Title: Complexity Classes Characterized by Semi-Random Sources

Authors: Ryuhei Uehara

Affiliations: Center for Information Science, Tokyo Woman's Christian University, Suginami-ku, Tokyo 167, Japan.

Corresponding author: Ryuhei Uehara, Center for Information Science, Tokyo Woman's Christian University, 2-6-1 Zempukuji, Suginami-ku, Tokyo 167, Japan. Phone: +81-3-5382-6403. Fax: +81-3-3395-1377. E-mail: uehara@twcu.ac.jp

Key words: Computational complexity, Semi-random sources, Interactive proof systems.

Abstract: The complexity classes PP, BPP and RP are usually defined via probabilistic Turing machines (PTMs) that have access to a perfect random source. A natural question is this: Do these classes change if the PTMs only have access to an arbitrary semi-random source rather than a perfect random source? The notion of semi-randomness are defined by Satha and Vazirani in 1984, and this question was first considered by Vazirani and Vazirani in 1985, who proved that RP does not change even if the PTMs only have access to an *arbitrary* semi-random source. Later, Vazirani proved that BPP does not change, either. In this paper, we show a surprising result that PP collapses to BPP if the PTMs only have access to an *arbitrary* semi-random source. We also consider the following question: Do the classes PP, BPP and RP change if the PTMs only have access to a *specific* semi-random source? As answers to this question, we show that RP changes to NP while both BPP and PP change to PSPACE.

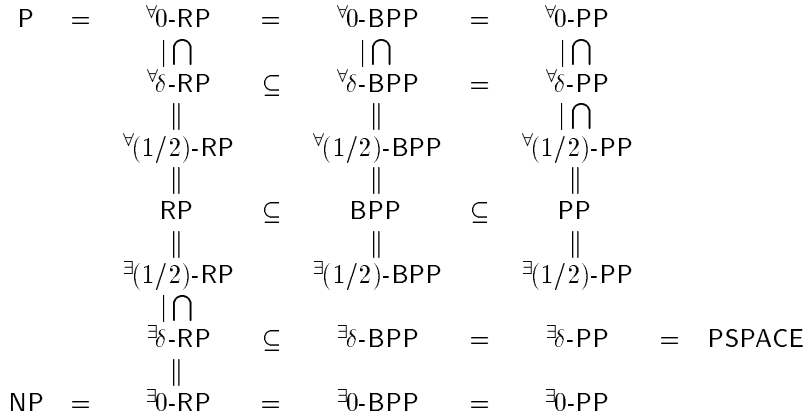

Figure 1: Summary of the results ($0 < \delta < \frac{1}{2}$)

1 Introduction

Satha and Vazirani introduced, as an extremely general model of an imperfect source of randomness, a "slightly random source" in [SV84], or "semi-random source" in [SV86]. In this paper, we refer to the random source as a "*semi-random source*." In a semi-random source, the previous bits generated by the source can condition the next bit in an arbitrarily way. Accordingly, the next bit is generated by the flip of a coin whose bias is fixed by an adversary who has complete knowledge of the history of the process. The adversary is limited to choosing a bias in $[\delta, 1-\delta]$ with some positive number $0 < \delta < \frac{1}{2}$. We call a semi-random source with parameter δ a *δ-random source*.

Gill defined the classes, PP, BPP and RP via PTMs that have access to a perfect random source [Gil77]. What if the PTMs only have access to a semi-random source (rather than a perfect random source)? This question was first considered by Vazirani and Vazirani [VV85], who defined the class SR_p as follows: A language L is in SR_p if there exists a PTM M such that (1) for $x \in L$, M accepts with probability greater than $\frac{1}{2}$ for all δ-random sources and (2) for $x \notin L$, M always rejects. Their result is that $SR_p = \mathsf{RP}$. Later, Vazirani similarly defined the class $SBPP$ (corresponding to BPP) and prove that $SBPP = \mathsf{BPP}$. In the same manner, we introduce the class $^{\forall}\delta$-PP (corresponding to PP) as follows: A language L is in $^{\forall}\delta$-PP if there exists a PTM M such that (1) for $x \in L$, M accepts with probability greater than $\frac{1}{2}$ for all δ-random sources and (2) for $x \notin L$, M accepts with probability less than $\frac{1}{2}$ for all δ-random sources. We prove that $^{\forall}\delta\text{-PP} = \mathsf{BPP}$ for $0 < \delta < \frac{1}{2}$. Compared with that $SR_p = \mathsf{RP}$ and $SBPP = \mathsf{BPP}$, this result is very interesting.

Recall that in the definition of $^{\forall}\delta$-PP, the PTM M is required to work for *every* δ-random source. This is what the symbol "$\forall$" means. This also motivated us to introduce the class $^{\exists}\delta$-PP. More precisely, we define the class $^{\exists}\delta$-PP as follows: A language L is in $^{\exists}\delta$-PP if there exists a PTM M such that (1) for $x \in L$, M accepts with probability greater than $\frac{1}{2}$ for *at least one* δ-random source and (2) for $x \notin L$, M accepts with probability less than $\frac{1}{2}$ for *all* δ-random sources. The classes $^{\exists}\delta$-BPP and $^{\exists}\delta$-RP are defined similarly. We prove that $^{\exists}\delta\text{-RP} = \mathsf{NP}$, and $^{\exists}\delta\text{-BPP} = {}^{\exists}\delta\text{-PP} = \mathsf{PSPACE}$ for $0 < \delta < \frac{1}{2}$. These results give new characterizations of NP and PSPACE. We summarize the relations between the classes in Figure 1. To unify the notations, we use $^{\forall}\delta$-RP to denote SR_p and $^{\forall}\delta$-BPP to denote $SBPP$.

Shamir showed that interactive proof systems recognize every language in PSPACE [Sha90]. Condon and Ladner investigated interactive proof systems with some restrictions on provers, and showed that some of such systems characterize the classes MA and NP [CL95]. The PTM M which witnesses $L(M) \in {}^{\exists}\delta$-RP can be seen as an interactive proof system with a restriction on the condition of the probability that system accepts. This observation implies that the interactive proof system which accepts with probability either exceeds $\frac{1}{2}$ or 0, characterizes the class NP.

2 Preliminaries

We assume a standard Turing machine model. For formal definitions of a *deterministic Turing machine* (DTM) and a *nondeterministic Turing machine* (NTM), see [HU79]. A *probabilistic Turing machine* (PTM) is a Turing machine with distinguished states called coin-tossing states. For formal definitions of a PTM, see [Gil77, BDG88]. Without loss of generality, we assume that an NTM (or PTM) is standardized as follows: It is a polynomial time bounded NTM (or PTM) with exactly two choices (called *0-choice* and *1-choice*, respectively) per step. Therefore, its computation tree is a full binary tree whose depth is polynomial in the length of the input.

A semi-random source with parameter δ, or simply a δ-random source, is defined as follows.

