数理解析研究所講究録
943 巻 1996 年 1-14

Complexity Classes Characterized by Semi-Random Sources

上原 隆平 (Ryuhei Uehara)
Center for Information Science, Tokyo Woman's Christian University,
Zempukuji, Suginami-Ku, Tokyo 167, Japan, uehara@twcu.ac.jp

Abstract

The complexity classes characterized by semi-random sources were investigated. U.V. Vazirani and V.V Vazirani [VV85] showed that $^{\forall}\delta\text{-}\mathsf{RP} = \mathsf{RP}$, and U.V. Vazirani [Vaz86] showed that $^{\forall}\delta\text{-}\mathsf{BPP} = \mathsf{BPP}$, where, for the random class C, the class $^{\forall}\delta\text{-}C$ is a set of all languages which satisfy the condition for C by using any δ-random source. First, we show that

$$^{\forall}\delta\text{-}\mathsf{PP} = \mathsf{BPP},$$

which means that the class PP is weakened by using some semi-random source unless $\mathsf{BPP} = \mathsf{PP}$, whereas RP and BPP don't change by using any semi-random source. The characterization above of the complexity classes by using semi-random source is defined by using *any* δ-random source. We introduce the dual characterization, which is defined by using *some* δ-random source. In other words, for the random class C, the class $^{\exists}\delta\text{-}C$ is defined by the existence of a δ-random source which satisfies the condition for C. Secondly, for these classes, we show that

$$^{\exists}\delta\text{-}\mathsf{RP} = \mathsf{NP}, \text{ and } ^{\exists}\delta\text{-}\mathsf{BPP} = {}^{\exists}\delta\text{-}\mathsf{PP} = \mathsf{PSPACE}.$$

These equations give the new characterization of NP and PSPACE, especially, the characterization for PSPACE improves a series of the research for Interactive Proof System.

1 Introduction

The existence of a fair coin has been extensively assumed for applications such as randomizing algorithms, cryptographic protocols, and stochastic simulation experiments. However, it beset with a difficulty; the available sources of randomness, such as Zener diodes, and Geiger counters are imperfect. They don't output unbiased, independent random bits. J. von Neumann[Neu51] proposed a simple algorithm to extract unbiased flips from an imperfect source, which is the simplest model of an imperfect source of randomness being a coin whose bias is unknown, but fixed. M. Blum[Blu86] considered when the imperfect random source is a deterministic finite state Markov process. M. Santha and U.V. 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]. The model of this random source is also called "SV-model" in [CG88]. This random source is referred as a "*semi-random source*" in this paper. A semi-random source is assumed that the previous bits output by the source can condition the next bit in an arbitrarily bad way. Accordingly, the next bit is output 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 $0 \leq \delta \leq \frac{1}{2}$. More precisely:

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 $x_1, x_2, \cdots$,*
$\delta \leq \Pr[X_i = x_i \mid X_1 = x_1, \cdots, X_{i-1} = x_{i-1}] \leq 1-\delta$.
A semi-random source with parameter δ will be termed δ-random source.

In the paper, they proved that there is no way to generate fair random bits from one semi-random source (U.V. Vazirani[Vaz85] showed how to generate random bits from two independent semi-random sources).

A semi-random source is weak as a random source in a sense as mentioned above. J. Gill[Gil77] defined the classes, such as RP, BPP and PP, by using a fair random source. The influence by using a semi-random source, instead of a fair random source, over these classes has been investigated. (The terminology of the classes below are unified by the author, and it will be clear what a symbol "$\forall$" means in the next paragraph.) The class $^{\forall}\delta$-RP, corresponding to RP, was introduced by U.V. Vazirani and V.V. Vazirani (they referred as SR_p):

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

Notice that the class RP, defined by J. Gill[Gil77], is defined by the definition by letting $\delta = \frac{1}{2}$. In other words, since a $\frac{1}{2}$-random source is a fair random source, $^{\forall}\frac{1}{2}$-RP defines the same class as RP. In the paper, they showed that $^{\forall}\delta$-RP $=$ RP with $0 < \delta \leq \frac{1}{2}$. The class $^{\forall}\delta$-BPP, corresponding to BPP, was introduced by U. Vazirani (he referred as $SBPP$):

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

Notice that $^{\forall}\frac{1}{2}$-BPP defines the same class as BPP. He showed that $^{\forall}\delta$-BPP $=$ BPP with $0 < \delta \leq \frac{1}{2}$ in the paper. The proof of the result is also given by C.H. Papadimitriou in [Pap94], and the result is generalized by B. Chor and O. Goldreich in [CG88], D. Zuckerman in [Zuc91], and A. Srinivasan and D. Zuckerman in [SZ94]. In the same manner as $^{\forall}\delta$-RP and $^{\forall}\delta$-BPP, we introduce the class $^{\forall}\delta$-PP, corresponding to PP:

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

Notice that $^{\forall}\frac{1}{2}$-PP defines the same class as PP. The first theorem in this paper is the following:

Theorem 1

$$\text{For } 0 < \delta < \frac{1}{2},\ ^{\forall}\delta\text{-PP} = \text{BPP}.$$

This result is different from the results for $^{\forall}\delta$-RP being equal to RP, and $^{\forall}\delta$-BPP being equal to BPP. In other words, whereas

RP and BPP are robust for using any semi-random source, PP is weakened by using some semi-random source unless BPP = PP.

