アルゴリズム 18－4
（1990．11．22）

中央値を発見することの計算複雑さ

戸田誠之助
電気通信大学　情報工学科

本稿では、Krentel[JCSS 36(1988)]によるPF^{NP}の特徴付けに類似した方法によって、$PF^{\#P}$の単純な特徴付けを示す。今、MidPをmetric Turing machine（各計算列毎に2進数を出力する多項式時間限定の非決定性Turing機械）の出力に於ける中央値を与える関数のクラスとする。このとき、$PF^{\#P}$は次のように特徴付けられる：$PF^{\#P} = PF^{MidP[1]}$。この結果を用いることによって、幾つかの自然な問題が$PF^{\#P}$完全であることが示される。

The Complexity of Finding Medians

Seinosuke TODA
Department of Computer Science and Information Mathematics
University of Electro-Communications
Chofu-shi, Tokyo 182, Japan

Abstract: The purpose of this article is to characterize $PF^{\#P}$ in a similar manner to Krentel's characterization of PF^{NP} [JCSS 36(1988), 490–509]. Let MidP be the class of functions that give the medians in the outputs of metric Turing machines (for more precise definition, see the next section). Then we show the following characterization of $PF^{\#P}$: $PF^{\#P} = PF^{MidP[1]}$. Hence, intuitively speaking, finding a median is as hard as $PF^{\#P}$; this forms a contrast to an intuitive interpretation of Kretel's result that finding a maximum (or minimum) is as hard as PF^{NP}. As its applications, we show several natural $\leq_m^P$-complete problems for D#P.

1 Definitions and notations.

Definition. A *metric Turing machine* [6] (metric TM for short) is a polynomial time bounded NTM such that every branch writes a binary number and halts. Let $\hat{N}$ be a metric TM. We denote by $Out_{\hat{N}}(x)$ the set of outputs of $\hat{N}$ on an input x. We define KthValue(x,k) to be the k-th number in $Out_{\hat{N}}(x)$. We also define $mid_{\hat{N}}(x)$ to be the median in $Out_{\hat{N}}(x)$. More precisely, for each input x to $\hat{N}$, $mid_{\hat{N}}(x)$ is the number m in $Out_{\hat{N}}(x)$ such that $0 \leq \|\{k \in Out_{\hat{N}}(x) : k \leq m\}\| - \|\{k \in Out_{\hat{N}}(x) : m < k\}\| \leq 1$. We define KthP = { K$th$Value$_{\hat{N}}$: $\hat{N}$ is a metric Turing machine } and MidP = { $mid_{\hat{N}}$: $\hat{N}$ is a metric Turing machine }.

Definition. A function f is polynomial time 1-Turing reducible to a function g ($f \leq^{PF}_{1\text{-}T} g$) if there exist polynomial time computable functions T_1 and T_2 such that for every x, $f(x) = T_1(x, g(T_2(x)))$. We denote by $\mathrm{PF}^{g[1]}$ the class of functions that are $\leq^{PF}_{1\text{-}T}$-reducible to g and we define $\mathrm{PF}^{\mathrm{G}[1]} = \cup_{g \in \mathrm{G}} \mathrm{PF}^{g[1]}$ for any class G of functions.

Notations. In the next section, we denote by M or M_i a nondeterministic Turing machine that is an acceptor, by N or N_i a deterministic transducer, and denote by $\hat{N}$ or $\hat{N}_i$ a metric TM. We denote by f, g, or h a polynomial time computable function and denote by F, G, and H a function that has higher complexity.

2 The main result.

In this section, we show that $\mathrm{PF}^{\#\mathrm{P}} = \mathrm{PF}^{\mathrm{MidP}[1]}$.

Proposition 1 [10] $\#\mathrm{P}^{\mathrm{PH}} \subseteq \mathrm{PF}^{\#\mathrm{P}[1]}$.

Main Theorem $\mathrm{PF}^{\#\mathrm{P}} = \mathrm{PF}^{\mathrm{KthP}[1]} = \mathrm{PF}^{\mathrm{MidP}[1]}$.

Proof. ($\mathrm{PF}^{\#\mathrm{P}} \subseteq \mathrm{PF}^{\mathrm{KthP}[1]} \subseteq \mathrm{PF}^{\mathrm{MidP}[1]}$) This follows from Lemma 3 and 4 below.

