We recall the definition of \( BPP \). A language \( L \) is in \( BPP \) if there exists a Turing machine \( M \) running in polynomial time \( p(n) \) on all input \( x \) such that \( |x| = n \) and random tape \( r \) of size \( p(n) \) such that:

- If \( x \in L \), then \( \Pr_{r}[M(x, r) = \top] \geq 2/3 \);
- If \( x \notin L \), then \( \Pr_{r}[M(x, r) = \top] \leq 1/3 \).

We also recall the Chernoff’s bound: Let \( X_1, \ldots, X_N \) be random independent variables with value in \( \{0, 1\} \) with the same law \( \Pr(X_i = 1) = p \), then:

\[
\Pr(X_1 + X_2 + \ldots + X_N \geq (1 + \theta) \cdot p \cdot N) \leq e^{-\frac{\theta^2 \cdot p \cdot N}{3}}
\]

**Exercise 1** (\( RP^* \)). We define \( RP^* \) as the class of all languages \( L \) for which there exists a probabilistic Turing machine \( M \) running in polynomial time, such that:

- If \( x \in L \) then \( \Pr[M(x, r) \text{ reject}] < 1 \)
- If \( x \notin L \) then \( \Pr[M(x, r) \text{ accept}] = 0 \)

Do you recognize this class?

**Solution 1.** This is in fact \( NP \).

- \( RP^* \subseteq NP \): For the same reason than \( RP \subseteq NP \)
- \( NP \subseteq RP^* \): Let us show that \( SAT \in RP^* \). Let \( M \) be the probabilistic Turing machine that, on a formula \( \phi \) with \( p \) free variable, and \( r \) a random tape of bits (of length \( \geq p \)), evaluates \( \phi \) on \( r \). We have that \( M \) runs in polynomial time. In addition, if we denote by \( \text{eval}(\phi) \) the proportion of valuations that satisfy \( \phi \), we have \( \Pr[M(\phi, r) = \top] = \text{eval}(\phi) \) and \( \Pr[M(\phi, r) = \bot] = 1 - \text{eval}(\phi) \). Therefore:
  - If \( \phi \in SAT \), \( \text{eval}(\phi) > 0 \) and we have \( \Pr[M(\phi, r) = \bot] < 1 \).
  - If \( \phi \notin SAT \), \( \text{eval}(\phi) = 0 \) and \( \Pr[M(\phi, r) = \top] = 0 \).

It follows that \( SAT \in RP^* \). As \( RP^* \) is closed under logspace reduction, we have \( NP \subseteq RP^* \).

**Exercise 2** (A quick come back to oracles). Give a lower and upper bound on the following complexity classes: \( NP^{NP} \), \( RP^{RP} \) and \( BPP^{BPP} \).

**Solution 2.** The class \( NP^{NP} \) corresponds to the second level of the polynomial hierarchy. Furthermore:
• **BPP \subseteq BPP^{BPP}**: This is straightforward, since one can ask the oracle the answer.

• **BPP^{BPP} \subseteq BPP**: Let L ∈ BPP^{BPP}. By definition, there exists B ∈ BPP and M a PTM (of execution time lower than a polynomial p) which decides L (with a BPP semantic) by calling the oracle B. We know that for all polynomial q, there exists a probabilistic Turing machine M_{q} running in polynomial time which decides B with a two-sided error lower than 2^{-q(n)}. Consider now the probabilistic TM M' that executes M and simulates all calls to the oracle B by simulating the execution of the TM M_{q}. Note that the complete size of random words we need is polynomial as we make at most p(n) calls to M_{q} which uses polynomial size random tapes. Furthermore, if no mistake is made in all the calls to M_{q}, then M' does not make a mistake with probability at most 1/3 and correctly accepts or rejects inputs belonging or not to L. Hence, for n = |x|, we have \Pr[M'(x, r) \text{ errs}] = \Pr[M'(x, r) \text{ errs } | \text{ the simulation did no mistake } ] + \Pr[M'(x, r) \text{ errs } | \text{ the simulation made a mistake } ] \leq \Pr[M(x, r) \text{ errs}] + \Pr[ \text{ the simulation made a mistake } ] \leq 1/3 + 1 - (1 - 2^{-q(n)})^{p(n)} \leq 1/3 + 2^{-q(n)} \cdot p(n) \leq 2/5 \text{ for } q(n) = p(n) + 4. \text{ Note that } q(n) \text{ can be chosen as a function of } p(n) \text{ since } p \text{ is given by the Turing machine } M. \text{ Therefore } L \in BPP.

Then, we have RP \cup \text{coRP} \subseteq \text{RP}^{BPP} \subseteq BPP.

**Exercise 3 (NP, RP and BPP).** Show that if NP \subseteq BPP then NP = RP.

**Solution 3.** In any case, we have RP \subseteq NP. Let’s now assume that NP \subseteq BPP. So SAT ∈ BPP. We know that, for all polynomial q, we have M a probabilistic Turing machine running in polynomial time which recognizes SAT, with an error lower than or equal to 2^{-q(n)}. We will define the M' a PTM which works as the following pseudocode (with \phi a formulae with p free variables; r, r' random words):

\[
\psi = \phi \\
\text{For } (i < p) :
\begin{align*}
\text{If } (M(\psi[x_i = \top], r_i) = \top) & : \\
\psi := \psi[x_i = \top] \\
\quad \text{Else } M(\psi[x_i = \bot], r'_i) = \top : \\
\psi := \psi[x_i = \bot] \\
\text{Else :}
\end{align*}
\text{Return } \bot
\]

Notice that \( p < |\phi| \). There is a at most 2p calls to M. Hence, the running time of this algorithm is polynomial and total length of random word used is also polynomial. Therefore, for \( \phi \) a formulae with p free variables, \( |\phi| = n \) and \( x = 2^{-q(n)} \):

- if \( \phi \notin L \) then \( \Pr[M'(\phi, r) = \top] = 0 \) (since we check that the last \( \psi \) is satisfied, which implies that the valuation chosen satisfies \( \phi \)).