The classes $^{\forall}\delta$-RP, $^{\forall}\delta$-BPP, and $^{\forall}\delta$-PP request to satisfy the conditions for *all* δ-random sources. The symbol "$\forall$" means it. In this sense, we can define the *dual* classes characterized by the symbol "$\exists$".

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

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

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

Notice that since a $\frac{1}{2}$-random source is a fair random source, $^{\exists}\frac{1}{2}$-RP ($^{\exists}\frac{1}{2}$-BPP and $^{\exists}\frac{1}{2}$-PP) defines the same class as RP (BPP and PP, respectively). Note that in the definition of $^{\exists}\delta$-BPP and $^{\exists}\delta$-PP, it must be "for all" for $x \notin L$ to make sense. In the definitions above, intuitively, a PTM makes a nondeterministic *and* a probabilistic choice on a coin-tossing state. More precisely, a PTM, on a coin-tossing state, nondeterministically assigns the value between δ and $1-\delta$ to the probability that an outcome of a coin-tossing is head, tosses it, and follows the outcome. The second and the third theorem in this paper are the following:

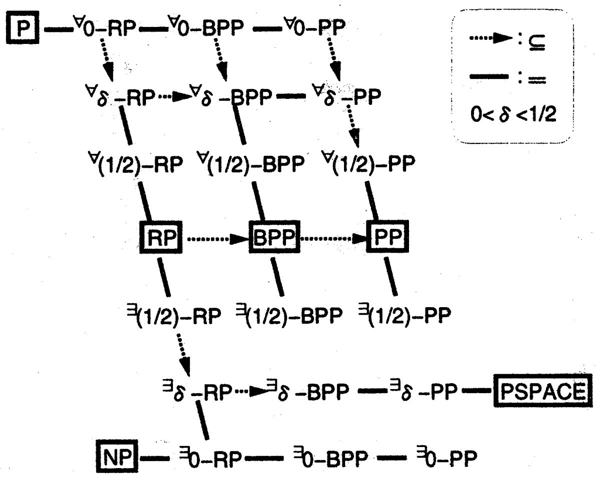
Theorem 2

$$\textit{For } 0 < \delta < \frac{1}{2},\ ^{\exists}\delta\text{-RP} = \text{NP}.$$

Theorem 3

$$\textit{For } 0 < \delta < \frac{1}{2},\ ^{\exists}\delta\text{-BPP} = {}^{\exists}\delta\text{-PP} = \text{PSPACE}.$$

These results give new characterizations for the class NP and PSPACE. Especially, the new characterization for the class PSPACE improves a series of the research for Interactive Proof System [Pap83, Bab85, GMR85, GS86, Sha90], in the sense that, only one kind of quantifier is used. The relations are summarized as follows:



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]. Note that a PTM in this paper, generally, chooses on a coin-tossing state, with probability not equal to $\frac{1}{2}$, as defined in [Gil77]. As mentioned in Introduction, by using a $\frac{1}{2}$-random source being a fair random source, we define the class RP by $^{\forall}(1/2)$-RP (equal to $^{\exists}(1/2)$-RP), BPP by $^{\forall}(1/2)$-BPP (equal to $^{\exists}(1/2)$-BPP), and PP by $^{\forall}(1/2)$-PP (equal to $^{\exists}(1/2)$-PP).

In this paper, without loss of generality, we assume that an NTM or PTM is standardized as follows: Let M be a precise, polynomially bounded NTM or PTM with exactly two choices per step. We denote by $M(x)$ the computation path(s) of M on input x. The two choices available at each step are denoted the *0-choice* and *1-choice*. On input x of length n, the computation $M(x)$ is in effect a full binary tree of depth $p(n)$, where $p(n)$ is some polynomial for n. This tree has $(2^{p(n)+1} - 1)$-many nodes among which there are $2^{p(n)}$-many leaves (corresponding to an accepting state or a rejecting state), and $(2^{p(n)} - 1)$-many internal nodes. The tree has $(2^{p(n)+1} - 2)$-many edges, each corresponding to one of the two choices from an internal node.

We sometimes abbreviate by $*$ for short, e.g. $^{*}\delta$-RP for $^{\forall}\delta$-RP and $^{\exists}\delta$-RP, and $^{\forall}\delta$-$*$ for $^{\forall}\delta$-RP, $^{\forall}\delta$-BPP, and $^{\forall}\delta$-PP. The following proposition is shown by definitions.

Proposition 4 *The following holds for* $^{\forall}\delta$-$*$ *with* $0 < \delta < \frac{1}{2}$*:*

P$=^{\forall}0$-RP $\subseteq ^{\forall}\delta$-RP $\subseteq ^{\forall}(1/2)$-RP $=$RP,
P$=^{\forall}0$-BPP$\subseteq ^{\forall}\delta$-BPP$\subseteq ^{\forall}(1/2)$-BPP$=$BPP, *and*
P$=^{\forall}0$-PP $\subseteq ^{\forall}\delta$-PP $\subseteq ^{\forall}(1/2)$-PP $=$PP.

The following holds for $^{\exists}\delta$-$*$ *with* $0 < \delta < \frac{1}{2}$*:*

RP $=^{\exists}(1/2)$-RP $\subseteq$ $^{\exists}\delta$-RP $\subseteq ^{\exists}0$-RP $=$NP,
BPP$=^{\exists}(1/2)$-BPP, $^{\exists}0$-BPP$=$NP, *and*
PP $=^{\exists}(1/2)$-PP, $^{\exists}0$-PP $=$NP.