Definition 1 [SV84] *Let δ be a number such that $0 \leq \delta \leq \frac{1}{2}$. A* semi-random source *with parameter δ outputs bits $X_1X_2\cdots$, such that for all i and for all $x_1, x_2, \cdots$,*

$$\delta \leq \Pr[X_i = x_i \mid X_1 = x_1, \cdots, X_{i-1} = x_{i-1}] \leq 1 - \delta.$$

Definition 2 [VV85] *A language L is in ${}^{\forall}\delta$-RP if there exists a PTM M such that (1) for $x \in L$, M accepts with probability greater than $\frac{1}{2}$ and (2) for $x \notin L$, M always rejects.*

We note that the class ${}^{\forall}\delta$-RP is denoted by SR_p in [VV85].

Definition 3 [Vaz86] *A language L is in ${}^{\forall}\delta$-BPP if there exists a PTM M such that (1) for $x \in L$, M accepts with probability greater than $\frac{3}{4}$ and (2) for $x \notin L$, M accepts with probability less than $\frac{1}{4}$ for every δ-random source.*

We remind that the class ${}^{\forall}\delta$-BPP is denoted by $SBPP$ in [Vaz86].

Definition 4 *A language L is in ${}^{\forall}\delta$-PP if there exists a PTM M such that (1) for $x \in L$, M accepts with probability greater than $\frac{1}{2}$ and (2) for $x \notin L$, M accepts with probability less than $\frac{1}{2}$ for every δ-random source.*

We define the class ${}^{\exists}\delta$-RP as follows.

Definition 5 *A language L is in ${}^{\exists}\delta$-RP if there exists a PTM M such that (1) for $x \in L$, M accepts with probability greater than $\frac{1}{2}$ for* at least one *δ-random sources and (2) for $x \notin L$, M always rejects.*

Similarly, we define the classes ${}^{\exists}\delta$-BPP and ${}^{\exists}\delta$-PP as follows.

Definition 6 *A language L is in ${}^{\exists}\delta$-BPP if there exists a PTM M such that (1) for $x \in L$, M accepts with probability greater than $\frac{3}{4}$ for* at least one *δ-random source and (2) for $x \notin L$, M accepts with probability less than $\frac{1}{4}$ for* every *δ-random source.*

Definition 7 *A language L is in ${}^{\exists}\delta$-PP if there exists a PTM M such that (1) for $x \in L$, M accepts with probability greater than $\frac{1}{2}$ for* at least one *δ-random source and (2) for $x \notin L$, M accepts with probability less than $\frac{1}{2}$ for* every *δ-random source.*

A $\frac{1}{2}$-random source is a perfect random source. Thus, it is easy to see that RP $= {}^{\forall}(1/2)$-RP $= {}^{\exists}(1/2)$-RP, BPP $= {}^{\forall}(1/2)$-BPP $= {}^{\exists}(1/2)$-BPP, and PP $= {}^{\forall}(1/2)$-PP $= {}^{\exists}(1/2)$-PP.

The following proposition follows from the definitions immediately:

Proposition 1 *For every δ with $0 < \delta < \frac{1}{2}$, the following hold.*

$$\begin{aligned}
\mathsf{P} &= {}^{\forall}0\text{-RP} \subseteq {}^{\forall}\delta\text{-RP} \subseteq {}^{\forall}(1/2)\text{-RP} = \mathsf{RP} \\
&= {}^{\exists}(1/2)\text{-RP} \subseteq {}^{\exists}\delta\text{-RP} \subseteq {}^{\exists}0\text{-RP} = \mathsf{NP}, \\
\mathsf{P} &= {}^{\forall}0\text{-BPP} \subseteq {}^{\forall}\delta\text{-BPP} \subseteq {}^{\forall}(1/2)\text{-BPP} \\
&= \mathsf{BPP} = {}^{\exists}(1/2)\text{-BPP}, {}^{\exists}0\text{-BPP} = \mathsf{NP}, \\
\mathsf{P} &= {}^{\forall}0\text{-PP} \subseteq {}^{\forall}\delta\text{-PP} \subseteq {}^{\forall}(1/2)\text{-PP} \\
&= \mathsf{PP} = {}^{\exists}(1/2)\text{-PP}, \textit{ and } {}^{\exists}0\text{-PP} = \mathsf{NP}.
\end{aligned}$$

Proof. For δ and δ' with $0 \leq \delta \leq \delta' \leq \frac{1}{2}$, a δ'-random source is also a δ-random source. A 0-random source generates 1 with probability between 0 and 1. Thus for $^{\forall}0$-RP, $^{\forall}0$-BPP and $^{\forall}0$-PP, since every computation for each input must produce same answer, the algorithm must be deterministic. This implies $^{\forall}0\text{-RP} = {}^{\forall}0\text{-BPP} = {}^{\forall}0\text{-PP} = \mathsf{P}$. Conversely, for $^{\exists}0$-RP, $^{\exists}0$-BPP and $^{\exists}0$-PP, since it is sufficient to accept that at least one computation reaches accepting state, the algorithm is nondeterministic. This implies $^{\exists}0\text{-RP} = {}^{\exists}0\text{-BPP} = {}^{\exists}0\text{-PP} = \mathsf{NP}$. ∎

Note that for $\delta < \delta'$, $^{\exists}\delta'\text{-BPP} \subseteq {}^{\exists}\delta\text{-BPP}$ and $^{\exists}\delta'\text{-PP} \subseteq {}^{\exists}\delta\text{-PP}$ do not hold, whereas $^{\exists}\delta'\text{-RP} \subseteq {}^{\exists}\delta\text{-RP}$ holds. To see this, let M be a PTM such that $L(M) \in {}^{\exists}\delta'\text{-BPP}$ and δ and δ' be two numbers with $0 \leq \delta < \delta' \leq \frac{1}{2}$. It likely holds that $L(M) \in {}^{\exists}\delta\text{-BPP}$ since the input x with $x \in L(M)$ satisfies the condition of the definition of $^{\exists}\delta$-BPP since a δ'-random source is also a δ-random source. However, for $x \notin L$, M must accepts with probability less than $\frac{1}{4}$ for *every* δ-random sources. Since the set of δ-random sources properly contain the set of δ'-random sources when $\delta < \delta'$, M may not satisfy the condition.

For $0 < \delta \leq \frac{1}{2}$, the following results are known.

Theorem 2 [VV85] $^{\forall}\delta\text{-RP} = \mathsf{RP}$.

Theorem 3 [Vaz86] $^{\forall}\delta\text{-BPP} = \mathsf{BPP}$.

Since the proof of Theorem 3 plays an important role in this paper, we show the outline of the proof.