- if \( \phi \in L \) then \( \Pr[M'(\phi, r) = \bot] \leq \sum_{i=0}^{2p-1} (1 - x)^i x \) (it’s the probability that one simulation of M fails). That is, \( \Pr[M'(\phi, r) = \bot] \leq \sum_{i=0}^{2p-1} x = 2p \cdot x = 2n \cdot 2^{-q(n)} \leq 2^{2n-q(n)} \).

So, with \( q(n) = 2n + 1 \) : if \( \phi \in L \) then \( \Pr[M'(\phi, r) = \bot] \leq \frac{1}{2} \)

Then: SAT ∈ RP.
Exercise 4 (Logarithmic space BPP). Define BPL as the class of languages decided by a probabilistic Turing machine running in logarithmic space and polynomial time (with a BPP-like semantic). Show that BPL ⊆ P.

Solution 4. Consider a language $L \in \text{BPL}$ and the corresponding PTM running time $p$. Consider the polynomial size (since the space used is logarithmic) configuration graph. It can be seen as a Markov chain where the probability transition are either 0,1 or $1/2$ (when a random bit is read). One can then compute the probability to reach $p_{\text{acc}}$ from $q_0$ in at most $p(n)$ for an input of size $n$. This can be done in polynomial time (by an iteration of matrix multiplication). Then, we accept iff the computed probability is at least $2/3$.

Exercise 5 (The PP class). The class PP is the class of languages $L$ for which there exists a polynomial time probabilistic Turing machine $M$ such that:

1. If $x \in L$ then $\Pr[M(x,r) \text{ accepts}] > 1/2$
2. If $x \notin L$ then $\Pr[M(x,r) \text{ accepts}] \leq 1/2$

1. Show that BPP ⊆ PP and NP ⊆ PP;
2. Exhibit a PP-complete language;
3. Show that PP = PP< and that PP is closed under complement;
4. Consider the decision problem MAJSAT:
   a. Input: a boolean formula $\phi$ on $n$ variables
   b. Output: the (strict) majority of the $2^n$ valuations satisfy $\phi$.

Show that MAJSAT ∈ PP. In fact, MAJSAT is PP-complete.

One may also consider the decision problem MAXSAT:

a. Input: a boolean formula $\phi$ on $n$ variables, a number $K$

b. Output: more than $K$ valuations satisfy $\phi$.

Show that MAXSAT is also PP-complete (to prove that MAXSAT ∈ PP one may reduce MAXSAT to MAJSAT).

Solution 5. 1. A language $L \in \text{BPP}$ is recognized by a PTM $M$ such that if $x \in L$ then $\Pr[M(x,r) \text{ accepts}] \geq 2/3$ and if $x \notin L$ then $\Pr[M(x,r) \text{ accepts}] \leq 1/3$. It follows that $L \in \text{PP}$.

The class PP is closed under logspace reduction. It suffice to show that SAT ∈ PP. Consider now a probabilistic Turing machine with an input that is a formula $\phi$. According to the first bit of the random tape, it either accepts or reads what remains of the random tape for a valuation and accepts if and only if it satisfies $\phi$. Then, if $\phi \in \text{SAT}$, we have $\Pr[M(x,r) \text{ accepts}] > 1/2$, otherwise $\Pr[M(x,r) \text{ accepts}] = 1/2$.

2. The language $L = \{(M,x,t) \mid \text{ the PTM } M \text{ accepts } x \text{ in time } t \text{ with proba } > 1/2\}$ is PP-complete.

3. Let us define PP< as the class of languages $L$ for which there exists a polynomial time probabilistic Turing machine $M$ such that:
• if \( x \in L \) then \( \Pr[M(x, r) \text{ accepts}] > \frac{1}{2} \)
• if \( x \notin L \) then \( \Pr[M(x, r) \text{ accepts}] < \frac{1}{2} \)

Trivially, we have \( \text{PP}_< \subseteq \text{PP} \). Now, consider \( L \in \text{PP} \) and its associated Turing machine \( M \) running in polynomial time \( p \). Without loss of generality, we assume that the alphabet of the random tape is of size 2, hence the probability of a random word for \( M \) on an input \( x \) such that \( |x| = n \) is \( 2^{-p(n)} \). Therefore, if \( x \in L \) then \( \Pr[M(x, r) \text{ accepts}] \geq \frac{1}{2} + \frac{1}{2^{2p(n)}} \). Now, we construct another Turing machine \( M' \) that runs \( M \) on an input. If \( M \) would reject, \( M' \) rejects too, and if \( M \) would accept then \( M' \) rejects with probability \( \frac{1}{2^{p(n)}} \) (for instance, by reading a word in the random tape of length \( p(n) \) and accepting only if there are only 0s). Then:

• if \( x \in L \): \( \Pr[M(x, r) \text{ accepts}] \geq (\frac{1}{2} + \frac{1}{2^{2p(n)}}) \cdot (1 - \frac{1}{2^{p(n)}}) = \frac{1}{2} + \frac{1}{2^{2p(n)}} - \frac{1}{2^{2p(n)}} > \frac{1}{2} \)
• if \( x \notin L \): \( \Pr[M(x, r) \text{ accepts}] \leq \frac{1}{2} \cdot (1 - \frac{1}{2^{p(n)}}) < \frac{1}{2} \)

That is, \( L \in \text{PP}_< \). The stability under complement then follows by inverting the accepting and rejecting states.