Proof. Any 0-assignment gives the probability equal to 1 or 0 to each computation path. Thus for $^{\forall}0$-$*$, all leaves must agree on the outcome, or this algorithm must in fact be deterministic. This implies $^{\forall}0$-RP $=$ $^{\forall}0$-BPP $=$ $^{\forall}0$-PP $=$ P. Conversely, for $^{\exists}0$-$*$, it is sufficient that only one leaf agrees on the outcome, or this algorithm must in fact be nondeterministic. This imply $^{\exists}0$-RP $=$ $^{\exists}0$-BPP $=$ $^{\exists}0$-PP $=$ NP. ∎

Note that the simple inclusion does not hold for $^{\exists}\delta$-BPP and $^{\exists}\delta$-PP, whereas it holds for $^{\exists}\delta$-RP and $^{\forall}\delta$-$*$.

The following results have been shown:

Theorem 5 ([VV85]) *For* $0 < \delta \leq \frac{1}{2}$, $^{\forall}\delta$-RP $=$ RP.

Theorem 6 ([Vaz86]) *For* $0 < \delta \leq \frac{1}{2}$, $^{\forall}\delta$-BPP $=$ BPP.

Since the proof of Theorem 6 in [Pap94] plays an important role in this paper, we show the outline of the proof.

Proof of Theorem 6 ([Pap94]). Let L be a language with $L \in$ 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 $r(n) = \lceil \frac{3 \log p(n)+5}{2\delta - 2\delta^2} \rceil$. (This is referred to as "an important parameter k" in [Pap94, Proof of Theorem 11.4].) A sequence of $r(n)$ bits will be called *block*. The $2^{r(n)}$-many 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 $2^{r(n)}$-many M_0 in parallel. Without loss of generality, we can assume that every computation of M_0 has $p(n)$-many choices. The jth choice of ith simulation of M_0 is performed as follows;

{∗ simulating a probabilistic choice ∗}
generate $r(n)$-many δ-random bits in β_j;
$h_{(i,j)} = \beta_j \cdot i$;
choose $h_{(i,j)}$-choice;

Notice that β_j depends only j. In other words, β_j is used $2^{r(n)}$ times of jth choices on the $2^{r(n)}$-many simulations. At the end of the simulation, M_0' accepts if a majority of $2^{r(n)}$-many 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 $\mid B \mid \leq \frac{1}{32} 2^{p(n)}$. C.H. Papadimitriou have shown in [Pap94, Proof of Theorem 11.4] that

$$\Pr[\ |T \cap B| \geq \frac{1}{2}\ |T|\ ] < \frac{1}{4}.$$

This imply that M_0' accepts with the probability greater than $\frac{3}{4}$ for $x \in L$, and it accepts with the 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 $0 < \delta < \frac{1}{2}$.

Here we show a lemma will be often used in this paper, which is shown by J.H. Luts by using Chernoff Bounds[Che52]:

Lemma 7 ([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 will prove Theorem 1, which states that $^{\forall}\delta\text{-}\mathsf{PP} = \mathsf{BPP}$ for $0 < \delta < \frac{1}{2}$. For a PTM with δ-random source, it is not clear how to assign the probability to the computation paths to maximize the probability that a given PTM accepts. It depends on the distribution of the accepting paths in the computation tree of the PTM. We define some notation to deal with the computation paths which are regarded as a simple full binary tree whose edges are labeled the value between δ and $1-\delta$.

Definition 8 *A* computation tree *is a full binary tree whose leaves are labeled by "accept" or "reject".*

We call the path to a leaf labeled by "accept" (or "reject") is an accepting (or a rejecting, respectively) path. For a computation tree T, we denote by $\mid T \mid$ the number of the accepting paths of T.

Definition 9 *For each δ with $0 \leq \delta \leq \frac{1}{2}$, a δ-*assignment F *to a computation tree is a mapping from the set of edges of the tree to the interval $[\delta, 1-\delta]$ such that the two edges leaving each internal node are assigned numbers adding up to 1.*

Definition 10 *Let T be a computation tree, and F be a δ-assignment to T. The* probability of a node *of T for F is defined by the product of each value which is mapped from the edge, on the path from root to the node, by F. The* probability of T *for F, denoted by* $\Pr[T \mid F]$, *is defined by the sum of every probability of the leaf labeled "accept".*

For a given computation tree, we consider an assignment which maximizes the probability of the tree:

Definition 11 *For a given computation tree T,* a maximal assignment $F_{\max}(T)$ *is defined by the following rules:*

(i) For the node whose sons are leaves; assign $(1-\delta)$ *to an edge incidenting a leaf labeled "accept" and assign* δ *to another edge, if there exists at least one leaf labeled "accept"; or assign* $(1-\delta)$ *and* δ *to edges if both are labeled "reject".*
(ii) For the internal node whose sons are the subtrees whose assignments are already defined; let T_0 *and* T_1 *are the subtrees; assign* $(1-\delta)$ *to the edge incidenting the root of* T_0 *(or* T_1*) and assign* δ *to the edge incidenting the root of* T_1 *(or* T_0*), if* $\Pr[T_0 \mid F_{\max}(T_0)] > \Pr[T_1 \mid F_{\max}(T_1)]$ *(or otherwise, respectively).*

By using the induction for the depth of the tree, it is easily shown that $\Pr[T \mid F_{\max}(T)] \geq \Pr[T \mid F']$ for any δ-assignment F'. Notice that to maximize the probability, it is sufficient to consider δ-assignments which only assign the values δ and $1-\delta$.