($\mathrm{PF}^{\mathrm{MidP}[1]} \subseteq \mathrm{PF}^{\#\mathrm{P}}$) It suffices to show that MidP $\subseteq \mathrm{PF}^{\#\mathrm{P}}$. Let $\hat{N}$ be any metric TM. We define a funciton F as follows: for every string x and natural number k, $F(x,k) = \|\{j \in Out_{\hat{N}}(x) : j \leq k\}\|$.

It is easy to see that $F \in \#\mathrm{P}^{\mathrm{NP}}$. By using a standard binary search technique, we see that $mid_{\hat{N}} \in \mathrm{PF}^{F}$. Hence, from Proposition 1, we have $mid_{\hat{N}} \in \mathrm{PF}^{\#\mathrm{P}^{\mathrm{NP}}} \subseteq \mathrm{PF}^{\mathrm{PF}^{\#\mathrm{P}[1]}} \subseteq \mathrm{PF}^{\#\mathrm{P}}$. □

Definition [2] A set A is in PP if there exists a polynomial time bounded NTM M such that for every y, if $y \in A$, then more than half of computations of M on y are accepting; otherwise, more than half of computations of M on y are rejecting.

By the following proposition, we will deal with $\mathrm{PF}^{\mathrm{PP}}$ instead of $\mathrm{PF}^{\#\mathrm{P}}$.

Proposition 2 [9] $\mathrm{PF}^{\#\mathrm{P}} = \mathrm{PF}^{\mathrm{PP}}$.

Lemma 3 Let F be any function in $\mathrm{PF}^{\#\mathrm{P}}$. Then there exists a metric Turing machine $\hat{N}$ such that $F \leq^{PF}_{1\text{-}T}$ KthValue$_{\hat{N}}$.

Proof. In this proof, we use Proposition 3 above. Let N and A be a polynomial time bounded transducer and a set in PP, respectively, that witness $F \in \mathrm{PF}^{\mathrm{PP}}$. Let M be a PP-machine accepting A.

Without loss of generality, we may assume the following conditions:

(1) M's transition function has exactly two posibilities for the next ID from each ID.

(2) There exists a polynomial p such that for every input of length m, all of M's computation paths have the same length $p(m)$.

(3) There exists a polynomial l such that for every input of length n, all queries made by N on the input have the same length $l(n)$.

(4) There exists a polynomial q such that for every input of length n, the number of queries made by N during the computation is $q(n)$.

We encode each computation of M into a binary string and identify a computation with the string encoding it. Under this setting, we define a metric TM $\hat{N}$ working on a given input x as follows:

(Phase 1) Set I to the initial ID of N on x.

(Phase 2) Execute the following steps one after another for $i = 1,2,\ldots, q(|x|)$.
(Step i)

(a) Simulate N from I until the time that a query ID occurs.
Let y be a query string made by N at this time.

(b) Guess an oracle answer a_i, either 0 ("no") or 1 ("yes"), to the query string y.
Guess a computation w_i of M on input y.

(c) According to each case below, $\hat{N}$ operates as follows:

(Case 1 : if $a = 0$ and w is rejecting **)**. Set I to the no ID corresponding to the query ID above and proceed to the next step.

(Case 2 : if $a = 1$ and w is accepting **)**. Set I to the yes ID corresponding to the query ID above and proceed to the next step.

(Otherwise) Output $2^{t(|x|)}$ and halt (for the definition of t, see below).

(Phase 3) Output $a_1w_1a_2w_2a_3w_3\ldots a_{q(|x|)}w_{q(|x|)}$ and halt.

In the above, we define t to be a polynomial such that for every input x, $2^{t(|x|)}$ is greater than all outputs in Phase 3. Obviosely, such a polynomial exists. Next, we define k_x by

$$k_x = 1 + \sum_{i=0}^{q(|x|)-1} 2^{p(l(|x|))-1} \cdot (2^{p(l(|x|))})^i.$$

As a usual manner, it is common to specify a computation of N^A on input x by a sequence of oracle answers to queries made by N^A on x. Furthermore, it is obvious that once we have the sequence, we can compute the output of N^A on x in polynomial time in $|x|$. The purpose below is to show that if $a_1w_1a_2w_2\ldots a_{q(|x|)}w_{q(|x|)}$ is the k_x-th number in $Out_{\hat{N}}(x)$, then $a_1a_2\ldots a_{q(|x|)}$ is the sequence of correct oracle answers of the oracle set A to the queries made by N^A on input x. If we can accomplish this, then we have a $\leq^{PF}_{1\text{-}T}$-reduction from F to $Kth\text{value}_{\hat{N}}$ because k_x is computable in polynomial time in $|x|$ and

we can compute $a_1a_2\ldots a_{q(|x|)}$ from $a_1w_1\ldots a_{q(|x|)}w_{q(|x|)}$ in polynomial time in $|x|$ (recall that the length of each w_i is $p(l(|x|))$).