Proof of Theorem 3 [Pap94]. Let L be a language with $L \in \mathsf{BPP}$, and M_0 be a PTM such that $L(M_0) = L$. Let $p(n)$ be the length of a computation path of M_0 on input of length n. Without loss of generality, we can assume that the number of the accepting path is, by repeating the algorithm enough times, at least $\frac{31}{32}2^{p(n)}$ for $x \in L$, and at most $\frac{1}{32}2^{p(n)}$ for $x \notin L$. Let $r(n) = \lceil \frac{3\log p(n)+6}{2\delta - 2\delta^2} \rceil$. A sequence of $r(n)$ bits will be called *block*. The $2^{r(n)}$ possible blocks are denoted by the corresponding binary integers $0, 1, \cdots, 2^{r(n)} - 1$. If $\kappa = (\kappa_1, \kappa_2, \cdots, \kappa_{r(n)})$ and $\lambda = (\lambda_1, \lambda_2, \cdots, \lambda_{r(n)})$ are blocks, then their *inner product* is defined $\kappa \cdot \lambda = \sum_{i=1}^{r(n)} \kappa_i \lambda_i \pmod 2$. Notice that the inner product of two blocks is a bit. Now we construct a PTM M_0' simulating M_0. M_0' simulates $2^{r(n)}$ M_0s in parallel. Without loss of generality, we can assume that every computation of M_0 has $p(n)$ choices. The jth choice of ith simulation of M_0 is performed as follows;

> (Simulation of a probabilistic choice:)
> generate $r(n)$ bits by a δ-random source in β_j;
> $h_{(i,j)} = \beta_j \cdot i$;
> select $h_{(i,j)}$-choice;

Notice that β_j depends only j. In other words, β_j is used $2^{r(n)}$ times of jth choices on the $2^{r(n)}$ simulations. At the end of the simulation, M_0' accepts if at least half of $2^{r(n)}$ simulations accepts, or rejects otherwise. Let $T = \{(\beta_0 \cdot \kappa, \beta_1 \cdot \kappa, \cdots, \beta_{p(n)-1} \cdot \kappa) \mid \kappa = 0, 1, \cdots, 2^{r(n)} - 1\}$, and $B \subset \{0,1\}^{p(n)}$ be an arbitrary set with $|B| \leq \frac{1}{32}2^{p(n)}$. It was shown in [Pap94, the proof of Theorem 11.4] that $\Pr[\ |T \cap B| \geq \frac{1}{2}\ |T|\ ] < \frac{1}{8}$. This imply that M_0' accepts with probability greater than $\frac{3}{4}$ for $x \in L$, and it accepts with probability less than $\frac{1}{4}$ for $x \notin L$. Thus $L \in \mathsf{BPP}$. ∎

Notice that M_0' works for every δ-random source with $0 < \delta < \frac{1}{2}$. Intuitively, Theorem 3 shows that the PTM can simulate polynomial number of probabilistic choices using polynomial number of bits generated by an arbitrary δ-random source with high probability.

The following lemma due to Luts can be proved using Chernoff Bounds [Che52].

Lemma 4 [Lut90] *Let $h(x, y)$ be a weighted entropy defined by $-x\log y - (1-x)\log(1-y)$. Then,*
$\sum_{i=0}^{bt} \binom{t}{i} a^i (1-a)^{t-i} \leq 2^{-ct}$ *for* $0 < b < a < 1$, *and*
$\sum_{i=bt}^{t} \binom{t}{i} a^i (1-a)^{t-i} \leq 2^{-ct}$ *for* $0 < a < b < 1$,
where $c = h(b, a) - h(b, b)$.

3 Results for $^{\forall}\delta$-PP

In this section, we prove the following theorem.

Theorem 5 $^{\forall}\delta\text{-PP} = \mathsf{BPP}$ *for* $0 < \delta < \frac{1}{2}$.

The probability that a PTM with a δ-random source accepts, depends not only on the number of the accepting paths, but also on the distribution of the accepting paths in the computation tree of the PTM. To consider the distribution, we define some notations. A *δ-tree* is a rooted full binary tree whose every leaf is labeled "accept" or "reject", and every edge is assigned a probability between δ and $1-\delta$. A path to a leaf with label "accept" ("reject") is called an accepting (resp. rejecting) path. For a δ-tree T, we denote by $|T|$ the number of accepting paths of T.

Definition 8 *For each δ with $0 \leq \delta \leq \frac{1}{2}$, a* δ-assignment F *to a δ-tree is a mapping from the set of edges of the δ-tree to the interval $[\delta, 1-\delta]$ such that the two edges leaving each internal node are assigned values whose sum is one.*

The probability that a PTM accepts is completely determined by the PTM and a δ-assignment.

Definition 9 *Let T be a δ-tree, and F be a δ-assignment to T. The* probability of a node *of T under F is the product of the values assigned to the edges on the path from root to the node by F. The* probability of T *under F, denoted by* $\Pr[T \mid F]$*, is the sum of the probabilities of the leaves labeled "accept".*

For a given δ-tree T and an assignment F to T, F is *a maximum assignment* if $\Pr[T \mid F] \geq \Pr[T \mid F']$ holds for every δ-assignment F' to T.

Proposition 6 *For every δ-tree T, there exists a maximum assignment F to T such that F only assigns δ or $1-\delta$ to each edge of T.*

Proof. For a given δ-tree T, we recursively define an assignment F as follows:

First, we consider the nodes whose sons are leaves. Let v be such a node. If v has at least one son labeled "accept", then assign $(1-\delta)$ to the edge between v and the son, and assign δ to the other edge incident to v. Otherwise, assign $(1-\delta)$ and δ to the two edges incident to v, respectively.

Secondly, we consider the nodes whose sons are the root of the subtrees already assigned. Let v be such a node, and T_0 and T_1 be such subtrees. If $\Pr[T_0 \mid F] > \Pr[T_1 \mid F]$, then assign $(1-\delta)$ to the edge between v and the root of T_0 and assign δ to the edge between v and the root of T_1. Otherwise, assign $(1-\delta)$ to the edge between v and the root of T_1 and assign δ to the edge between v and the root of T_0.

A simple induction on the depth of the tree shows that $\Pr[T \mid F] \geq \Pr[T \mid F']$ for every δ-assignment F' to T. ∎

This proposition shows that to maximize the probability, it is sufficient to consider δ-assignments which only assign the values δ or $1-\delta$. For a given δ-tree T, we denote by $F_{\max}(T)$ the maximum assignment defined in the proof of the proposition.

Definition 10 *A δ-tree T is the* worst *if* $\Pr[T \mid F_{\max}(T)] \leq \Pr[T' \mid F_{\max}(T')]$ *for every δ-tree T' with $|T| = |T'|$.*

Proposition 7 *Let T be a worst tree. Then, for every δ-tree T' such that* $\Pr[T \mid F_{\max}(T)] = \Pr[T' \mid F_{\max}(T')]$*, $|T| \geq |T'|$ holds.*

Proof. There exists a worst tree T'' such that $\Pr[T'' \mid F_{\max}(T'')] \leq \Pr[T' \mid F_{\max}(T')]$ with $|T''| = |T'|$. Here, $\Pr[T'' \mid F_{\max}(T'')] \leq \Pr[T \mid F_{\max}(T)]$ holds for worst trees T'' and T. Thus, $|T| \geq |T''| = |T'|$ holds. ∎

Next, we consider which δ-trees can be the worst. From now on, we assume that the two sons of each internal node in a δ-tree are distinguished as the left and the right sons, respectively.