Definition 12 *Let* a *be an integer. The computation tree* T *with* a*-many accepting paths is the* worst *if* $\Pr[T \mid F_{\max}(T)] \leq \Pr[T' \mid F_{\max}(T')]$ *holds for any computation tree* T' *with* a*-many accepting paths.*

We note that a worst tree gives the maximal number of the accepting paths for a given probability. To construct a worst tree, we consider to draw the computation tree as a planar tree, whose root is drawn on the top.

Definition 13 *A computation tree* T *with* a*-many accepting paths is* unbalanced *if it can be drawn such that the first* a*th leaves in order from right side are labeled "accept".*

Notice that for a given unbalanced tree T, $F_{\max}(T)$ assigns $(1-\delta)$ to each edge to a right son, and δ to each edge to a left son. Firstly, we show two lemmas for an unbalanced tree.

Lemma 8 *Let* T *be an unbalanced tree of depth* d *with* a*-many accepting paths. Let* $a_0, a_1, \cdots, a_k$ *be the integers such that* $a = 2^{a_k} + \cdots + 2^{a_1} + 2^{a_0}$ *with* $a_k > \cdots > a_1 > a_0 \geq 0$*, which are uniquely determined by the representation of* a *on the binary system. Then it holds that:*

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

Proof. For a subtree, its *parent* is the node whose son is the subtree. For given a, we construct a computation tree of depth d with a-many accepting paths from a computation tree of depth d with no accepting path as follows:

For k: Let T_k be the rightmost subtree of depth a_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 a_i of T_{i+1}. (Note that this step works since $a_{i+1} > a_i$.) 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 an unbalanced tree of depth d after the construction, and its number of accepting paths is equal to a. Thus the constructed tree is the same tree as T. The path to the root of T_i $(k \geq i \geq 0)$ consists of the path to the parent of the root of T_k (whose $(d - a_k - 1)$-many 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 $(a_k - a_{k-1} - 1)$-many 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, equal to $\delta^i(1-\delta)^{d-a_{k-i}-i}$. The constructed unbalanced tree is a mixture of each T_i. Hence the probability of T is given by the sum of the probability of the root of each T_i with $0 \leq i \leq k$. This implies the lemma. ∎

Lemma 9 *Let T be an unbalanced tree of depth d with $(a+b)$-many accepting paths with $0 \leq a \leq b$. Let T_a (or T_b) be an unbalanced tree of depth $d-1$ with a-many (or b-many, respectively) accepting paths. Let T' be the tree of depth d such that the left (or right) son of the root is T_a (or T_b, respectively). Then it holds that;*

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

Proof. We show the lemma by induction for the depth of the tree. Since it is clear when $d = 1$ and $d = 2$, we assume $d > 2$. Let T_{al} (or T_{ar}) be the subtree rooted the left son (or right son, respectively) of the root of T_a, and T_{bl} (or T_{br}) be the subtree rooted the left son (or right son, respectively) of the root of T_b. We note that $\Pr[T' \mid F_{\max}(T')] = \delta^2\Pr[T_{al} \mid F_{\max}(T')] + \delta(1-\delta)\Pr[T_{ar} \mid F_{\max}(T')] + (1-\delta)\delta\Pr[T_{bl} \mid F_{\max}(T')] + (1-\delta)^2\Pr[T_{br} \mid F_{\max}(T')]$. Thus the probability of T' doesn't change by exchanging T_{ar} and T_{bl}. For these four subtrees, four case arises:

Case (*i*). Suppose $|T_{al}| > 0$, $|T_{bl}| = 0$. This case is impossible since $0 \leq a \leq b$.

Case (*ii*). Suppose $|T_{al}| = |T_{bl}| = 0$. In this case, by exchanging T_{ar} and T_{bl}, we can regard that only T_b has accepting paths, where $|T_{bl}| = a$ and $|T_{br}| = b$. Thus, by using inductive hypothesis to T_b of depth $d-1$, and T_{bl} and T_{br}, $\Pr[T_b \mid F_{\max}(T_b)] \geq \Pr[T'' \mid F_{\max}(T'')]$, where T'' is an unbalanced tree of depth $d-1$ with $(a+b)$-many accepting path. Thus lemma holds.

Case (*iii*). Suppose $|T_{al}| > 0$, $|T_{bl}| > 0$. In this case, since T_a and T_b are unbalanced trees, $|T_{ar}| = |T_{br}| = 2^{d-2}$. Thus, by exchanging T_{ar} and T_{bl}, every path of T_b is accepting path. On the other hand, since $|T_{al}| = a - 2^{d-2}$ and $|T_{ar}| = b - 2^{d-2}$, by using inductive hypothesis to T_a of depth $d-1$, $\Pr[T_b \mid F_{\max}(T_b)] \geq \Pr[T''' \mid F_{\max}(T''')]$, where T''' is an unbalanced tree of depth $d-1$ with $(a+b-2^{d-1})$-many accepting path. Thus, since a mixture of T''' and T_b is an unbalanced tree of depth d with $(a+b)$-many accepting path, lemma holds.