In order to accomplish the above purpose, we define some terminology and state some conventions. We say that a computation of $\hat{N}$ on an input x is *valid* if the computation halts in Phase 3. In the above definition of $\hat{N}$, nondeterministic moves appear only on each Step i of Phase 2. Hence, we can specify a valid computation of $\hat{N}$ on an input x by a binary string, say $a_1w_1a_2w_2\ldots a_{q(|x|)}w_{q(|x|)}$, guessed in each Step i(b). Below, we identify the sequence with its specifying valid computation. Furthermore, if a binary string aw guessed in some Step i(b) is followed by a valid computation of $\hat{N}$, then we call it a *partial valid computation*. We also regard all partial valid computations as binary numbers.

Let x be any input to $\hat{N}$. From the assumptions (1), (2), and (3), we can easily see that the number of all valid computations of $\hat{N}$ on x is exactly $2^{p(l(|x|))q(|x|)}$ and this value is greater than k_x defined above. Hence, $Kth\text{Value}_{\hat{N}}(x,k_x)$ is an output in Phase 3. Let $Kth\text{Value}_{\hat{N}}(x,k_x) = a_1w_1a_2w_2\ldots a_{q(|x|)}w_{q(|x|)}$, where each a_i is an oracle answer guessed in Step i(b) and w_i is a computation of M on the query string concerned in that step. First, we show that a_1 is the correct anwser of the oracle set A to the first query string made by N^A on input x. The following is the key claim to show this.

Claim 1 Let $n_{1,pacc}$ denote the number of all partial valid computations in Step 1 (note that $n_{1,pacc}$ is even from the assumptions (1), (2), and (3)). Then a_1w_1 above is the $(n_{1,pacc}/2+1)$-th partial valid computation in all partial valid computations in Step 1.

Before proving this claim, we show, by using Claim 1, that a_1 is the correct answer of the oracle set A to the first query made by N^A on input x. Let y be the first query string. From the definitin of $\hat{N}$, it is obvious that the number of accepting (rejecting) computations of M on y is equal to the number of partial valid computations of $\hat{N}$ on x in Step 1 that are of the form $1w$ (resp., $0w$). Furthermore, from the assumptions (1), (2), and (3), the number of all computations of M on y is equal to $2^{p(l(|x|))}$ and it is also the number of partial valid computations of $\hat{N}$ on x in Step 1 (i.e. $n_{1,pacc}$ in Claim 1).

Now assume $y \in A$. From the definition of PP, more than half of all compuations of M on y are accepting. Hence, we know that the number of partial valid computations of $\hat{N}$ on x in Step 1 that is of the form $0w$ is less than half of all partial valid computations of $\hat{N}$ on x in Step 1 because the string w should be a rejecting computation of M on y. Hence, from Claim 1, a_1w_1 must be of the form $1w$; that is, a_1 represent the oracle answer "yes". Conversely, assume $y \notin A$. Then, more than half of computations of M on y are rejecting. Thus, the number of partial valid computations of $\hat{N}$ on x in Step 1 that are of the form $0w$ is greater than half of all partial valid computations in Step 1. Hence, from Claim 1, a_1w_1 must has the form $0w$; that is, a_1 represent the oracle answer "no".

Now let us prove Claim 1. First, for any partial valid computation aw in Step 1, we estimate the number of valid computations of $\hat{N}$ on input x of which aw is a prefix. Let y be a query string concerned in Step 2 after $\hat{N}$ on input x has guessed aw in Step 1. Then, from the definition of $\hat{N}$, we see that a binary string $a'w'$ is a partial valid computation of $\hat{N}$ in Step 2 if and only if $a'=0$ and w' is a rejecting computation of M on y, or, $a'=1$ and w' is an accepting computation of M on y. Thus, as in the case of Step 1, the number of partial valid computations of $\hat{N}$ in Step 2 is equal to the number of all computations of M on y, which is equal to $2^{p(l(|x|))}$ from the assumptions (1), (2), and (3). By a similar argument, we

can observe that in every Step i, the number of partial valid computations of $\hat{N}$ on x is equal to $2^{p(l(|x|))}$. From this observation and the assumption (4), we have the following fact.

