the class #P for $\mathcal{CF}$, and the class PF $\circ$ #P for $\mathcal{C}(\mathcal{CF})$.

It is clear that #P is closed under **poly-pre** (without context, or in any context). Thus, for discussing closure properties of #P, our choice of a pairing function $\langle \cdot, \cdot \rangle$ is not essential in the above definition. By a similar argument, it is easy to show that we lose no generality by using the operator maj (resp., $\mathrm{maj}_{\mathrm{w}}$) defined in the introduction, when considering whether #P is closed under **maj** (resp., $\mathbf{maj}_{\mathrm{w}}$). That is, #P is closed under maj (resp., $\mathrm{maj}^{\phi}_{\mathrm{w}}$) in context PF $\circ$ #P (or, in any other context) if and only if it is closed under **maj** (resp., $\mathbf{maj}^{\psi}_{\mathrm{w}}$) in the same context. (Note that we may need to consider ψ that is different from ϕ.)

3 Main Results

First, we show that with respect to some function $\phi : \mathbf{N}^* \to \mathbf{N}$, #P is closed under $\mathbf{maj}^{\phi}_{\mathrm{w}}$ in context PF $\circ$ #P.

Theorem 3.1. For some function $\phi : \mathbf{N}^* \to \mathbf{N}$, #P is closed under $\mathbf{maj}^{\phi}_{\mathrm{w}}$ in context PF $\circ$ #P.

Preceding to the present work, the result essentially same as above was shown in a quite different style in the submition version of [BTT92], while their result have been mentioned in a weak form in their current paper. For the sake of completeness, we include the full proof of the theorem.

For the proof, we need the following result due to Toda [Tod91].

Lemma 3.2. [Tod91] Let $T' \in \#\mathrm{P}$, q be a polynomial, and $m \geq 2$ be a natural number. Define $h(n) = 3n^4 + 4n^3$ and define

$$T(x) = \underbrace{h \circ \cdots \circ h}_{\lceil \log_2 q(|x|) \rceil}(T'(x)).$$

Then $T \in \#\mathrm{P}$, and for all x, it holds that

$$T'(x) \equiv 0 \pmod{m}$$
$$\Longrightarrow T(x) \equiv 0 \pmod{m^{q(|x|)}}, \text{ and}$$
$$T'(x) \equiv -1 \pmod{m}$$
$$\Longrightarrow T(x) \equiv -1 \pmod{m^{q(|x|)}}.$$

Proof of Theorem 3.1. In the following, we show one way, i.e., an algorithm, to compute a majority value *if it exists*. Although, whenever the algorithm finds no majority exists, it tells us this by giving, say "0" as output, it cannot always detect the nonexistence of majority. Thus, our function $\phi : \mathbf{N}^* \to \mathbf{N}$ is implicitly defined as the value that the algorithm yields when no majority exists.

Let $f \in \#\mathrm{P}$ and e be a polynomial-time computable function. We show how to compute $g(x) = \mathrm{maj}(f(1,x),...,f(e(x),x))$, if the majority exists (recall that the value of maj is "?", if no majority exists). Our goal is to construct a polynomial-time deterministic transducer M_0 that computes g by asking one query (per input) to some function f_0 in #P. Noting that $\mathrm{PF} \circ \#\mathrm{P} = \mathrm{PF}^{\#\mathrm{P}[1]}$, this clearly proves the theorem.

Let x be fixed and let n denote the length of x. Let p be a polynomial such that for all $i \leq e(x)$ we have $f(i,x) < 2^{p(n)}$ and $e(x) < 2^{p(n)}$.

For each i, let m_i denote the i-th prime number. By the Prime Number Theorem, $m_i \leq 2i^2$, for every i. Hence, primes $m_1,...,m_{p(n)}$ are computable within polynomial time in n. Also, note that $f(i,x) < m_1 \cdots m_{p(n)}$, for all $i \leq e(x)$.