Definition 11 *A δ-tree T is* clustered *if its rightmost $|T|$ leaves are all labeled "accept".*

For a given clustered tree T, clearly, the following assignment gives $F_{\max}(T)$: Assign $(1-\delta)$ to every edge incident to a right son, and δ to every edge incident to a left son. First, we show two lemmas for a clustered tree.

Lemma 8 *Let T be a clustered tree of depth d. Let $t_0, t_1, \cdots, t_k$ be the integers such that $|T| = 2^{t_k} + \cdots + 2^{t_1} + 2^{t_0}$ with $t_k > \cdots > t_1 > t_0 \geq 0$, which are uniquely determined by the binary representation of $|T|$. Then it holds that:*

$$\Pr[T \mid F_{\max}(T)] = \sum_{i=0}^{k} \delta^i (1-\delta)^{d-t_{k-i}-i}.$$

Proof. For a subtree, its *parent* is the node whose son is the root of the subtree. From given $|T|$ and a δ-tree of depth d with no accepting path, we construct a δ-tree of depth d with $|T|$ accepting paths as follows:

For k: Let T_k be the rightmost subtree of depth t_k of the tree with no accepting path. Change all of the label of the leaves of T_k from "reject" to "accept".

For i ($i = k-1, k-2, \cdots, 0$): Let T'_{i+1} be the subtree whose parent is as same as T_{i+1}. Let T_i be the rightmost subtree of depth t_i of T'_{i+1}. (Note that this step works since $t_{i+1} > t_i$.) Change all of the label of the leaves of T_i from "reject" to "accept".

Since each T_i ($k \geq i \geq 0$) is always taken from rightmost side, we obtain a clustered tree after the construction, and its number of accepting paths is equal to $|T|$. Thus the constructed tree is the same tree as T. The path from the root of T to the root of T_i ($k \geq i \geq 0$) consists of the path from the root of T to the parent of the root of T_k (whose $(d - t_k - 1)$ edges are assigned $(1-\delta)$), an edge to the root of T'_k (which is assigned δ), the path to the parent of the root of T_{k-1} (whose $(t_k - t_{k-1} - 1)$ edges are assigned $(1-\delta)$), an edge to the root of T'_{k-1}, $\cdots$, and the path to the root of T_i. Thus, the probability of the root of T_i ($k \geq i \geq 0$) is given by the product of the probabilities, being equal to $\delta^i(1-\delta)^{d-t_{k-i}-i}$. The constructed clustered tree is a mixture of each T_i. Hence the probability of T is given by the sum of every probability of the root of each T_i with $0 \leq i \leq k$. This implies the lemma. ∎

Lemma 9 *Let T be a clustered tree of depth d with $(s+t)$ accepting paths with $0 \leq s \leq t$. Let T_s (resp. T_t) be a clustered tree of depth $d-1$ with s (resp. t) accepting paths. Let T' be a δ-tree of depth d such that the left son of the root of T' is the root of T_s, and the right son of the root of T' is the root of T_t. Then,*

$$\Pr[T' \mid F_{\max}(T')] \geq \Pr[T \mid F_{\max}(T)].$$

Proof. We show the lemma by induction on the depth of the tree. Since it is clear when $d = 1$ and $d = 2$, we assume $d > 2$.

Let T_{sl} (resp. T_{tl}) be the subtree rooted at the left son of the root of T_s (resp. T_t), and T_{sr} (resp. T_{tr}) be the subtree rooted at the right son of the root of T_s (resp. T_t).

We note that $\Pr[T' \mid F_{\max}(T')] = \delta^2\Pr[T_{sl} \mid F_{\max}(T')] + \delta(1-\delta)\Pr[T_{sr} \mid F_{\max}(T')] + (1-\delta)\delta\Pr[T_{tl} \mid F_{\max}(T')] + (1-\delta)^2\Pr[T_{tr} \mid F_{\max}(T')]$. Thus the probability of T' does not change even if we exchange T_{sr} and T_{tl}. For these four subtrees, four cases arise:

Case 1: $|T_{sl}| > 0$ and $|T_{tl}| = 0$. This case is impossible since $0 \leq s \leq t$.

Case 2: $|T_{sl}| = |T_{tl}| = 0$. By exchanging T_{sr} and T_{tl}, we can regard that only T_t has accepting paths, where $|T_{tl}| = s$ and $|T_{tr}| = t$. Thus, by the inductive hypothesis, $\Pr[T_t \mid F_{\max}(T_t)] \geq \Pr[T'' \mid F_{\max}(T'')]$, where T'' is a clustered tree of depth $d-1$ with $(s+t)$ accepting paths. Thus the lemma holds.

Case 3: $|T_{sl}| > 0$ and $|T_{tl}| > 0$. Since T_s and T_t are clustered trees, Every path of T_{sr} and T_{tr} is an accepting path. Thus, every path of T_t becomes an accepting path by exchanging T_{sr} and T_{tl}. Since $|T_{sl}| = s - 2^{d-2}$ and $|T_{sr}| = t - 2^{d-2}$, by the inductive hypothesis, $\Pr[T_s \mid F_{\max}(T_s)] \geq \Pr[T''' \mid F_{\max}(T''')]$, where T''' is a clustered tree of depth $d-1$ with $(s+t-2^{d-1})$ accepting paths. Thus, combining T''' and T_t, we get a clustered tree of depth d with $(s+t)$ accepting paths. This establishes the lemma.

Case 4: $|T_{sl}| = 0$ and $|T_{tl}| > 0$. Let T_{slr} (resp. T_{srr}, T_{tlr}) be the subtree rooted at the right son of the root of T_{sl} (resp. T_{sr}, T_{tl}), and T_{sll} (resp. T_{srl}, T_{tll}) be the subtree rooted at the left son of the root of T_{sl} (resp. T_{sr}, T_{tl}). Here, T_{slr}, T_{srl} and T_{tll} are exchangeable each other, and so are T_{srr} and T_{tlr}. For these four subtrees, four subcases arise:

Subcase 4a: $|T_{srl}| = |T_{tll}| = 0$. The three edges on the path from the root of T to the root of T_{srr} are assigned δ, $(1-\delta)$ and $(1-\delta)$, respectively. On the other hand, the three edges on the path from the root of T to the root of T_{tll} are assigned $(1-\delta)$, δ and δ, respectively. Thus, even if we exchange T_{srl} and T_{tll}, the probability of T' does not increase. Thus the inductive hypothesis for T_{tl} implies the lemma.

