

Equivalence of the UPP Classes

Chris Calabro *

March 31, 2002

If $a, b \in \mathbb{R}$ let (a, b) , $[a, b]$, $[a, b)$, $(a, b]$ denote the open, closed, and half open intervals from a to b . Let Σ be a finite alphabet of at least 2 symbols and let Σ^* be the set of all finite strings over Σ . If M is a randomized Turing machine and $x \in \Sigma^*$, then let $AccPr_M(x)$ be the accept probability of $M(x)$.

Let $p \in (0, 1)$. Then Unbounded Probabilistic Polynomial time of p is

$$\text{UPP}(p) = \left\{ L \subseteq \Sigma^* \mid \exists \text{ randomized Turing machine } M, \text{ polynomial } t \right. \\ \left. \forall x \in \Sigma^* \left((x \in L \rightarrow AccPr_M(x) > p) \right. \right. \\ \left. \left. \wedge (x \notin L \rightarrow AccPr_M(x) < p) \right. \right. \\ \left. \left. \wedge \text{time } M(x) \leq t(|x|) \right) \right\}.$$

Also

$$\text{coUPP}(p) = \{ L \subseteq \Sigma^* \mid \bar{L} \in \text{UPP}(p) \}.$$

Lemma 1. $\forall p \in (0, 1)$ $\text{UPP}(p) = \text{coUPP}(1 - p)$.

Proof. Suppose $L \in \text{UPP}(p)$ and M is a poly-time randomized Turing machine such that

$$\forall x \in \Sigma^* \left((x \in L \rightarrow AccPr_M(x) > p) \right. \\ \left. \wedge (x \notin L \rightarrow AccPr_M(x) < p) \right).$$

Let M' be the same as M but with accept and reject states swapped. Then for each $x \in \Sigma^*$

$$(x \in L \rightarrow AccPr_{M'}(x) = 1 - AccPr_M(x) < 1 - p) \\ \wedge (x \notin L \rightarrow AccPr_{M'}(x) = 1 - AccPr_M(x) > 1 - p).$$

So $\bar{L} \in \text{UPP}(1 - p) \Rightarrow L \in \text{coUPP}(1 - p)$. So $\text{UPP}(p) \subseteq \text{coUPP}(1 - p)$. Showing that $\text{coUPP}(1 - p) \subseteq \text{UPP}(p)$ is symmetric. \square

*in collaboration with Dana Dahlstrom

Suppose that we want to approximate a Bernoulli random variable of parameter $\frac{a}{b} \in \mathbb{Q}$ with a randomized Turing machine. By this we mean we want a randomized Turing machine which ignores its input and accepts with probability as close to $\frac{a}{b}$ as possible. If $\frac{a}{b}$ is a diadic decimal, that is, of the form $\frac{i}{2^j}$ for some $i, j \in \mathbb{Z}^{\geq 0}$, then this can be done exactly; but if not, the approximation can only be so close given a limited amount of computation time. We will need upper and lower bounds on the error.

Let us say that a probability sample space of size n is *uniform* iff each point in the space has probability $\frac{1}{n}$.

Lemma 2. Let $\frac{a}{b} \in \mathbb{Q} \cap [0, 1]$, $n \in \mathbb{Z}^{\geq 0}$. Define

$$\text{error}_* = \min \left\{ \frac{a}{b} - E(X) \mid X \text{ is a Bernoulli random variable defined on a uniform sample space of size } 2^n \text{ and } E(X) \leq \frac{a}{b} \right\}.$$

Then $\text{error}_* = \frac{a2^n \bmod b}{b2^n} \in [0, (\frac{1}{2})^n)$, and if $\text{error}_* \neq 0$, then $\text{error}_* \geq \frac{1}{b2^n}$. Furthermore, the output of the following randomized Turing machine, which reads n random bits, is a Bernoulli random variable achieving this error.

let $c = n$ random bits
return $(c < \lfloor \frac{a2^n}{b} \rfloor)$

Proof.

$$\begin{aligned} \text{error}_* &= \frac{a}{b} - \frac{\lfloor \frac{a2^n}{b} \rfloor}{2^n} \\ &= \frac{a}{b} - \frac{a2^n - (a2^n \bmod b)}{b2^n} \\ &= \frac{a2^n \bmod b}{b2^n} \in [0, (\frac{1}{2})^n). \end{aligned}$$

The remaining conclusions are obvious. □

Lemma 3. Let $\frac{a}{b} \in \mathbb{Q} \cap [0, 1]$, $n \in \mathbb{Z}^{\geq 0}$. Define

$$\text{error}^* = \min \left\{ E(X) - \frac{a}{b} \mid X \text{ is a Bernoulli random variable defined on a uniform sample space of size } 2^n \text{ and } E(X) \geq \frac{a}{b} \right\}.$$

Then $\text{error}^* = \frac{-a2^n \bmod b}{b2^n} \in [0, (\frac{1}{2})^n)$, and if $\text{error}^* \neq 0$, then $\text{error}^* \geq \frac{1}{b2^n}$. Furthermore, the output of the following randomized Turing machine, which reads n random bits, is a Bernoulli random variable achieving this error.

let $c = n$ random bits
return $(c < \lceil \frac{a2^n}{b} \rceil)$

Proof.

$$\begin{aligned}
error^* &= \frac{\lceil \frac{a2^n}{b} \rceil}{2^n} - \frac{a}{b} \\
&= \frac{\frac{a2^n + (-a2^n \bmod b)}{b}}{2^n} - \frac{a}{b} \\
&= \frac{-a2^n \bmod b}{b2^n} \in [0, \left(\frac{1}{2}\right)^n).
\end{aligned}$$

The remaining conclusions are obvious. \square

Lemma 4. *Let $p, q \in \mathbb{Q} \cap (0, \frac{1}{2}]$, $p \leq 2q$. Then $UPP(p) \subseteq UPP(q)$.*

Proof. First write $p = \frac{f}{g}$ where $f, g \in \mathbb{Z}^{>0}$. Define $p_y = 2q - p$. From our assumptions, $p_y \in \mathbb{Q} \cap [0, 1)$. So write $p_y = \frac{a}{b}$ where $a \in \mathbb{Z}^{>0}, b \in \mathbb{Z}^{>0}$.