For each $i \leq e(x)$, for each j, where $1 \leq j \leq p(n)$, and for each k, where $0 \leq k < m_j$, let

$$u'(i,x,j,k) = (f(i,x) + (m_j - k))^{m_j - 1}.$$

Clearly, u' is in #P. Note that, by the Fermat Theorem, for every $i \leq e(x)$, for every j, where $1 \leq j \leq p(n)$, and for every k, where $0 \leq k < m_j$, it holds that

- $f(i,x) \equiv k \pmod{m_j}$
  $\Longrightarrow u'(i,x,j,k) \equiv 0 \pmod{m_j}$, and
- $f(i,x) \not\equiv k \pmod{m_j}$
  $\Longrightarrow u'(i,x,j,k) \equiv 1 \pmod{m_j}$.

Our next step is to show the existence of a #P function u such that

- $u'(i,x,j,k) \equiv 0 \pmod{m_j}$
  $\Longrightarrow u(i,x,j,k) \equiv 0 \pmod{m_j^{p(n)}}$, and
- $u'(i,x,j,k) \equiv 1 \pmod{m_j}$
  $\Longrightarrow u(i,x,j,k) \equiv 1 \pmod{m_j^{p(n)}}$.

For this, we apply Lemma 3.2 with $T'(i,x,j,k) = u'(i,x,j,k) + (m_j - 1)$ and $q = p$, thereby getting $T \in \#\mathrm{P}$. Now $u = T + 1$ has the desired property.

Define

$$v(x,j,k) = \sum_{i \leq e(x)} u(i,x,j,k).$$

Clearly, v is in #P, and $v(x,j,k) \bmod m_j^{p(n)} = ||\{\, i \leq e(x) : f(i,x) \not\equiv k \pmod{m_j} \,\}||$, and therefore

$$e(x) - (v(x,j,k) \bmod m_j^{p(n)})$$
$$= ||\{\, i \leq e(x) : f(i,x) \equiv k \pmod{m_j} \,\}||.$$

Now, suppose $g(x) \neq ?$. Then, for each prime m_j there exists a unique $k_j < m_j$ such that $g(x) \equiv k_j \pmod{m_j}$, and therefore more than $e(x)/2$ of the i's satisfy $f(i,x) \equiv k_j \pmod{m_j}$, and consequently, for all $k < m_j$ that are different from k_j, we have that less than $e(x)/2$ of the i's satisfy $f(i,x) \equiv k \pmod{m_j}$. Therefore, for every j and k, where $1 \leq j \leq p(n)$ and $0 \leq k < m_j$, it holds that

$$g(x) \equiv k \pmod{m_j}$$
$$\Longleftrightarrow e(x) - (v(x,j,k) \bmod m_j^{p(n)}) > e(x)/2.$$

Therefore, if we know the values $v(x,j,k)$ for all j and k, where $1 \leq j \leq p(n)$ and $0 \leq k < m_j$, then, for every prime m_j, we can compute the unique $k_j < m_j$ such that $g(x) \equiv k_j \pmod{m_j}$ for every j; that is, $g(x)$ is uniquely determined as z such that $z < m_1 \cdots m_{p(n)}$ and $(\forall j,\ 1 \leq j \leq p(n))\ [\ z \equiv k_j \pmod{m_j}\ ]$. Thus, by the Chinese Remainder Theorem, one can easily get $g(x)$ from $k_1,...,k_{p(n)}$.

By standard methods, we can construct a function $f_0(x)$ in #P such that all the values $v(x,j,k)$ for all j and k, where $1 \leq j \leq p(n)$ and $0 \leq k < m_j$, are computable within polynomial time in n from $f_0(x)$. Now, it is clear that some polynomial-time deterministic query transducer M_0 can compute $g(x)$ by using $f_0(x)$ (that is given from the oracle f_0). □

Next we consider operators **maj**, **mid**, **plu**, and **max**, and show similar results. The following technical lemma due to Simon [Sim75] characterizes the acceptance condition for $\mathrm{C}_=\mathrm{P}$.

