Randomized Complexity Classes

**ZPP** = all languages (i.e. decision problems) with Las Vegas algorithms in expected polytime

**RP** = all languages \( L \) with one-sided Monte Carlo algorithms in worst-case polytime

\[
\begin{align*}
\text{St.: } & \text{Input } x, \\
& \left\{ \begin{array}{ll}
\text{if } x \in L \Rightarrow & \Pr[A \text{ outputs yes }] \geq \frac{1}{2} \\
\text{if } x \notin L \Rightarrow & \Pr[A \text{ outputs no }] = 1
\end{array} \right.
\end{align*}
\]

**Rmk.:** \( \frac{1}{2} \) can be changed to any constant \( \epsilon \in (0,1) \) by repeating & taking OR of output (with \( t \) iterations, err prob \( \leq \frac{1}{2^t} \)).

(e.g. Miller-Rabin: \textsc{Composite} \( \in \textbf{RP} \))

Adleman-Huang: \textsc{"\u0105"} \( \in \textbf{ZPP} \)

AKS: \textsc{\u0105} \( \in \textbf{P} \)

**Fact 1.** \( P \subseteq \textbf{ZPP} \subseteq \textbf{RP} \) (by Markov).

**Fact 2.** \( \textbf{RP} \subseteq \textbf{NP} \).

**Pf:** Certificate = seq of rand bits. \( \pentagram \)

**Fact 3.** \( \textbf{ZPP} = \textbf{RP} \cap \textbf{co-RP} \).

**Pf.** \( \pentagram \) since \( \textbf{ZPP} = \textbf{co-ZPP} \).
Pf: (C) ... since ZPP = co-ZPP.

(2) Suppose we have 2 one-sided Mont-Carlo algms $A$ for $L$
and $A'$ for $L^c$.

run both: if $A$ says no, know $x \notin L$.
if $A'$ says no, know $x \in L$.

$\mathbb{w.prob} \leq \frac{1}{2}$ $\implies$ otherwise repeat

$\text{RP}$ is the set of languages with

*iterations $\leq 2$.

$\text{BPP} = \text{all languages } L \text{ with }$ 2-sided Monte-Carlo algms in Poly-time

Sit. A input $x$,
if $x \in L \implies \Pr[A \text{ outputs yes on } x] > \frac{3}{4}$
if $x \notin L \implies \Pr[A \text{ outputs no on } x] > \frac{1}{2}$

Rmk: $\frac{3}{4}$ can be changed to any const $\epsilon (\frac{1}{2}, 1)$.
by repeating & taking majority of output

( with $t$ repetitions, err prob $\leq \frac{1}{2^{O(t)}}$ by Chernoff bd...)

( if exactly $\frac{1}{2}$, we get different class PP )
Fact 4 \( \text{RP} \leq \text{BPP} \). (also, \( \text{co-RP} \leq \text{BPP} \), since \( \text{BPP} = \text{co-BPP} \))

Upper bd for \( \text{BPP} \)?

Known: (Sipser-Sac, Lauterman '83) \( \text{BPP} \leq \text{NP} \cap \text{co-NP} \)

In fact, \( \text{BPP} \leq \text{ZPP} \cap \text{NP} \)

In fact, \( \text{MA} \leq \text{ZPP} \cap \text{NP} \) (Goldreich-Forteman '97)

Possibility of general derandomization?

Then (Adelman '78) \( \text{RP} \leq \text{P/poly} \).

All languages \( L \) that can be solved by

- A non-uniform det. alg \( \text{m} \) in polytime

i.e. a sequence of algs \( \text{A}_1, \text{A}_2, \ldots, \text{A}_n, \ldots \)

- one for each input size \( n \)

Or : equiv. polytime alg \( \text{m} \) that is given on advice

String with poly length

Not relevant to us
or equiv.

- aym n
- string with poly length depending on n.

or equiv: Poly-size circuit

Pr:

By repeating cn times, runtime still poly.

get an algorithm with err prob. \( \leq \frac{1}{2^cn} \)

Fix \( n \).

Pick random sequence \( r \) of \( T(n) \) bits.

For any fixed input \( x \) of size \( n \),

\[ \Pr(\text{alg is wrong on } x \text{ using } r) \leq \frac{1}{2^cn} \]

By union bound,

\[ \Pr(\exists x \text{ of size } n, \text{ alg is wrong on } x \text{ using } r) \leq 2^n \cdot \frac{1}{2^cn} < 1 \text{ for } c > 1 \]

there exists sequence \( r \) s.t.

\( \exists \text{ input } x \text{ of size } n, \text{ alg using } r \text{ is correct on } x \).

this is a nonunif. algorithm.

\( \square \)

**Big Open Problem**

Is \( \text{BPP} = \text{P} \)?

**Known:** Impagliazzo & Wigderson '97 showed

\( \text{BPP} = \text{P} \) if there is a problem in \( \text{E} = \text{DTIME}(2^{O(n)}) \)...it complexity.
\[
\text{BPP} = \text{DTIME}(\leq 2^{\text{polylog}(n)})
\]

("hardness vs. randomness")

\[
\text{NP} \supset \text{BQP} \supset \text{P}
\]

Random Re-Ordering

Example: find min of \(n\) numbers \(S = \{x_1, \ldots, x_n\}\)

Standard incremental alg'nm:

0. \(\text{ans} = \infty\) ------ \(\text{RANDOMly permute } x_1, \ldots, x_n\)
1. for \(i = 1, \ldots, n\)
2. if \(x_i < \text{ans}\)
3. \(\text{ans} = x_i\) \(\text{(*)}\)
4. return \(\text{ans}\)

\(O(n)\) time \(\text{(n-1 comps)}\)

How many changes \(\text{(*)}\)?
Worst-case: \(n\) times
Expected? \(\left(\frac{n}{n-1, \ldots, 1}\right)\)
naively: list all $n!$ permutations, compute $Q$, take avg, ...

rewrite algm backwards:

\[
\begin{align*}
\min(S): & \\
0. \text{ if } S = \emptyset \text{ return } \infty \\
1. \text{ pick } x \in S \text{ randomly} \\
2. \text{ ans } = \min(S - \{x\}) \\
3. \text{ if } x \leq \text{ ans} \\
4. \quad \text{ ans } = x \\
5. \text{ return ans}
\end{align*}
\]

For any fixed $S$,

\[
\Pr \left[ (\ast) \text{ is done} \right] = \Pr \left[ x < \min(S - \{x\}) \right] = \Pr \left[ x = \min(S) \right] = \frac{1}{n}.
\]

\[\implies \text{ expected total # changes} \]

\[
F(n) = F(n-1) + \frac{1}{n} \cdot 1 + \left( -\frac{1}{n} \right) \cdot 0
\]

by linearity of expectation

\[\implies F(n) = \frac{1}{n} + \frac{1}{n-1} + \frac{1}{n-2} + \ldots + \frac{1}{2} + 1 \]

(Harmonic numbers)

\[= \Theta(\log n)\]