Fact 1. The number of valid computations of $\hat{N}$ on x of which aw is a prefix is equal to $(2^{p(l(|x|))})^{q(|x|)-1}$, where aw is any partial valid computation of $\hat{N}$ on x in Step 1.

For any partial valid computation aw in Step 1, let $aw00\ldots0$ ($aw11\ldots1$) denote the least (resp., greatest) binary number possibly written by $\hat{N}$ on input x of which aw is a prefix. We also define $PACC^1_{\hat{N}}(x)$ to be the set of all partial valid computation of $\hat{N}$ on x in Step 1.

Recall that $Kth\text{Value}_{\hat{N}}(x,k_x) = a_1w_1\ldots a_{q(|x|)}w_{q(|x|)}$. Then, a_1w_1 has to satisfy the following condition:

$$\|\{u \in Out_{\hat{N}}(x) : u < a_1w_100\ldots0\}\| < k_x \leq \|\{u \in Out_{\hat{N}}(x) : u \leq a_1w_111\ldots1\}\|.$$

From Fact 1, we have the following:

$$\|\{u \in Out_{\hat{N}}(x) : u < a_1w_100\ldots0\}\| = \|\{aw \in PACC^1_{\hat{N}}(x) : aw < a_1w_1\}\| \cdot (2^{p(l(|x|))})^{q(|x|)-1}, \text{ and}$$
$$\|\{u \in Out_{\hat{N}}(x) : u \leq a_1w_111\ldots1\}\| = \|\{aw \in PACC^1_{\hat{N}}(x) : aw \leq a_1w_1\}\| \cdot (2^{p(l(|x|))})^{q(|x|)-1}.$$

Let $m = \|\{aw \in PACC^1_{\hat{N}}(x) : aw \leq a_1w_1\}\|$. Then, from the above observations, we have the following inequality:

$$(m-1)\cdot(2^{p(l(|x|))})^{q(|x|)-1} < k_x \leq m\cdot(2^{p(l(|x|))})^{q(|x|)-1}.$$

From this, m has to satisfy

$$k_x/(2^{p(l(|x|))})^{q(|x|)-1} \leq m < k_x/(2^{p(l(|x|))})^{q(|x|)-1} + 1.$$

From the definition of k_x, there exists a real number α, $0 < \alpha < 1$, such that $k_x/(2^{p(l(|x|))})^{q(|x|)-1} = 2^{p(l(|x|))-1} + \alpha$. Hence, m must satisfy the following condition:

$$2^{p(l(|x|))-1} + \alpha \leq m < 2^{p(l(|x|))-1} + \alpha + 1.$$

Hence $m = 2^{p(l(|x|))-1} + 1$. We again notice that the number of partial valid computations of $\hat{N}$ on x in Step 1 (i.e., $n_{1,pacc}$ in Claim 1) is equal to the number of all computations of M on y, the first query string made by N^A on x, which is also equal to $2^{p(l(|x|))}$ from the assumptions (1), (2), and (3). Hence we have $m = n_{1,pacc}/2 + 1$. This completes the proof of Claim 1.

By the same argument, we see that a_2 is the correct oracle answer of the oracle set A to the second query made by N^A on input x, given that a_1 is correct; and hence, by induction, all of a_i's are the correct oracle answers that appear in the computation of N^A on input x.

As mentioned previously, once we have the correct oracle answers in the computation of N^A on input x, we can compute the output of N^A on x in polynomial time in $|x|$. This provides us with a $\leq^{PF}_{1\text{-}T}$-reduction from F to $Kth\text{Value}_{\hat{N}}$. □

Lemma 4 For any metric TM $\hat{N}$, there exists a metric TM $\hat{N}_1$ such that $Kth\text{Value}_{\hat{N}} \leq^{PF}_{1\text{-}T} mid_{\hat{N}_1}$.

Proof. Let t be a polynomial that is a strict upper bound on the length of outputs of $\hat{N}$. We defined a metric TM $\hat{N}_1$ working on input $x\#k$ as follows.