Lemma 3.3. [Sim75] Let $A \in \mathrm{C}_=\mathrm{P}$. Then there exist a polynomial q and a polynomial-time nondeterministic machine M such that for all x, the following conditions are satisfied: (i) $total_M(x) = 2^{q(|x|)}$, (ii) $acc_M(x) \leq 2^{q(|x|)-1}$, and (iii) $x \in A$ if and only if $acc_M(x) = rej_M(x) = 2^{q(|x|)-1}$.

The following characterizations of $\mathrm{PP}^{\mathrm{PP}}$ and $\mathrm{NP}^{\mathrm{PP}}$ are due to Torán [Tor88].

Proposition 3.4. [Tor88].
(1) A set L is in $\mathrm{PP}^{\mathrm{PP}}$ if and only if there exist a polynomial p and a set $A \in \mathrm{C}_{=}\mathrm{P}$ such that for every x, it holds that

$$\begin{aligned} &x \in L \\ &\iff \|\{\, w \in \Sigma^{p(|x|)} : x\#w \in A \,\}\| \geq 2^{p(|x|)-1} + 1, \\ &x \notin L \\ &\iff \|\{\, w \in \Sigma^{p(|x|)} : x\#w \in A \,\}\| \leq 2^{p(|x|)-1} - 1. \end{aligned}$$

(2) A set L is in $\mathrm{NP}^{\mathrm{PP}}$ if and only if there exist a polynomial p and a set $A \in \mathrm{C}_{=}\mathrm{P}$ such that for every x, it holds that

$$x \in L \iff \|\{\, w \in \Sigma^{p(|x|)} : x\#w \in A \,\}\| \geq 1.$$

In the results below, we sometimes have to compute a function g for a given input. The next lemma connects the complexity of the graph of g (i.e., set G below) with the complexity of actually computing g. (The proof is easy, and it is left to the reader.)

Lemma 3.5. Let e, f, and g be functions such that the length of $g(\, f(1,x), ..., f(e(x),x)\,)$ is bounded by some polynomial in $|x|$. Define sets G and B as follows.

$$\begin{aligned} G &= \{\, \langle x, y\rangle : g(\, f(1,x), ..., f(e(x),x)\,) = y \,\}, \text{ and} \\ B &= \{\, \langle x, j, b\rangle : \exists y \left[\begin{array}{l} \langle x, y\rangle \in G \text{ and} \\ \text{the } j\text{th bit of } y \text{ is } b \end{array} \right] \} \end{aligned}$$

Then B is in NP^{G} and g is in $\mathrm{PF}^{B}_{\mathrm{tt}}$. Additionally, if $B \in \mathrm{P}^{\#\mathrm{P}[1]}$, then $g \in \mathrm{PF}^{\#\mathrm{P}[1]}$, and if $B \in \mathrm{P}^{\#\mathrm{P}}$, then $g \in \mathrm{PF}^{\#\mathrm{P}}$.

Theorem 3.6.
(1) #P is closed under **maj** in context PF $\circ$ #P if and only if $\mathrm{P}^{\#\mathrm{P}[1]} = \mathrm{PP}^{\mathrm{PP}}$.
(2) #P is closed under **mid** in context PF $\circ$ #P if and only if $\mathrm{P}^{\#\mathrm{P}[1]} = \mathrm{PP}^{\mathrm{PP}}$.

Proof. Suppose that #P is closed under **maj** and **mid**, respectively, in context PF $\circ$ #P; we show show that every $\mathrm{PP}^{\mathrm{PP}}$ set belongs to $\mathrm{P}^{\#\mathrm{P}[1]}$. Consider any $L \in \mathrm{PP}^{\mathrm{PP}}$. Let A and p respectively be a set in $\mathrm{C}_{=}\mathrm{P}$ and a polynomial that characterize L as in Proposition 3.4 (1). Also let M and q be respectively a nondeterministic machine and a polynomial that characterize A as in Lemma 3.3.

