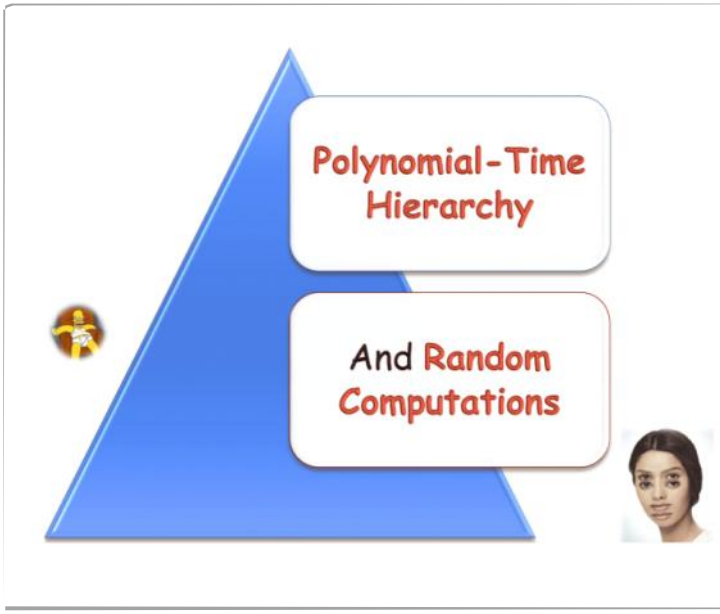
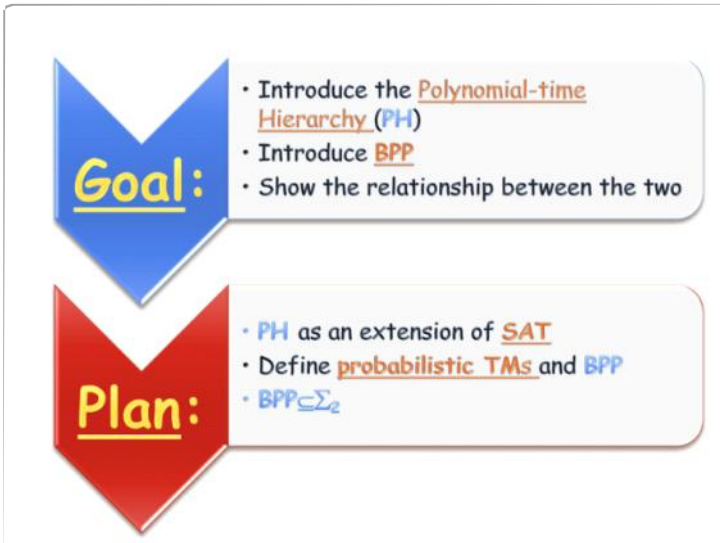


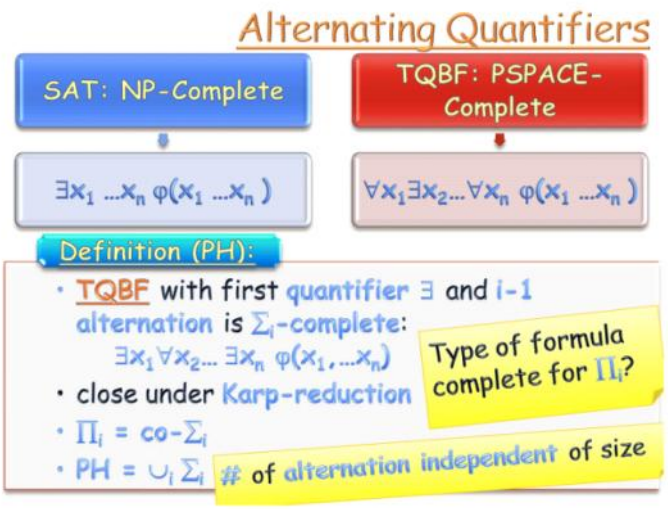
Let us now discuss two important notions regarding two aspects of computing, and furthermore show an interesting connection between the two.