(Phase 1) $\hat{N}_1$ nondeterministicaly chooses one of Phases 2, 3, and 4 below

(Phase 2) $\hat{N}_1$ nondeterministicaly chooses one of (a) and (b) below.

(a) It simulates $\hat{N}$ on input x.

(b) It guesses a binary number j such that $0 \leq j < 2^{t(|x|)}$.

If $j < 2^{t(|x|)} - k - 1$, then it outputs $2^{t(|x|)} + j$; otherwise, it outputs $2^{t(|x|)+1}$.

(Phase 3) $\hat{N}_1$ simulates $\hat{N}$ on input x. When $\hat{N}$ outputs j, $\hat{N}_1$ outputs $2^{t(|x|)+2} + j$.

(Phase 4) $\hat{N}$ guesses a binary number j such that $0 \leq j < 2^{t(|x|)+1}$.

If $j < 2^{t(|x|)} + k - 1$, then it outputs $2^{t(|x|)+3} + j$; otherwise, it outputs $2^{t(|x|)+4}$.

For each $i = 2,3,4$, we denote by $Out^i_{\hat{N}_1}(x)$ the set of binary numbers written in the Phase i of $\hat{N}_1$ on input x. Then it is easy to see that $\|Out^2_{\hat{N}_1}(x)\| = 2^{t(|x|)} - k + \|Out_{\hat{N}}(x)\|$, $\|Out^3_{\hat{N}_1}(x)\| = \|Out_{\hat{N}}(x)\|$, and $\|Out^4_{\hat{N}_1}(x)\| = 2^{t(|x|)} + k$. Furthermore, it is easy to see that for every $i \in Out^2_{\hat{N}_1}(x)$, $j \in Out^3_{\hat{N}_1}(x)$, and $k \in Out^4_{\hat{N}_1}(x)$, $i < j < k$. Denoting by m the k-th number in $Out_{\hat{N}}(x)$, it follows from these observations that $m + 2^{t(|x|)+2}$ is the median in $Out_{\hat{N}_1}(x)$. This implies $Kth\text{Value}_{\hat{N}} \leq^{PF}_{1\text{-}T} mid_{\hat{N}_1}$. □

3 Complete functions for $PF^{\#P}$ and complete problems for D#P.

In this section, we show some natural complete functions for $PF^{\#P}$ under $\leq^{PF}_{1\text{-}T}$-reducibility.

Lemma 5 The following functions are $\leq^{PF}_{1\text{-}T}$-complete for $PF^{\#P}$.

- **LEXICAL K-th SAT**

  Input A boolean formula $\varphi(x_1, x_2, \ldots, x_n)$ and a natural number k.

  Output The lexicographically k-th satisfying assignment $x_1x_2\cdots x_n \in \{0,1\}^n$ of φ.

- **LEXICAL MIDDLE SAT**

  Input A boolean formula $\varphi(x_1, x_2, \ldots, x_n)$.

  Output The lexicographically middle satisfying assignment $x_1x_2\cdots x_n \in \{0,1\}^n$ of φ. □

Theorem 6 The following functions are $\leq^{PF}_{1\text{-}T}$-complete for $PF^{\#P}$.

- **LEXICAL K-th 3SAT**

  Input A boolean formula $\varphi(x_1, x_2, \ldots, x_n)$ in 3CNF and a positive integer k.

  Output The lexicographically k-th satisfying assignment $x_1x_2\cdots x_n \in \{0,1\}^n$ of φ.

- **LEXICAL K-th HAMILTONIAN CIRCUIT**

  Input An undirected graph $G = (V, E)$ and a positive integer k.

  Output The lexicographically k-th Hamiltonian circuit of G, where we assume that a linear ordering on E is given, and for any subsets E_1, E_2 of E, E_1 *is greater than* E_2 if the smallest edge in $E_1 - E_2$ is greater than the smallest edge in $E_2 - E_1$.

- **K-th SHORTEST PATH**

  Input An undirected graph G with edge weights and a positive integer k.

  Output The k-th shortest path in the graph.

- **K-th LARGEST SUBSET**

Input A finite set S of positive integers, and positive integers B and k.

Output The sum of integers of the k-th largest subset X of S such that $\sum_{i \in X} i \leq B$, where all subsets of S are ordered by the sum of all elements in the subsets. □

Theorem 7 The following functions are $\leq^{PF}_{1\text{-}T}$-complete for $\mathrm{PF}^{\#\mathrm{P}}$.