Define functions f, g, and h. For each $x \in \Sigma^*$ and each $i \in \mathbf{N}$, if $1 \leq i \leq 2^{p(|x|)}$, then let $f(i,x) = acc_M(x\#w)$, where w is the i-th smallest string of length $p(|x|)$; otherwise, let $f(i,x) = 0$. By using f, g and h are defined as follows.

$$\begin{aligned} g(x) &= \mathrm{maj}(\, f(1,x), ..., f(2^{p(|x|)},x)\,), \text{ and} \\ h(x) &= \mathrm{mid}(\, f(1,x), ..., f(2^{p(|x|)},x)\,). \end{aligned}$$

Now consider any string x in Σ^*. Let l_x be the length of strings of the form $x\#w$ for some $w \in \Sigma^{p(|x|)}$. (Recall that $|x\#w|$ is the same for any $w \in \Sigma^{p(|x|)}$.) Then from the choice of A, p, M, and q, we have that $x \in L$ if and only if for more than half of the strings w of length $p(|x|)$, $x\#w$ is A, i.e., $acc_M(x\#w) = 2^{q(l_x)-1}$. Thus,

$$x \in L \iff g(x) = 2^{q(l_x)-1} \iff h(x) = 2^{q(l_x)-1}.$$

On the other hand, f is in #P, and thus, by our assumption, g and h are in PF $\circ$ #P = $\mathrm{PF}^{\#\mathrm{P}[1]}$; that is, there are polynomial-time one-query transducers computing g and h by using functions in #P as an oracle. Then it is easy to modify such transducers to acceptors M_g and M_h so that M_g (resp., M_h) accepts x if and only if $g(x) = 2^{q(l_x)-1}$ (resp., $h(x) = 2^{q(l_x)-1}$), i.e., if and only if $x \in L$. Therefore, in both cases, L is in $\mathrm{P}^{\#\mathrm{P}[1]}$.

Conversely, suppose that $\mathrm{PP}^{\mathrm{PP}} = \mathrm{P}^{\#\mathrm{P}[1]}$. Let any $f \in \#\mathrm{P}$ and $e \in \mathrm{PF}$ be fixed. We consider how to compute

$$\begin{aligned} g(x) &= \mathrm{maj}(\, f(1,x), ..., f(e(x),x)\,), \text{ and} \\ h(x) &= \mathrm{mid}(\, f(1,x), ..., f(e(x),x)\,). \end{aligned}$$

Define $G = \{\, \langle x, y\rangle : g(x) = y \text{ and } y \neq ?\,\}$ and $H = \{\, \langle x, y\rangle : h(x) = y\,\}$.

Clearly, $g(x) = y$ if and only if $f(i,x) = y$ for more than $e(x)/2$ of the i's. Hence, $G \in \mathrm{PP}^{\mathrm{C}_{=}\mathrm{P}}$. On the other hand, $h(x) = y$ if and only if (i) $\|\{\, i \leq e(x) : f(i,x) < y \,\}\| \leq e(x)/2$, and (ii) $\|\{\, i \leq e(x) : f(i,x) \geq y \,\}\| < e(x)/2$. Thus, $H \in \mathrm{PP}^{\mathrm{PP}}$.

Now, it follows from Lemma 3.5 that g and h are in $\mathrm{PF}^{\#\mathrm{P}[1]}$. (Note that $\mathrm{PP}^{\mathrm{PP}} = \mathrm{P}^{\#\mathrm{P}[1]}$ implies CH = $\mathrm{P}^{\#\mathrm{P}[1]}$.) Therefore, #P is closed under **maj** and **mid** in context PF $\circ$ #P. □

Following a similar proof to the above, we obtain an analogous result for context $\mathrm{PF}^{\#\mathrm{P}}$.

Corollary 3.7.
(1) #P is closed under **maj** in context $\mathrm{PF}^{\#\mathrm{P}}$ if and only if $\mathrm{P}^{\#\mathrm{P}} = \mathrm{PP}^{\mathrm{PP}}$.
(2) #P is closed under **mid** in context $\mathrm{PF}^{\#\mathrm{P}}$ if and only if $\mathrm{P}^{\#\mathrm{P}} = \mathrm{PP}^{\mathrm{PP}}$.