Case (*iv*). Suppose $|T_{al}| = 0$, $|T_{bl}| > 0$. Divide T_{al}, T_{ar}, and T_{bl} to T_{alr}, T_{all}, T_{arr}, T_{arl}, T_{blr}, and T_{bll} in the same manner. Here, T_{alr}, T_{arl}, and T_{bll} are exchangeable each other, and so T_{arr} and T_{blr} are. For these four subtrees, four case arises:

Case (*iv*)-(*i*). Suppose $|T_{arl}| = |T_{bll}| = 0$. The edges of the path to the root of T_{arr} are assigned δ, $(1-\delta)$, and $(1-\delta)$. On the other hand, the edges of the path to the root of T_{bll} are assigned $(1-\delta)$, δ, and δ. Thus, by exchanging T_{arl} and T_{blr}, the probability of T' does not increase. Thus by inductive hypothesis for T_{bl}, lemma holds.

Case (*iv*)-(*ii*). Suppose $|T_{arl}| = 0$, and $|T_{bll}| > 0$. First, exchange T_{arl} and T_{bll}. Then $|T_{arl}| > 0$, $|T_{arr}| > 0$, and $|T_{bll}| = 0$ hold. By inductive hypothesis for T_{ar}, T_{ar} can be replaced by an unbalanced tree of as same accepting paths as T_{ar}. If $|T_{arl}| = 0$ then this case can be reduced to the case (*iv*)-(*i*). If $|T_{arl}| > 0$, then $|T_{arl}| > 0$ and $|T_{bll}| = 0$ holds. Moreover, T_{arr} and T_{blr} are the subtrees whose all leaves are labeled "accepted". Let p be the probability equal to $\Pr[T_{arl} \mid F_{\max}(T')]$. Here, first, exchange T_{arr} and T_{bll}, and secondly, exchange

T_{arl} and T_{arr}. By these exchanges, T' reduce to an unbalanced tree. Thus, it is sufficient to show that these exchanges do not increase the probability. The change of the probability by this exchanges is equal to $-\delta(1-\delta)^2+\delta^2(1-\delta)-\delta^2(1-\delta)p+\delta(1-\delta)^2p = \delta(1-\delta)(1-2\delta)(p-1) < 0$ for $0 < \delta < \frac{1}{2}$. This implies lemma.

Case (*iv*)-(*iii*). Suppose $|T_{arl}| > 0$, and $|T_{bll}| = 0$. By exchanging T_{ar} and T_{bl}, this case can be reduced to the case (*iv*)-(*ii*).

Case (*iv*)-(*iv*). Suppose $|T_{arl}| > 0$, and $|T_{bll}| > 0$. First, exchange T_{alr} and T_{bll}. By inductive hypothesis for T_a, T_a can be replaced by an unbalanced tree of as same accepting paths as T_{ar}. If $|T_{alr}| = 0$, then $T_{arl} > 0$ and $|T_{bll}| = 0$ hold. This case can be reduced the case (*iv*)-(*ii*). On the other hand, if $|T_{alr}| > 0$, then T_{ar} is the subtree whose all leaves are labeled "accept" and $|T_{bll}| = 0$ holds. Here, first, exchange T_{ar} and T_{bl}, and secondly, exchange T_{alr} and T_{arl}. Then T' is now an unbalanced tree. This implies lemma. ∎

We show the crucial lemma in this section.

Lemma 10 *Any unbalanced tree is the worst.*

Proof. Let T be a given unbalanced tree depth d with a-many accepting paths, and T' be any worst tree of depth d with a-many 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 (or T'_r) be the subtree, with a_l-many (or a_r-many) accepting paths, rooted the left son (or right son, respectively) of the root of T'. If T'_l (or T'_r) is not the worst, we can improve the probability of T' by replacing it. Thus, T'_l and T'_r are the worst. By inductive hypothesis, we can replace T'_l (or T'_r) by an unbalanced tree T_l (or T_r) with a_l-many (or a_r-many, respectively) accepting path without changing the probability of T'. Thus, by Lemma 9, $\Pr[T' \mid F_{\max}(T')] \geq \Pr[T \mid F_{\max}(T)]$. ∎

Here we show the proof of the main theorem in this section, which states that $^{\forall}\delta$-PP = BPP for $0 < \delta < \frac{1}{2}$.

Proof of Theorem 1. Since $^{\forall}\delta$-BPP = BPP as stated as Theorem 6, and $^{\forall}\delta$-BPP $\subseteq$ $^{\forall}\delta$-PP holds by definition, BPP $\subseteq$ $^{\forall}\delta$-PP for $0 < \delta \leq \frac{1}{2}$. Thus it is sufficient to show $^{\forall}\delta$-PP $\subseteq$ BPP. Let L be a language with $L \in {}^{\forall}\delta$-PP for some δ, and M_1 be a PTM with δ-random source such that $L(M_1) = L$. Let $p(n)$ be the depth of the computation path of M_1 on input of length n. Since $^{\forall}\delta$-PP is clearly closed under complement, we only consider the input x of length n with $x \notin L$. Let a be the number of accepting path of M_1 on input x.