- **LEXICAL MIDDLE 3SAT**

  Input A boolean formula $\varphi(x_1, x_2, \ldots, x_n)$ in 3CNF.

  Output The lexicographically middle satisfying assignment $x_1 x_2 \cdots x_n \in \{0,1\}^n$ of φ.

- **LEXICAL MIDDLE HAMILTONIAN CIRCUIT**

  Input An undirected graph $G = (V, E)$.

  Output The lexicographically middle Hamiltonian circuit of G.

- **MIDDLE SHORTEST SIMPLE PATH**

  Input An undirected graph G with edge weights.

  Output The median in all the weights of simple paths in the graph.

- **MIDDLE SUBSET**

  Input A finite set S of positive integers, and positive integers B and C.

  Output The median in all the sums of integers of subsets X of S such that $B \leq \sum_{i \in X} i \leq C$. □

We also show that some decision problems corresponding to the above functions are $\leq^{P}_{m}$-complete for D#P.

Theorem 8 The following decision problems are $\leq^{P}_{m}$-complete for D#P.

- **LEXICAL MIDDLE 3SAT**

  Input A boolean formula $\varphi(x_1, x_2, \ldots, x_n)$ in 3CNF.

  Question Is x_n=1 in the lex. middle satisfying assignment $x_1 x_2 \cdots x_n \in \{0,1\}^n$ of φ?

- **LEXICAL MIDDLE HAMILTONIAN CIRCUIT**

  Input An undirected graph $G = (V, E)$ and an edge e.

  Question Does the lexicographically middle Hamiltonian circuit of G contains e?

- **LEXICAL MIDDLE SIMPLE PATH**

  Input An undirected graph G and an edge e.

  Question Does the lexcographically middle simple path of G contains e?

- **MIDDLE SIMPLE PATH**

  Input An undirected graph G with edge weight.

  Question Is the median in all the weights of simple paths in G even?

- **MIDDLE LARGEST SUBSET**

  Input A finite set S of positive integers, and positive integers B and C.

  Output Is the median in all the sums of integers of subsets X of S such that $B \leq \sum_{i \in X} i \leq C$. □

Acknowledgement The author would like to be very thankful to Prof. Ronald V. Book for his kind support and his valuable comments on this work.

References

[1] S. A. Cook, The complexity of theorem proving procedures, *Proc. 3rd ACM Symposium on Theory of Computing*(1971), 151–158.

[2] J. Gill, Computational Complexity of Probabilistic Turing Machines, *SIAM Journal on Computing* 6(1977), 675–695.

[3] M. Garey and D. B. Johnson, Computers and Intractability: A Guide to the Theory of NP-Completeness, Freeman, San Francisco, 1979.

[4] D. B. Johnson and S. D. Kashdan, Lower Bounds for Selecting in $X+Y$ and Other Multisets, *Journal of ACM*, 25(1978), 556–570.

[5] D. B. Johnson and T. Mizoguchi, Selecting the Kth Element in $X+Y$ and $X_1+X_2+\ldots+X_m$, *SIAM Journal on Computing*, 7(1978), 147–153.

[6] M. W. Krentel, The Complexity of Optimization Problems, *Journal of Computer and System Science*, 36(1988), 490–509.

[7] S. Miyano, Δ_2^{P}-complete lexicographically first maximal subgraph problems, *Proc. Mathematical Foundations of Computer Science*, Lecture Notes in Computer Science **324** (1988), Springer-Verlag, 454–462.

[8] C. H. Papadimitriou, On the Complexity of Unique Solutions, *Journal of ACM*, 31(1984), 392–400.

[9] J. Simon, On the Difference between One and Many, *Proc. 4th Colloq. on Automata, Languages, and Programming*, Lecture Notes in Computer Science, 52(1977), Springer, Berlin.

[10] S. Toda and O. Watanabe, Polynomial Time 1-Turing Reductions from #PH to #P, *Theoretical Compuer. Science*, to appear.

[11] L. G. Valiant, The Complexity of Computing Permanent, *Theoretical Computer Science*, 8(1979), 189–201.

[12] L. G. Valiant, The Complexity of Enumeration and Reliability Problems, *SIAM Journal on Computing*, 8(1979), 410–421.