Theorem 3.8. #P is closed under **plu** in context PF $\circ$ #P if and only if $\mathrm{P}^{\#\mathrm{P}[1]} = \mathrm{PP}^{\mathrm{PP}}$.

Proof. Suppose that #P is closed under **plu** in context PF $\circ$ #P, and we show that every $\mathrm{PP}^{\mathrm{PP}}$ set belongs to $\mathrm{P}^{\#\mathrm{P}[1]}$. Consider any $L \in \mathrm{PP}^{\mathrm{PP}}$. Let A and p respectively be a set in $\mathrm{C}_{=}\mathrm{P}$ and a polynomial that characterize the membership of L as in Proposition 3.4 (1). Also let M and q respectively be a nondeterministic machine and a polynomial that characterize A as in Lemma 3.3.

Let N be a nondeterministic machine that operates as follows on input $x\#wb$, where $|w| = p(|x|)$ and $b \in \{0,1\}$:

(1) If $b = 0$, then N simulates M on input $x\#w$;
(2) If $b = 1$ and the last bit of w is a 0, then N nondeterministically guesses u of length $q(|x\#w|)$, and accepts on each path; and
(3) If $b = 1$ and the last bit of w is a 1, then N nondeterministically guesses u of length $q(|x\#w|)$, and rejects on each path.

Let x be any string in Σ^*, and l_x denote the length of strings of the form $x\#w$ for some $w \in \Sigma^{p(|x|)}$. By Lemma 3.3, for every w of length $p(|x|)$, $0 < acc_M(x\#w) < 2^{q(l_x)}$. Therefore, it holds that

- for exactly one forth of the v's of length $p(|x|)+1$, we have $acc_N(x\#v) = 0$,
- for exactly one fourth of the v's of length $p(|x|)+1$, we have $acc_N(x\#v) = 2^{q(l_x)}$,
- $x \in L$ if and only if for more than one fourth of the v's of length $p(|x|)+1$, we have $acc_N(x\#v) = 2^{q(l_x)-1}$.

Now consider the following functions f and g. For each $x \in \Sigma^*$ and each $i \in \mathbf{N}$, if $1 \le i \le 2^{p(|x|)+1}$, then let $f(i,x) = acc_N(x\#v)$, where v is the i-th smallest string of length $p(|x|)+1$; otherwise, let $f(i,x) = 0$. For each $x \in \Sigma^*$, define $g(x)$ by

$$g(x) = \mathrm{plu}(\, f(1,x), ..., f(2^{p(|x|)+1}, x)\,).$$

Then from the above discussion, it is easy to show that for any $x \in \Sigma^*$,

$$x \in L \iff g(x) = 2^{q(l_x)-1}.$$

On the other hand, since $f \in \#\mathrm{P}$, by our assumption, g is in $\mathrm{PF}^{\#\mathrm{P}[1]}$. Therefore, L is in $\mathrm{P}^{\#\mathrm{P}[1]}$.

Conversely, suppose that $\mathrm{PP}^{\mathrm{PP}} = \mathrm{P}^{\#\mathrm{P}[1]}$. Let $f \in \#\mathrm{P}$ and $e \in \mathrm{PF}$ and consider

$$g(x) = \mathrm{plu}(\, f(1,x), ..., f(e(x),x)\,).$$

Define $G = \{\, \langle x,y\rangle : g(x) = y \,\}$. Clearly, $g(x) = y$ if and only if (i) $\forall y\, [\, ||\{\, i \le e(x) : f(i,x) = y' \,\}|| \le ||\{\, i \le e(x) : f(i,x) = y \,\}|| \,]$, and (ii) $\forall y' < y\, [\, ||\{\, i \le e(x) : f(i,x) = y' \,\}|| < ||\{\, i \le e(x) : f(i,x) = y \,\}|| \,]$. Then it is easy to see that $G \in \text{co-}\mathrm{NP}^{\mathrm{PP}^{\mathrm{PP}}}$. Thus, it follows from Lemma 3.5 that $g \in \mathrm{PF}^{\#\mathrm{P}[1]}$, and hence, we get the desired closure property. □

By following a similar proof to the above, we obtain an analogous result for context $\mathrm{PF}^{\#\mathrm{P}}$.