Subcase 4b: $|T_{srl}| = 0$ and $|T_{tll}| > 0$. First, exchanging T_{srl} and T_{tll}, we get $|T_{srl}| > 0$, $|T_{srr}| > 0$ and $|T_{tll}| = 0$. The inductive hypothesis for T_{sr} shows that the probability of T' does not increase even if we replace T_{sr} by a clustered tree with $|T_{sr}|$ accepting paths. If $|T_{srl}| = 0$ holds, this case is reduced to Subcase 4a. If $|T_{srl}| > 0$ holds, then every leaf of T_{srr} is labeled "accept". Since $|T_{tll}| = 0$ and every leaf of T_{tlr} is labeled "accept", exchanging T_{srr} and T_{tll} and further exchanging T_{srl} and T_{srr}, we see that T' is reduced to a clustered tree. Thus, it is sufficient to show that these exchanges do not increase the probability. Let $p = \Pr[T_{srl} \mid F_{\max}(T')]$. Then the change of the probability is equal to $-\delta(1-\delta)^2 + \delta^2(1-\delta) - \delta^2(1-\delta)p + \delta(1-\delta)^2 p = \delta(1-\delta)(1-2\delta)(p-1)$, which is less than 0 for $0 < \delta < \frac{1}{2}$ and $0 < p < 1$. This implies the lemma.

Subcase 4c: $|T_{srl}| > 0$ and $|T_{tll}| = 0$. This case is symmetric to Subcase 4b.

Subcase 4d: $|T_{srl}| > 0$ and $|T_{tll}| > 0$. Exchanging T_{slr} and T_{tll}, the inductive hypothesis for T_s shows that T_s can be replaced by a clustered tree with $|T_s|$ accepting paths. If $|T_{slr}| = 0$, then since $T_{srl} > 0$ and $|T_{tll}| = 0$ hold, this case is reduced to Subcase 4b. Otherwise, if $|T_{slr}| > 0$, then every leaf of T_{sr} is labeled "accept", and $|T_{tll}| = 0$ holds. Here, exchanging T_{sr} and T_{tl} and further exchanging T_{slr} and T_{srl}, we see that T' is reduced to a clustered tree. This implies the lemma. ∎

Now, we are ready to show the main lemma in this section.

Lemma 10 *Every clustered tree is a worst tree.*

Proof. Let T be a given clustered tree of depth d, and T' be any worst tree of depth d with $|T|$ accepting paths. Since T' is the worst, it is sufficient to show that $\Pr[T' \mid F_{\max}(T')] \geq \Pr[T \mid F_{\max}(T)]$. Let T'_l (resp. T'_r) be the subtree rooted at the left (resp. right) son of the root of T'. If either T'_l or T'_r are not the worst, we can improve the probability of T' by replacing them. Since this contradicts that T' is the worst, both T'_l and T'_r are the worst. By the inductive hypothesis, we can replace T'_l (resp. T'_r) by a clustered tree T_l (resp. T_r) with $|T'_l|$ (resp. $|T'_r|$) accepting paths without increasing the probability of T'. Thus, by Lemma 9, $\Pr[T' \mid F_{\max}(T')] \geq \Pr[T \mid F_{\max}(T)]$ holds. ∎

Remark. A δ-tree and a δ-assignment to it correspond to the computation tree of the PTM with a δ-random source. We consider the PTM M with an arbitrary δ-random source such that $L(M) \in {}^{\forall}\delta\text{-PP}$ and the input not in $L(M)$. Then for the δ-tree T corresponding to the computation tree and every δ-assignment F to T, $\Pr[T \mid F] < \frac{1}{2}$ holds. Consequently, $\Pr[T \mid F_{\max}(T)] < \frac{1}{2}$ holds for the δ-tree T. Proposition 7 and Lemma 10 show that the maximum number of accepting paths on the computation of M on input not in $L(M)$ is less than the number of accepting paths of a clustered tree T' with $\Pr[T' \mid F_{\max}(T')] \geq \frac{1}{2}$.

Finally, we proceed to the proof of the main theorem in this section.

Proof of Theorem 5. Theorem 3 shows that ${}^{\forall}\delta\text{-BPP} = \text{BPP}$, and the definition shows that ${}^{\forall}\delta\text{-BPP} \subseteq {}^{\forall}\delta\text{-PP}$. Thus $\text{BPP} \subseteq {}^{\forall}\delta\text{-PP}$ holds for $0 < \delta \leq \frac{1}{2}$. Thus it is sufficient to show that ${}^{\forall}\delta\text{-PP} \subseteq \text{BPP}$. Let M_1 be a PTM with $L(M_1) \in {}^{\forall}\delta\text{-PP}$ for some δ, and $p(n)$ be the depth of the computation path of M_1 on input of length n. Fix an x of length n with $x \notin L(M_1)$.

Let m be a minimum positive integer such that $(1-\delta)(1-\delta^m) \geq \frac{1}{2}$. Since $\lim_{m\to\infty}(1-\delta)(1-\delta^m) = 1-\delta > \frac{1}{2}$, m must be a constant. Without loss of generality, we assume that $p(n) > m+1$. Here we consider a clustered tree T of depth $p(n)$, with $2^{p(n)-1} - 2^{p(n)-m-1}$ accepting paths. Here, since $2^{p(n)-1} - 2^{p(n)-m-1} = 2^{p(n)-2} + 2^{p(n)-3} + \cdots + 2^{p(n)-m-1}$, using Lemma 8, $\Pr[T \mid F_{\max}(T)] = \delta^0(1-\delta)^{p(n)-(p(n)-2)-0} + \delta^1(1-\delta)^{p(n)-(p(n)-3)-1} + \cdots + \delta^{m-1}(1-\delta)^{p(n)-(p(n)-m-1)-(m-1)} = (1-\delta)^2(\delta^0 + \delta^1 + \cdots + \delta^{m-1}) = (1-\delta)(1-\delta^m) \geq \frac{1}{2}$. Thus, as mentioned in the remark above, the number of accepting paths of M_1 on input x is less than $|T|$ equal to $2^{p(n)-1} - 2^{p(n)-m-1}$. Hence if M_1 computes on input x with a perfect random source, M_1 accepts with probability less than or equal to $\frac{2^{p(n)-1}-2^{p(n)-m-1}}{2^{p(n)}} = \frac{1}{2} - \frac{1}{2^{m+1}}$. Since m is a constant, by repeating the algorithm enough times, the probability can be improved to a value less than $\frac{1}{4}$.

For inputs in L, the dual arguments work. That is, the dual arguments show that, for a clustered tree T, $\Pr[T \mid F_{\min}(T)] \geq \Pr[T' \mid F_{\min}(T')]$ holds for every δ-tree T' with $|T| = |T'|$. The number of accepting paths on M_1 is greater than the number of accepting paths of a clustered tree T of depth $p(n)$ such that $\Pr[T \mid F_{\min}(T)] \leq \frac{1}{2}$, and so on. Thus, $L \in \text{BPP}$. ∎

4 Results for ${}^{\exists}\delta$-RP, ${}^{\exists}\delta$-BPP, and ${}^{\exists}\delta$-PP

In this section, we show the result for the classes ${}^{\exists}\delta\text{-RP}$, ${}^{\exists}\delta\text{-BPP}$, and ${}^{\exists}\delta\text{-PP}$.

Theorem 11 ${}^{\exists}\delta\text{-RP} = \text{NP}$ *for* $0 < \delta < \frac{1}{2}$.