Let m be a positive constant such that $(1-\delta)(1-\delta^m) \geq \frac{1}{2}$. The positive integer m must exist since $\lim_{m\to\infty}(1-\delta)(1-\delta^m) = 1-\delta > \frac{1}{2}$. Without loss of generality, we can assume that $p(n) \gg m$. We consider an unbalanced tree T of depth $m+1$ with (2^m-1)-many accepting paths. Then $\Pr[T \mid F_{\max}(T)] = (1-\delta) - (1-\delta)\delta^m = (1-\delta)(1-\delta^m) \geq \frac{1}{2}$. (This equation is easily seen by the following fact: For the subtree, which rooted the left son of the root of T, every leaf is labeled "reject". For the subtree, which rooted the right son of the root of T, All but one leaf of T_1 is labeled "accept". In other words, the right subtree is an unbalanced tree of depth m with (2^m-1)-many of accepting path.)

By expanding T, an unbalanced tree of depth m', where $m' > m+1$, with $(2^{m'-(m+1)}(2^m-1))$-many of accepting path, has a probability greater than $\frac{1}{2}$. Thus, by the property of the worst tree and Lemma 10, $a \leq 2^{p(n)-(m+1)}(2^m-1) = 2^{p(n)-1} - 2^{p(n)-(m+1)}$. Thus if M_1 compute on input x with a fair 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 the value less than $\frac{1}{4}$. This witnesses $L \in \mathsf{BPP}$. ∎

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

In this section, we will show that $\exists\delta$-RP = NP, and $\exists\delta$-BPP = $\exists\delta$-PP = PSPACE for $0 < \delta < \frac{1}{2}$. First, we show the proof of Theorem 2, which states $\exists\delta$-RP = NP.

Proof of Theorem 2. It is sufficient to show that NP $\subseteq$ $\exists\delta$-RP. Let L be a language with $L \in$ NP, and M_2 be an NTM such that $L(M_2) = L$. Let $p(n)$ be the length of M_2's computation on input of length n. Let $q(n)$ be a polynomial of n defined as follows;

$$q(n) = \lceil -\frac{\log(2(p(n)+1))}{\log(2\sqrt{\delta(1-\delta)})} \rceil.$$

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', simulating M_2, with a δ-random source. M_2' simulates M_2 straightforwardly if M_2 is not in a nondeterministic state. Otherwise, M_2' simulates as follows;

(*i*) when M_2 nondeterministically chooses 0-choice (or 1-choice), nondeterministically assign $(1-\delta)$ to the probability that the outcome of a coin tossing is 0 (or 1, respectively); and

(*ii*) choose i-choice, where i is a majority of the outcomes of $q(n)$-many coin tossing.

It is clear that M_2' simulates M_2 in polynomial time of n, and M_2' reject x for $x \notin L$. We consider the probability that M_2' accepts x for $x \in L$. On the step (*ii*), M_2' gets a wrong answer with probability $\sum_{i=0}^{\frac{1}{2}q(n)} \binom{q(n)}{i} \delta^{q(n)-i}(1-\delta)^i$. By Lemma 7, since $0 < \frac{1}{2} < (1-\delta) < 1$,

$$\begin{aligned}
&\sum_{i=0}^{\frac{1}{2}q(n)} \binom{q(n)}{i} \delta^{q(n)-i}(1-\delta)^i \\
&\leq 2^{q(n)\log(2\sqrt{\delta(1-\delta)})} \\
&\leq 2^{-\log(2(p(n)+1))} = \frac{1}{2(p(n)+1)}.
\end{aligned}$$

Thus M_2' successes to simulate at most $p(n)$-many nondeterministic choices of M_2 with probability greater than

$$\left(1 - \sum_{i=0}^{\frac{1}{2}q(n)} \binom{q(n)}{i} \delta^{q(n)-i}(1-\delta)^i\right)^{p(n)} \geq \left(1 - \frac{1}{2(p(n)+1)}\right)^{p(n)}.$$

Here, $e^{-p} < \left(1 - \frac{p}{n+1}\right)^n$ holds for $0 < p < 1$ and any positive integer n. (This is proved by as follows: For the sequence defined by $a_n(p) = (1 - \frac{p}{n+1})^n$, it is easy to check $e^{-p} < a_1(p)$ and $\lim_{n\to\infty} a_n = e^{-p}$. Since $\frac{n+d}{m+d} > \frac{n}{m}$ holds for $m > n > 0$ and $d > 0$, $\frac{a_{n-1}(p)}{a_n(p)} = \left(\frac{n-p}{n}\right)^{n-1}\left(\frac{n+1}{n+1-p}\right)^n > \left(\frac{n-p}{n}\right)^{n-1}\left(\frac{n}{n-p}\right)^n = \frac{n}{n-p} > 1$.) Thus, $\Pr[M_2' \text{ accepts } x \text{ when } x \in L] \geq \left(1 - \frac{1}{2(p(n)+1)}\right)^{p(n)} > e^{-\frac{1}{2}} > \frac{1}{2}$, consequently, $L \in \exists\delta$-RP. ∎

Secondly, we show that $\exists\delta$-BPP = $\exists\delta$-PP = PSPACE. To this end, we introduce a probabilistic alternating Turing machine and the class ABPP defined by C.H. Papadimitriou:

Definition 14 ([Pap94]) A probabilistic alternating Turing machine *(PATM) is an alternating polynomial time Turing machine M, all the computations of which on input x* *of length n have equal length $2p(n)$ for some polynomial p, and the number of nondeterministic choices is uniformly two. Furthermore, the computation strictly alternates between states in two disjoint sets, which we shall now call K_+ and $K_{\max}$.*