Corollary 3.9. #P is closed under **plu** in context $\mathrm{PF}^{\#\mathrm{P}}$ if and only if $\mathrm{P}^{\#\mathrm{P}} = \mathrm{PP}^{\mathrm{PP}}$.

For the **max** operator, we have a somewhat different result, that indicates in turn that the **max** operator is weaker than the other operators.

Theorem 3.10. #P is closed under **max** in context PF $\circ$ #P if and only if $\mathrm{P}^{\#\mathrm{P}[1]} = \mathrm{NP}^{\mathrm{PP}}$.

Proof. Suppose that #P is closed under **max** in context PF $\circ$ #P, and show that every $\mathrm{NP}^{\mathrm{PP}}$ set belongs to $\mathrm{PF}^{\#\mathrm{P}[1]}$. Consider any $L \in \mathrm{NP}^{\mathrm{PP}}$. Let A and p respectively be a set in $\mathrm{C}_{=}\mathrm{P}$ and a polynomial that characterize L as in Proposition 3.4 (2), and let M and q respectively be a machine and a polynomial that characterize A as in Lemma 3.3.

Let x be any string in Σ^*, and let l_x be the length of strings of the form $x\#w$ for some $w \in \Sigma^{p(|x|)}$. It follows from Lemma 3.3 that $acc_M(x\#w) \le 2^{q(l_x)-1}$ for every $w \in \Sigma^{p(|x|)}$. Furthermore, if $x \in L$, then there exists some $w \in \Sigma^{p(|x|)}$ such that $acc_M(x\#w) = 2^{q(l_x)-1}$; if $x \notin L$, then for all $w \in \Sigma^{p(|x|)}$, $acc_M(x\#w) < 2^{q(l_x)-1}$.

Here we consider the following functions f and g. For each $x \in \Sigma^*$ and $i \in \mathbf{N}$, if $1 \le i \le 2^{p(|x|)}$, then $f(i,x) = acc_M(x\#w)$, where w is the i-th smallest string of length $p(|x|)$; otherwise, $f(i,x) = 0$. For each $x \in \Sigma^*$, define $g(x)$ by

$$g(x) = \max(\, f(1,x), ..., f(2^{p(|x|)}, x)\,).$$

Then it follows from the discussion above that, for every $x \in \Sigma^*$,

$$x \in L \iff g(x) = 2^{q(l_x)-1}.$$

On the other hand, $f \in \#\mathrm{P}$; thus, by our assumption, g is in PF $\circ$ #P. Therefore $L \in \mathrm{P}^{\#\mathrm{P}[1]}$.

Conversely, suppose that $\mathrm{NP}^{\mathrm{PP}} = \mathrm{P}^{\#\mathrm{P}[1]}$. Let $f \in \#\mathrm{P}$ and $e \in \mathrm{PF}$ and consider

$$g(x) = \max(\, f(1,x), ..., f(e(x),x)\,).$$

Define $G = \{\, \langle x,y\rangle : g(x) = y \,\}$. Clearly, $g(x) = y$ if and only if (i) there exists some $i \le e(x)$ such

that $f(i,x) = y$, and (ii) for all $j \leq e(x)$ we have $f(j,x) \leq y$. Thus, $G \in \mathrm{NP}^{\mathrm{NP}^{\mathrm{PP}}}$. Then it follows from Lemma 3.5 that $g \in \mathrm{PF}^{\#\mathrm{P}[1]}$. (Note that $\mathrm{NP}^{\mathrm{PP}} = \mathrm{P}^{\#\mathrm{P}[1]}$ implies $\mathrm{PH}^{\mathrm{PP}} = \mathrm{P}^{\#\mathrm{P}[1]}$.) Therefore, #P is closed under **max** in context PF $\circ$ #P. □

Again a similar result holds for context $\mathrm{PF}^{\#\mathrm{P}}$.