Let $L \in UPP(p)$. Then \exists poly-time randomized Turing machine M such that

$$\begin{aligned}
\forall x \in \Sigma^* &\left((x \in L \rightarrow AccPr_M(x) > p) \right. \\
&\quad \left. \wedge (x \notin L \rightarrow AccPr_M(x) < p) \right).
\end{aligned}$$

We will define a new randomized Turing machine M' which behaves as follows on input x .

$X = M(x)$
 $t = \text{time } M(x) \text{ took}$
 $n = t + g$
 $c = n \text{ random bits}$
 $Y = (c < \lfloor \frac{a2^n}{b} \rfloor)$
 $Z = 1 \text{ random bit (different from those used for } c)$
if $X = Y = 1$, accept
else if $X = Y = 0$, reject
else return Z .

Let $p_x = AccPr_M(x)$, $p_z = P(Z = 1)$. From lemma 2, Y is Bernoulli with parameter $p_y - error_*$ where $error_* \in [0, (\frac{1}{2})^n)$. So for each $x \in \Sigma^*$

$$\begin{aligned}
&P(M'(x) \text{ accepts}) \\
&= p_x(p_y - error_*) + (p_x(1 - p_y + error_*) + (1 - p_x)(p_y - error_*))p_z \\
&= p_x p_y - p_x error_* + \frac{1}{2}(p_x - p_x p_y + p_x error_* + p_y - error_* - p_x p_y + p_x error_*) \\
&= \frac{1}{2}(p_x + p_y - error_*).
\end{aligned}$$

Suppose $x \in L$. Then $p_x = p + error^*$ for some $error^* > 0$. If p is a diadic decimal, then $error^* \geq (\frac{1}{2})^t$; if not, then from lemma 3, $error^* \geq \frac{1}{g2^t}$. In either

case,

$$\begin{aligned}
& \lg g < g \\
\Rightarrow & t + \lg g < t + g = n \\
\Rightarrow & g2^t < 2^n \\
\Rightarrow & \left(\frac{1}{2}\right)^n < \frac{1}{g2^t} \\
\Rightarrow & error_* < \left(\frac{1}{2}\right)^n < \frac{1}{g2^t} \leq error^*.
\end{aligned}$$

So

$$\begin{aligned}
P(M'(x) \text{ accepts}) &= \frac{1}{2}(p + error^* + 2q - p - error_*) \\
&= q + \frac{1}{2}(error^* - error_*) > q.
\end{aligned}$$

Next suppose $x \notin L$. Then $p_x = p - error^*$ for some $error^* > 0$. So

$$\begin{aligned}
P(M'(x) \text{ accepts}) &= \frac{1}{2}(p - error^* + 2q - p - error_*) \\
&= q - \frac{1}{2}(error^* + error_*) < q.
\end{aligned}$$

So indeed $L \in \text{UPP}(q)$ and the proof is complete. □

Lemma 5. $\forall p, q \in \mathbb{Q} \cap (0, \frac{1}{2}] \text{ UPP}(p) = \text{UPP}(q)$.

Proof. From lemma 4, we have

$$\forall i \in \mathbb{Z}^{>0}, p, q \in \mathbb{Q} \cap \left[\left(\frac{1}{2}\right)^{i+1}, \left(\frac{1}{2}\right)^i\right] \text{ UPP}(p) = \text{UPP}(q).$$

By induction, we have

$$\forall i \in \mathbb{Z}^{>0}, p, q \in \mathbb{Q} \cap \left[\left(\frac{1}{2}\right)^{i+1}, \frac{1}{2}\right] \text{ UPP}(p) = \text{UPP}(q).$$

The result follows. □

Lemma 6. $\forall p, q \in \mathbb{Q} \cap (0, \frac{1}{2}] \text{ coUPP}(p) = \text{coUPP}(q)$.

Proof.

$$\begin{aligned}
L \in \text{coUPP}(p) &\Rightarrow \overline{L} \in \text{UPP}(p) \\
&\Rightarrow \overline{L} \in \text{UPP}(q) && \text{by lemma 5} \\
&\Rightarrow L \in \text{coUPP}(q).
\end{aligned}$$

□

Lemma 7. $\forall p, q \in \mathbb{Q} \cap [\frac{1}{2}, 1)$ $\text{UPP}(p) = \text{UPP}(q)$.

Proof.

$$\begin{aligned} \text{UPP}(p) &= \text{coUPP}(1 - p) && \text{by lemma 1} \\ &= \text{coUPP}(1 - q) && \text{by lemma 6} \\ &= \text{UPP}(q) && \text{by lemma 1.} \end{aligned}$$

□

Theorem 8. $\forall p, q \in \mathbb{Q} \cap (0, 1)$ $\text{UPP}(p) = \text{UPP}(q) = \text{coUPP}(p) = \text{coUPP}(q)$.

Proof. From lemmas 1, 5, 7. □

Thus it makes sense to talk about UPP without the parameter p .

Open Problems

- Are there any languages which distinguish UPP from more commonly studied languages such as BPP?
- Are the classes $\text{UPP}(p)$ all the same for arbitrary reals $p \in (0, 1)$? Keep in mind that not all reals are computable since there are uncountably many reals in $(0, 1)$ while there are only countably many algorithms.