Randomized Computation



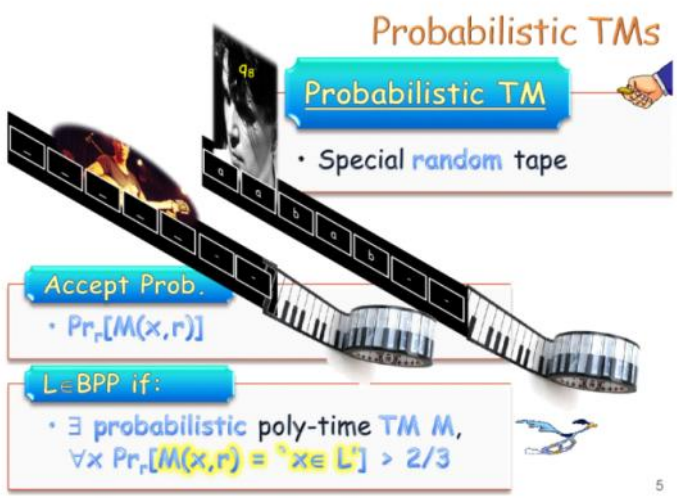
The first, fundamental aspect is Random Computation, where one allows the use of random bits for the computation, while willing to allow some small probability of error, or for the running time to be only an expectation. In particular the Complexity class BPP. The other basic notion is the extension of the P, NP, coNP framework to form a hierarchy of complexity classes --- the Polynomial-Time Hierarchy. We then show that BPP is in fact contained in the polynomial-time hierarchy.

Probablistic Turing Machines

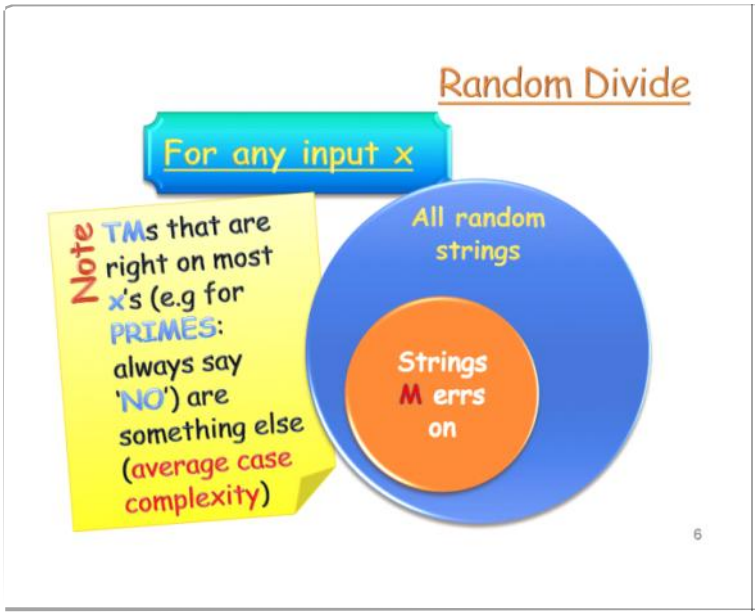


Consider a QBF (Quantified Boolean Formula): we've already proved TQBF is PCPACE-complete. We could also have formulated SAT as a special case of TQBF, where only existential quantifiers are aloud.
Now, what if we look at some less restrictive forms of QBFs?
Let us count the number of times the quantifier used in the formula changes between existential and universal, and add 1 to it --- in other words, count how many blocks of recurring quantifiers there are in the formula.
Some Languages can be Karp-reduced to such a formula with i blocks of quantifiers which also starts with an existential quantifier --- let the class of these languages be denoted Si.
Note that this is an alternative, still-legitimate manner by which to define a class of languages: usually we define a class and then find a problem complete for it; this time we define a language and then define the class it is complete for.
We can then define Pi as the class of all problems whose complement is in Si, for which the language of TRUE formulas with i block of quantifiers that begin with a universal quantifier is complete.
The Polynomial-time Hierarchy comprise all those classes for some i.
Note, however, that these are still not general TQBF formulas as the number of blocks cannot grow with the input size, hence PH is not necessarily the same as PSPACE.