Proof. Since ${}^{\exists}\delta\text{-RP} \subseteq \text{NP}$ is shown in Proposition 1, it is sufficient to show that $\text{NP} \subseteq {}^{\exists}\delta\text{-RP}$. Let M_2 be an NTM with $L(M_2) \in \text{NP}$, and $p(n)$ be the length of M_2's computation on input of length n. Let $q(n) = \lceil -\frac{\log(8(p(n)+1))}{\log(2\sqrt{\delta(1-\delta)})} \rceil$. We note that $q(n) > 0$, since $\log(2\sqrt{\delta(1-\delta)}) < 0$ when $0 < \delta < \frac{1}{2}$. We construct a PTM M'_2 which simulates M_2, with a specific δ-random source. M'_2 simulates M_2 straightforwardly if M_2 is not in a nondeterministic state. Otherwise, M'_2 simulates as follows: When M_2 nondeterministically selects 0-choice (resp. 1-choice), M'_2 assigns $(1-\delta)$ to the probability that the outcome of a coin tossing is 0 (resp. 1); tosses a coin $q(n)$ times; and selects i-choice, where i is the majority of the outcomes.

It is clear that M_2' simulates M_2 in time polynomial in n, and M_2' reject x for $x \notin L$. We consider the probability that M_2' accepts x for $x \in L$. M_2' fails to simulate one nondeterministic choice of M_2 with probability equal to $\sum_{i=0}^{\frac{1}{2}q(n)} \binom{q(n)}{i} \delta^{q(n)-i}(1-\delta)^i$. By Lemma 4, since $0 < \frac{1}{2} < (1-\delta) < 1$, we have $\sum_{i=0}^{\frac{1}{2}q(n)} \binom{q(n)}{i} \delta^{q(n)-i}(1-\delta)^i \leq 2^{q(n)\log(2\sqrt{\delta(1-\delta)})} \leq 2^{-\log(8(p(n)+1))} = \frac{1}{8(p(n)+1)}$. Thus the probability that M_2' successes in simulating at most $p(n)$ nondeterministic choices of M_2 is $\geq \left(1 - \sum_{i=0}^{\frac{1}{2}q(n)} \binom{q(n)}{i} \delta^{q(n)-i}(1-\delta)^i\right)^{p(n)} \geq \left(1 - \frac{1}{8(p(n)+1)}\right)^{p(n)}$. Here, $e^{-p} < \left(1 - \frac{p}{n+1}\right)^n$ holds for $0 < p < 1$ and any positive integer n. Thus, $\Pr[M_2' \text{ accepts } x \text{ for } x \in L] \geq \left(1 - \frac{1}{8(p(n)+1)}\right)^{p(n)} > e^{-\frac{1}{8}} > \frac{1}{2}$. Consequently, $L \in {}^{\exists}\delta\text{-RP}$. ∎

Notice that M_2' works for a specific δ-random source with $0 < \delta < \frac{1}{2}$. Intuitively, Theorem 11 shows that the PTM can simulate polynomial number of nondeterministic choices using polynomial number of bits generated by a specific δ-random source with high probability.

To prove that ${}^{\exists}\delta\text{-BPP} = {}^{\exists}\delta\text{-PP} = \text{PSPACE}$ for $0 < \delta < \frac{1}{2}$, we need several definitions and lemmas.

Definition 12 [Pap94] A probabilistic alternating Turing machine *(PATM) is an alternating polynomial time Turing machine M, every computation of which on input x of length n have equal length $2p(n)$ for some polynomial p. The computation alternates between states in two disjoint sets, which we call K_+ and $K_{\max}$.*

Consider a configuration C in a computation of the PATM M. The acceptance count *of configuration C, denoted by $AC(C)$, is defined as follows: If the state of C is an accepting state, then $AC(C) = 1$; if the state of C is a rejecting state, then $AC(C) = 0$; if the state of C is in K_+, then $AC(C) = AC(C_0) + AC(C_1)$, where C_0 and C_1 are the two successor configurations; and if the state of C is in $K_{\max}$, then $AC(C) = \max\{AC(C_0), AC(C_1)\}$, where C_0 and C_1 are the two successor configurations.*

Definition 13 [Pap94] *A language L is in* ABPP *if there exists a PATM M, whose initial configuration is denoted by C_I, such that (1) for $x \in L$, $AC(C_I) \geq \frac{3}{4} \cdot 2^{p(n)}$ and (2) for $x \notin L$, $AC(C_I) \leq \frac{1}{4} \cdot 2^{p(n)}$, where n is the length of the inputs.*

Intuitively, a state in K_+ is a probabilistic state, and a state in $K_{\max}$ is a nondeterministic state. For ABPP, the following lemma holds:

Lemma 12 [Pap94] $\text{ABPP} = \text{PSPACE}$.

Now, we consider how to simulate a PTM with a specific δ-random source by a DTM with polynomial space. When δ can be represented in constant space (e.g., $\delta = \frac{1}{4}$), it is clear that ${}^{\exists}\delta\text{-PP} \subseteq \text{PSPACE}$. We claim that ${}^{\exists}\delta\text{-PP} \subseteq \text{PSPACE}$ even if δ is arbitrary. Since it is not essential in this article, the proof of the claim is shown in appendix A.

Now we are ready to show the theorem.

Theorem 13 ${}^{\exists}\delta\text{-BPP} = {}^{\exists}\delta\text{-PP} = \text{PSPACE}$ *for* $0 < \delta < \frac{1}{2}$.

Proof. Using Lemma 12, it is sufficient to show that $\text{ABPP} \subseteq {}^{\exists}\delta\text{-BPP}$ for $0 < \delta < \frac{1}{2}$, since ${}^{\exists}\delta\text{-BPP} \subseteq {}^{\exists}\delta\text{-PP}$ holds. Let M_3 be a PATM such that $L(M_3) \in \text{ABPP}$, and $2p(n)$ be the length of the computation of M_3 on input x of length n. Without loss of generality, we assume that $AC(C_I) \geq \frac{63}{64} \cdot 2^{p(n)}$ for $x \in L$, and $AC(C_I) \leq \frac{1}{64} \cdot 2^{p(n)}$ for $x \notin L$, where C_I denotes the acceptance count of the initial configuration of M_3. On the computation of M_3, a *state pair* contains a nondeterministic state and the next probabilistic state. That is, a computation of M_3 contains $p(n)$ state pairs.