Consider a configuration C in a computation of the PATM M. The acceptance count *of configuration C is defined as follows: If the state of C is an accepting state, then its count is 1; if the state of C is a rejecting state, then its count is 0; if the state of C is in K_+, then its count is the sum of the acceptance counts of the two successor configurations; and if the state of C is in $K_{\max}$, then its count is the maximum between the two acceptance counts of the two successor configuration.*

The class ABPP *contains all languages L for which there is a PATM M with the following property: For all input x of length n, if $x \in L$ then the acceptance count of the initial configuration of M is at least $\frac{3}{4} \cdot 2^{p(n)}$; and if $x \notin L$ then the acceptance count of the initial configuration of M is at most $\frac{1}{4} \cdot 2^{p(n)}$.*

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 11 ([Pap94]) ABPP = PSPACE.

The outline of the proof of Lemma 11 is the following: L. Babai[Bab85] introduced "Arthur vs. Merlin games", and the class AM(*Poly*) defined by the games. An Arthur vs. Merlin game directly corresponds to the computation of a PATM; an Arthur's turn corresponds to a state in K_+, and a Merlin's turn corresponds to a state in $K_{\max}$. Thus we can easily see that ABPP = AM(*Poly*). On the other hand, S. Goldwasser, S. Micali, and C. Rackoff[GMR85] introduced Interactive Proof Systems and the class IP defined by the systems, and S. Goldwasser and M. Sipser[GS86] showed that IP = AM(*Poly*). Moreover, A. Shamir[Sha90] showed that PSPACE = IP. Thus ABPP = AM(*Poly*) = IP = PSPACE.

When a Turing machine simulates δ-random source without such a source, it is not clear how to simulate it in polynomial space, if δ can not be represented in polynomial space. Since it is not essential in this article, we will show how to simulate it in polynomial space in Appendix A. By Appendix A, without loss of generality, we assume that δ can be represented in constant space. For such a δ, it is clear that $^{\exists}\delta$-PP $\subseteq$ PSPACE. Moreover, it is clear that $^{\exists}\delta$-BPP $\subseteq$ $^{\exists}\delta$-PP by definition. Thus, Theorem 2, which states $^{\exists}\delta$-BPP = $^{\exists}\delta$-PP = PSPACE for $0 < \delta < \frac{1}{2}$, is proved by the following lemma:

Lemma 12 PSPACE $\subseteq$ $^{\exists}\delta$-BPP *with* $0 < \delta < \frac{1}{2}$.

Proof. By Lemma 11, it is sufficient to show that ABPP $\in$ $^{\exists}\delta$-BPP. Let L be a language with $L \in$ ABPP, and M_3 be a PATM such that $L(M_3) = L$. On input x of length n, let $2p(n)$ be the length of M_3's computation on x. Without loss of generality, we can assume that the acceptance count of the initial configuration of M_3 is at least $\frac{63}{64} \cdot 2^{p(n)}$ if $x \in L$, and at most $\frac{1}{64} \cdot 2^{p(n)}$ if $x \notin L$. On the computation of M_3, we call a *pair* of states a probabilistic state and a nondeterministic state following it. A computation of M_3 contains $p(n)$-many pairs of states.

The PTM M_0', constructed in Proof of Theorem 6, simulates probabilistic choices by using any δ-random source. On the other hand, the PTM M_2', constructed in Proof of Theorem 2, simulates nondeterministic choices by using a δ-random source. By putting M'_0 and M'_2 together, we construct a PTM M'_3, which simulates M_3 with a δ-random source.

Let $q(n) = \lceil \frac{-\log(15(p(n)+1))}{\log(2\sqrt{\delta(1-\delta)})} \rceil$, and $r(n) = \lceil \frac{3\log p(n)+6}{2\delta-2\delta^2} \rceil$. (Notice that these functions are slightly changed to improve the probability.) A *block* and an *inner product* are defined as same as in Proof of Theorem 6 for $r(n)$. M'_3 simulates $2^{r(n)}$-many M_3 in parallel to simulate probabilistic choices. The jth pair of ith simulation of M_3 is performed by the following a pair of simulations:

(Simulation for a probabilistic choice:)

(*i*) generate $r(n)$-many δ-random bits in β_j;

(*ii*) choose $h_{(i,j)}$-choice, where $h_{(i,j)} = \beta_j \cdot i$;

(Simulation for a nondeterministic choice:)

(*i*) when M_3 nondeterministically chooses 0-choice (or 1-choice), nondeterministically assign $(1-\delta)$ to the probability that the outcome of a coin tossing is 0 (or 1, respectively); and

(*ii*) choose i-choice, where i is a majority of the outcomes of $q(n)$-many coin tossing.

At the end of the simulation, M'_3 accepts if a majority of $2^{r(n)}$-many simulations accepts, or rejects otherwise.