Corollary 3.11. #P is closed under **max** in context $\mathrm{PF}^{\#\mathrm{P}}$ if and only if $\mathrm{P}^{\#\mathrm{P}} = \mathrm{NP}^{\mathrm{PP}}$.

Acknowledgement

The material presented in this paper was initiated and studied while the authors were visiting Professor Lane Hemachandra at University of Rochester. The authors would like to thank him for leading them to this subject and his hospitality. Without him, the present paper would not have existed.

References

[BDG88] J. Balcázar, J. Díaz, and J. Gabarró, "Structural Complexity I", EATCS Monographs on Theoretical Computer Science, Springer-Verlag (1988).

[BDG91] J. Balcázar, J. Díaz, and J. Gabarró, *Structural Complexity II*, EATCS Monographs on Theoretical Computer Science, Springer-Verlag (1991).

[BHW91] R. Beigel, L. Hemachandra, and G. Wechsung, Probabilistic polynomial time is closed under parity reductions, *Inform. Proc. Lett.* 37 (2) (1991), 91–94.

[BRS91] R. Beigel, N. Reingold, and D. Spielman, PP is closed under intersection, in *Proc. 23rd ACM Sympos. on Theory of Comput.*, ACM (1991), 1–9.

[BTT92] R. Beigel, J. Tarui, and S. Toda, On probabilistic ACC circuits with an exact-threshold output gate, in *Proc. 3rd Int. Sympos. on Algorithms and Computation*, Lecture Notes in Computer Science 650 (1992), 420–429.

[CH90] J. Cai and L. Hemachandra, On the power of parity polynomial time, *Math. Syst. Theory* 23 (2) (1990), 95–106.

[Gil77] J. Gill, Computational complexity of probabilistic Turing machines, *SIAM J. Comput.* 6 (4) (1977), 675–695.

[FFK91] S. Fenner, L. Fortnow, and S. Kurtz, Gap-definable counting classes, in *Proc. 6th Structure in Complexity Theory* IEEE (1991), 30–42.

[FR91] L. Fortnow and N. Reingold, PP is closed under truth-table reductions, in *Proc. 6th Structure in Complexity Theory* IEEE (1991), 13–15.

[KST89] J. Köbler, U. Schöning, and J. Torán, On counting and approximation, *Acta Informatica* 26 (1989), 363–379.

[Sch90] U. Schöning, The power of counting, in *Complexity Theory Retrospective* (A. Selman Ed.), Springer-Verlag (1990), 204–223.

[Sim75] J. Simon, On Some Central Problems in Computational Complexity, in *PhD thesis, Cornell University*, Ithaca, N.Y., January 1975. Available as Cornell Department of Computer Science Technical Report TR75-224.

[Tod91] S. Toda, PP is as hard as the polynomial-time hierarchy, *SIAM J. Comput.* 20 (1991), 865–877.

[Tor88] J. Toran, An oracle characterization of the counting hierarchy, in *Proc. 3rd Structure in Complexity Theory* IEEE (1988), 13–15.

[OH91] M. Ogiwara and L. Hemachandra, A complexity theory for feasible closure properties, in *Proc. 6th Structure in Complexity Theory* IEEE (1991), 16–29.

[PZ83] C. Papadimitriou and S. Zachos, Two remarks on the power of counting, in *Proc. 6th GI Conference on Theoret. Comput. Sci.*, Lecture Notes in Computer Science 145 (1983), 269–276.

[Val79] L. Valiant, The complexity of enumeration and reliability problems, *SIAM J. Comput.* 8 (3) (1979), 410–421.

[Wag86a] K. Wagner, Some observations on the connection between counting and recursion, *Theoret. Comput. Sci.* 47 (1986), 131–147.

[Wag86b] K. Wagner, The complexity of combinatorial problems with succinct input representations, *Acta Informatica* 23 (1986), 325–356.