Here, the PTM M_0' constructed in the proof of Theorem 3 simulates probabilistic choices with an arbitrary δ-random source. On the other hand, the PTM M_2' constructed in the proof of Theorem 11 simulates nondeterministic choices with a specific δ-random source. By combining M_0' and M_2', we construct a PTM M_3' which simulates M_3 with a specific δ-random source. Let $q(n) = \lceil -\frac{\log(8(p(n)+1))}{\log(2\sqrt{\delta(1-\delta)})} \rceil$, and $r(n) = \lceil \frac{3\log p(n)+6}{2\delta - 2\delta^2} \rceil$. A *block* and an *inner product* are defined in the same way as those in the proof of Theorem 3 for $r(n)$. M_3' simulates $2^{r(n)}$ M_3s in parallel to simulate probabilistic choices. The jth state pair of ith simulation of an M_3 is performed by the following:
(Simulation of a nondeterministic choice:)

(1) when M_3 nondeterministically selects l-choice ($l = 0$ or 1), assign $(1-\delta)$ to the probability that the outcome of a coin tossing is l; and

(2) select k-choice, where k is the majority of the outcomes of $q(n)$ coin tossing.

(Simulation of a probabilistic choice:)

(3) generate $r(n)$ bits by a δ-random source in β_j;

(4) select $h_{(i,j)}$-choice, where $h_{(i,j)} = \beta_j \cdot i$;

At the end of the simulation, M_3' accepts if at least half of the $2^{r(n)}$ simulations accepts, or rejects otherwise.

When $x \in L$, as in the proof of Theorem 11, M_3' successes in simulating $p(n)$ nondeterministic choices with probability greater than $e^{-1/8}$. In the case, as in the proof of Theorem 3, the majority of M_3s rejects with probability less than $\frac{1}{8}$. Thus M_3' accepts with probability greater than $e^{-1/8} \cdot \frac{7}{8} > \frac{3}{4}$. On the other hand, when $x \notin L$, as in the proof of Theorem 3, the majority of M_3s accepts with probability less than $\frac{1}{8}$ for every nondeterministic choices. Consequently, M_3' rejects with probability greater than $\frac{3}{4}$. Thus $L \in {}^{\exists}\delta\text{-BPP}$. ∎

In the proof, notice that we can't replace a simulation of a probabilistic choice by a fair coin tossing. Although fair coin tossing works for $x \in L$, it does not work for $x \notin L$, since a probabilistic choice must be simulated for every δ-random source.

5 Concluding Remarks

The outline of the proof of Lemma 12 is the following: Babai introduced "Arthur-Merlin games", and the class AM(*poly*) defined by the games [Bab85]. An Arthur-Merlin game directly corresponds to the computation of a PATM; an Arthur's (resp. Merlin's) turn corresponds to a state in K_+ (resp. $K_{\max}$). Thus it is easy to see that ABPP = AM(*poly*). On the other hand, Goldwasser, Micali, and Rackoff introduced Interactive Proof Systems and the class IP defined by the systems [GMR85], and Goldwasser and Sipser showed that IP = AM(*poly*) [GS86]. Moreover, Shamir showed that PSPACE = IP [Sha90]. Thus ABPP = AM(*poly*) = IP = PSPACE. An Arthur-Merlin games directly corresponds to a language in ABPP, and we have shown that the language is also in ${}^{\exists}\delta$-BPP. The class APP is defined as follows.

Definition 14 [Pap94] *A language L is in* APP *if there exists a PATM M, whose initial configuration is denoted by C_I, such that (1) for $x \in L$, $AC(C_I) > \frac{1}{2} \cdot 2^{p(n)}$ and (2) for $x \notin L$, $AC(C_I) < \frac{1}{2} \cdot 2^{p(n)}$, where n is the length of the inputs.*

We note that in the same way, a "game against Nature" introduced by Papadimitriou [Pap83] directly corresponds to a language in APP, and we can show that the language is also in ${}^{\exists}\delta$-PP.

As noted in section 4, ${}^{\exists}\delta\text{-PP} \subseteq \text{PSPACE} = \text{ABPP}$ holds. In fact, we can show directly that ${}^{\exists}\delta\text{-BPP} \subseteq \text{ABPP}$, using the gadget in Figure 2. Although the gadget in the figure only works for $\delta = \frac{1}{4}$, Knuth and Yao showed that for an arbitrarily rational number δ, we can construct such a gadget [KY76]. On the other hand, we have shown that $\text{ABPP} \subseteq {}^{\exists}\delta\text{-BPP}$ indirectly. In this sense, roughly speaking, ${}^{\exists}\delta$-PP looks like lower than ABPP.

Arthur-Merlin games and games against Nature have alternations. In other words, they are represented by Turing machines which have probabilistic states and nondeterministic states, or by quantified Boolean expressions which have "random" quantifiers and existential quantifiers (e.g., see SSAT in [Pap94]). Alternations disappear if we use the semi-random sources. For instance, we can define a "δ-random" quantifiers and construct a kind of satisfiability problem, being PSPACE-complete, and has only "δ-random" quantifiers.

In the two points noticed above, the result that ${}^{\exists}\delta\text{-BPP} = {}^{\exists}\delta\text{-PP} = \text{PSPACE}$ is an extension of the results in a series of the research for interactive proof systems.

Condon and Ladner investigated interactive proof systems with some restrictions on provers, and showed that some of such systems characterize the classes MA and NP [CL95]. We define an interactive proof system with a restriction on the condition of the probability that system accepts.

Definition 15 *A language L is is* restricted-AM(*poly*) *if there exists a PATM M, whose initial condition is denoted by C_I, such that (1) for $x \in L$, $AC(C_I) \geq \frac{1}{2} \cdot 2^{p(n)}$ and (2) for $x \notin L$, $AC(C_I) = 0$, where n is the length of the inputs.*

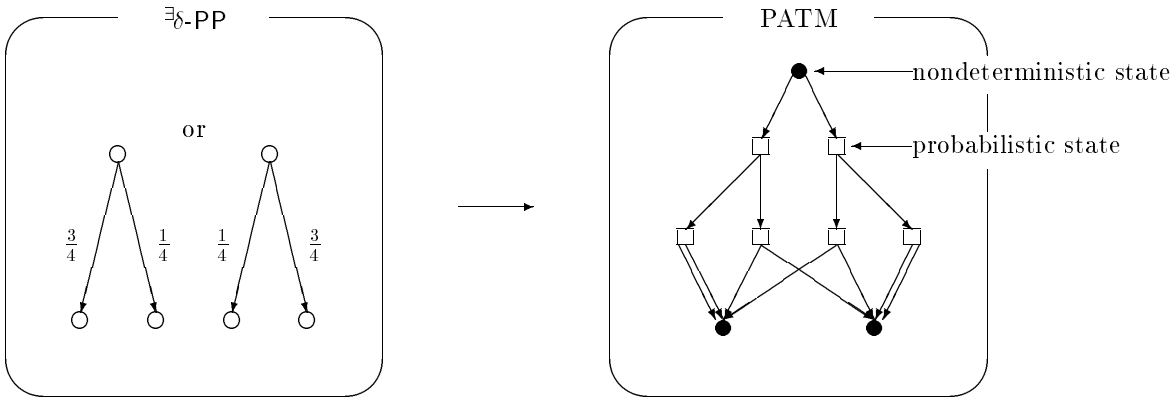

Figure 2: gadget for δ with $\delta = \frac{1}{4}$

It is easy to see that restricted-AM($poly$) $\subseteq$ NP holds. We can show that $^{\exists}\delta$-RP $\subseteq$ restricted-AM($poly$), using the gadget shown in Figure 2. This implies the following corollary:

Corollary 14 restricted-AM($poly$) = NP.