Assume $x \in L$. Proof of Theorem 2 implies that M'_3 successes $p(n)$-many simulations for nondeterministic choices with probability greater than $\frac{6}{7}$. In this case, Proof of Theorem 6 implies that M'_3 outputs correct answer with probability greater than $\frac{7}{8}$. Thus M'_3 accepts with probability greater than $\frac{7}{8} \cdot \frac{6}{7} = \frac{3}{4}$. Next, assume $x \notin L$. By hypothesis, M_3 rejects x with probability greater than $\frac{63}{64}$ for any nondeterministic choices. Thus, Proof of Theorem 6 implies that M'_3 outputs correct answer with probability greater than $\frac{7}{8}$ for any nondeterministic choices. Therefore, M'_3 rejects with probability greater than $\frac{7}{8}$, consequently, M_3 accepts with probability less than $\frac{1}{4}$. Thus $L \in \exists\delta$-BPP. ∎

5 Concluding Remarks

An "Arthur vs. Merlin games" introduced by L. Babai[Bab85] directly corresponds to a language in ABPP, and we have shown that the language is also in $\exists\delta$-BPP. We note that, in the same way, a "game against Nature" introduced by C.H. Papadimitriou[Pap83] directly corresponds to a language in APP, and we can show that the language is also in $\exists\delta$-PP. (The class APP, which is introduced by C.H. Papadimitriou in [Pap94], is a class as against ABPP, in the same manner as the class PP as against BPP.)

The games above have alternations. In other words, they are represented by Turing machines which have probabilistic states and nondeterministic states, and by quantified Boolean expressions which have "random" quantifiers and existential quantifiers (e.g., see SSAT in [Pap94]). The alternations are missing by using the semi-random sources. For instance, we can define a "δ-random" quantifiers and construct a kind of satisfiability problem, which is PSPACE-complete, and has only "δ-random" quantifiers.

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.

[Blu86] M. Blum. Independent Unbiased Coin Flips from a Correlated Biased Source – a Finite State Markov Chain. *Combinatorica*, 6(2):97–108, 1986.

[CG88] B. Chor and O. Goldreich. Unbiased Bits from Sources of Weak Randomness and Probabilistic Communication Complexity. *SIAM J. Comput.*, 17(2):230–261, April 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.

[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.

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

[Neu51] J. von Neumann. Various techniques used in connection with random digits. Notes by G.E.Forsythe. National Bureau of Standards. *Applied Math Series*, 12:36–38, 1951. Reprinted in von Neumann's Collected Works 5 (Pergamon Press, 1963), 768–770.

[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 Sciences*, 33:75–87, 1986.

[SZ94] A. Srinivasan and D. Zuckerman. Computing with Very Weak Random Sources. In *Proc. 35th Symp. on Foundations of Computer Science*, pages 264–275. IEEE, 1994.

[Vaz85] U.V. Vazirani. Towards a Strong Communication Complexity theory or Generating Quasi-Random Sequence from Two Communicating Slightly-random Sources. In *Proc. 17th ACM Symp. on the Theory of Computing*, pages 366–378. ACM, 1985.

[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.

[Zuc91] D. Zuckerman. Simulating BPP Using a General Weak Random Source. In *Proc. 32nd Symp. on Foundations of Computer Science*, pages 79–89. IEEE, 1991.

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

To deal with δ, an *arbitrary* number, we show the following lemma:

Lemma 13 *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 depth of the computation of M_4. (We write only p, which depends on the input length, for short.) Let $d = \frac{\delta^p}{2^{p+1}p(1-\delta)^{p-1}}$. We consider an approximate value δ' to δ by taking $|\,\delta' - \delta\,| < d$. Since d can be represented in polynomial space for the input length, there exists a δ' which also can be represented in polynomial space for the input length. It is sufficient to show that the error of the probability of any computation tree, which is made by replacing δ by δ', is less than a half of the probability of any leaf of a computation tree.

Without loss of generality, we assume that $\delta' > \delta$. The probability of a leaf with δ-random source is equal to $\delta^i(1-\delta)^{p-i}$ for some i with $0 \le i \le p$. Thus, the minimal probability of a leaf is equal to δ^p. On the other hand, an error of the probability of a leaf, which is made by replacing δ by δ', is at most $\max\{\delta'^p - \delta^p, (1-\delta)^p - (1-\delta')^p\}$. Two cases arise.

(Case 1.) Assume $\delta'^p - \delta^p < (1-\delta)^p - (1-\delta')^p$. Since M_4 has 2^p-many leaves, the error of the probability of a computation tree is at most

$$
\begin{aligned}
2^p\ & |\delta^i(1-\delta)^{p-i} - \delta'^i(1-\delta')^{p-i}| \\
& < 2^p\left((1-\delta)^p - (1-\delta')^p\right) \\
& < 2^p pd(1-\delta)^{p-1}.
\end{aligned}
$$

The last line is obtained by using Taylor series. Here, by substituting for d, $2^p pd(1-\delta)^{p-1} = \frac{\delta^p}{2}$.

(Case 2.) Assume $\delta'^p - \delta^p < (1-\delta)^p - (1-\delta')^p$. Then the error of the probability of a computation tree is at most

$$\begin{aligned} 2^p & \left| \delta^i (1-\delta)^{p-i} - \delta'^i (1-\delta')^{p-i} \right| \\ & < 2^p \left(\delta'^p - \delta^p \right) < 2^p p d \delta^{p-1} \\ & = \left(\frac{\delta}{1-\delta} \right)^{p-1} \frac{\delta^p}{2} < \frac{\delta^p}{2}. \end{aligned}$$

In each case, it is shown that the 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 14

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 an approximate value to δ given by using Lemma 13. 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 $L \in$ PSPACE. ∎