Acknowledgments

I would like to thank Seinosuke Toda, Zhi-Zhong Chen, and Misao Nagayama for many helpful discussions and suggestions.

References

[Bab85] L. Babai. Trading Group Theory for Randomness. In *Proc. 17th ACM Symp. on the Theory of Computing*, pages 421–429. ACM, 1985.

[BDG88] J.L. Balcázar, J. Díaz, and J. Gabarró. *Structural Complexity I.* Springer-Verlag, 1988.

[Che52] H. Chernoff. A Measure of Asymptotic Efficiency for Tests of a Hypothesis Based on the Sum of Observations. *Ann. of Math. Statist.*, 23:493–509, 1952.

[CL95] A. Condon and R. Ladner. Interactive Proof Systems with Polynomially Bounded Strategies. *Journal of Computer and System Science*, 50:506–518, 1995.

[Gil77] J. Gill. Computational Complexity of Probabilistic Turing Machines. *SIAM J. Comput.*, 6(4):675–695, Dec. 1977.

[GMR85] S. Goldwasser, S. Micali, and C. Rackoff. The Knowledge Complexity of Interactive Proof-Systems. In *Proc. 17th ACM Symp. on the Theory of Computing*, pages 291–304. ACM, 1985.

[GS86] S. Goldwasser and M. Sipser. Private Coins versus Public Coins in Interactive Proof Systems. In *Proc. 18th ACM Symp. on the Theory of Computing*, pages 59–68. ACM, 1986.

[HU79] J.E. Hopcroft and J.D. Ullman. *Introduction to Automata Theory, Languages, and Computation.* Addison-Wesley Publishing Company, 1979.

[KY76] D.E. Knuth and A.C. Yao. The complexity of nonuniform random number generation. In J.F.Traub, editor, *Algorithms and Complexity, New Directions and Results*, pages 357–428. Academic, New York, 1976.

[Lut90] J.H. Lutz. Pseudorandom Sources for BPP. *Journal of Computer and System Science*, 41:307–320, 1990.

[Pap83] C.H. Papadimitriou. Games Against Nature. In *Proc. 24th Symp. on Foundations of Computer Science*, pages 446–450. IEEE, 1983.

[Pap94] C.H. Papadimitriou. *Computational Complexity*. Addison-Wesley Publishing Company, 1994.

[Sha90] A. Shamir. IP=PSPACE. In *Proc. 31st Symp. on Foundations of Computer Science*, pages 11–15. IEEE, 1990.

[SV84] M.S. Satha and U.V. Vazirani. Generating Quasi-random Sequences from Slightly random Sources. In *Proc. 25th Symp. on Foundations of Computer Science*, pages 434–440. IEEE, 1984.

[SV86] M.S. Satha and U.V. Vazirani. Generating Quasi-random Sequences from Semi-random Sounces. *Journal of Computer and System Science*, 33:75–87, 1986.

[Vaz86] U. Vazirani. *Randomness, Adversaries and Computation*. PhD thesis, University of California, Berkeley, 1986.

[VV85] U.V. Vazirani and V.V. Vazirani. Random Polynomial Time is Equal to Slightly-random Polynomial Time. In *Proc. 26th Symp. on Foundations of Computer Science*, pages 417–428. IEEE, 1985.

A Proof for $^{\exists}\delta$-PP $\subseteq$ PSPACE

We show the following lemma to deal with arbitrary δ.

Lemma 15 *Let L be a language with $L \in {}^{\exists}\delta$-PP for some δ. Then there exists a number δ' such that; $L \in {}^{\exists}\delta'$-PP and δ' can be represented in polynomial space for the input length.*

Proof. Let L be a language with $L \in {}^{\exists}\delta$-PP for some δ, and M_4 be a PTM such that $L(M_4) = L$. Let p be the length of a computation of M_4. (We write only p, depending on the input length, for short.) Let $d = \frac{\delta^p}{2^{p+1}p(1-\delta)^{p-1}}$. Since d can be represented in polynomial space for the input length, there exists some δ' with $\mid \delta' - \delta \mid$ such that δ' can be represented in polynomial space for the input length. Without loss of generality, we assume that $\delta' > \delta$. We simulate M_4 in polynomial space by substituting δ' for δ. Then the error of the probability of a leaf by substituting is equal to $\delta^i(1-\delta)^{p-i} - \delta'^i(1-\delta')^{p-i}$ for some i with $0 \leq i \leq p$. If the sum of every error of the probability of the leaf is less than half of the probability of any leaf, $L(M_4)$ is also in ${}^{\exists}\delta'$-PP. Here, the minimum probability of a leaf is equal to δ^p. On the other hand, an error of the probability of a leaf is at most $\max\{\delta'^p - \delta^p, (1-\delta)^p - (1-\delta')^p\}$. Two cases arise.
Case 1: $\delta'^p - \delta^p < (1-\delta)^p - (1-\delta')^p$. Since M_4 has 2^p leaves, the sum of every error is at most

$$2^p \mid \delta^i(1-\delta)^{p-i} - \delta'^i(1-\delta')^{p-i} \mid < 2^p\left((1-\delta)^p - (1-\delta')^p\right).$$

Using Taylor series, we get

$$2^p\left((1-\delta)^p - (1-\delta')^p\right) < 2^p pd(1-\delta)^{p-1} = \frac{\delta^p}{2}.$$

Case 2: $\delta'^p - \delta^p < (1-\delta)^p - (1-\delta')^p$. The sum of every error is at most

$$2^p \mid \delta^i(1-\delta)^{p-i} - \delta'^i(1-\delta')^{p-i} \mid < 2^p\left(\delta'^p - \delta^p\right) < 2^p pd\delta^{p-1} = \left(\frac{\delta}{1-\delta}\right)^{p-1}\frac{\delta^p}{2} < \frac{\delta^p}{2}.$$

In each case, the sum of every error of the probability of any computation tree is less than a half of the probability of any leaf. This implies the lemma. ∎

We show the main lemma in this section:

Lemma 16 *For arbitrary δ with $0 < \delta < \frac{1}{2}$, ${}^{\exists}\delta$-PP $\subseteq$ PSPACE.*

Proof. Let L be a language with $L \in {}^{\exists}\delta$-PP for some δ. Let M_5 be a PTM, such that $L = L(M_5)$. Let δ' be a number satisfying the conditions in the proof of Lemma 15. We construct an NTM M_5', which accepts L as follows;

(*i*) nondeterministically compute δ';

(*ii*) simulate all computations of M_5, and counts up its probability by using δ' instead of δ; and

(*iv*) accept if the probability is greater than $\frac{1}{2}$; or reject otherwise.

Clearly, M_5' uses at most polynomial space for the input length, and $L = L(M_5')$. Thus $L \in \mathsf{PSPACE}$. ∎