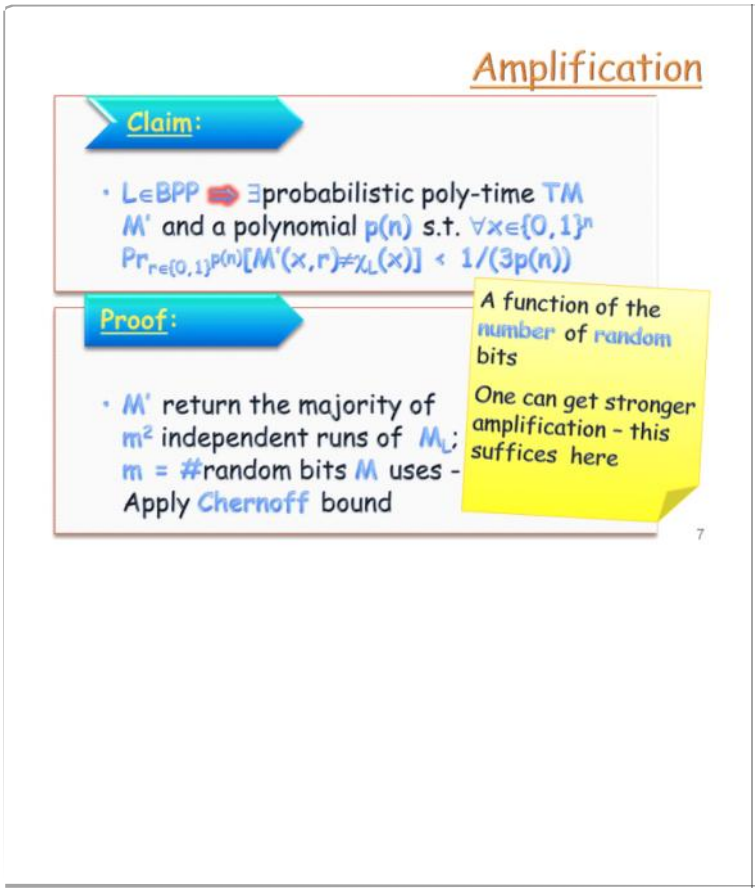
Let us now note some simple facts regarding the PH:
The first two levels of the hierarchy are the classes NP and coNP.
As necessary so as to refer to it as a hierarchy, the i'th level is contained in both classes in the next i+1'st level.
The entire hierarchy is contained in PSPACE.
And lastly, if the two classes of the hierarchy in some level turn out to be the same, then the hierarchy collapses for that level.
One proves that by induction: take the formula you get by fixing (in any manner legal) the variable associated with the first blocks of quantifiers, leaving the last i block intact; this formula can be replaced by a formula of the complement class. After that transformation, the formula has one less block of quantifiers.



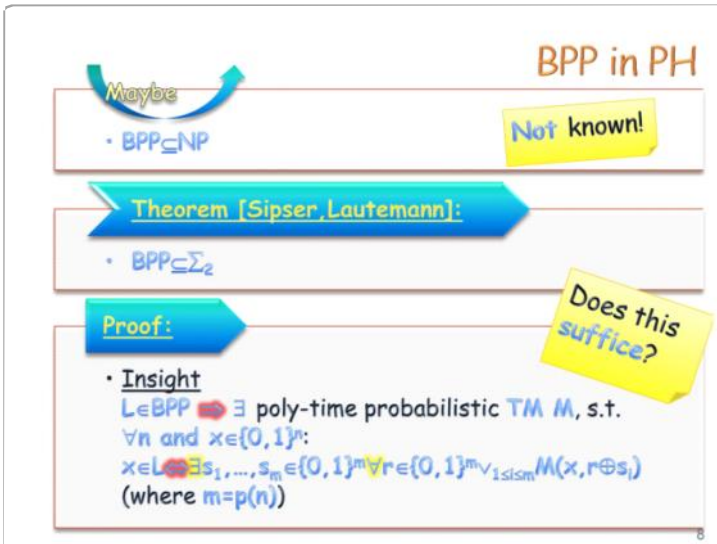
A Probabilistic or Random TM is one that uses an additional tape which is referred to as the Random tape.
For any given input x one may consider all possible random strings r that can be written to that tape, and look at all possible executions of the TM on x and r. (If M runs in polynomial time there is an upper-bound on the number of bits that can be read from r, hence one need not consider longer strings, which in turns allow us to consider uniform probability over all possible strings r).
The class BPP comprise all languages that have a polynomial-time TM that on ALL inputs x, accept x with probability >2/3 if $x \in L$, and accepts with probability <1/3 in case $x \notin L$.



A BPP TM for a language L returns on a definitive majority of the random strings the correct accept/reject answer. Nevertheless, it may err on a small fraction of those.
Note that we are still discussing worst-case complexity. Were we interested in average case complexity –namely, where the algorithm returns the correct answer on most of the inputs– some problems may become easier. A BPP TM must answer w.h.p. the correct answer on ALL inputs.

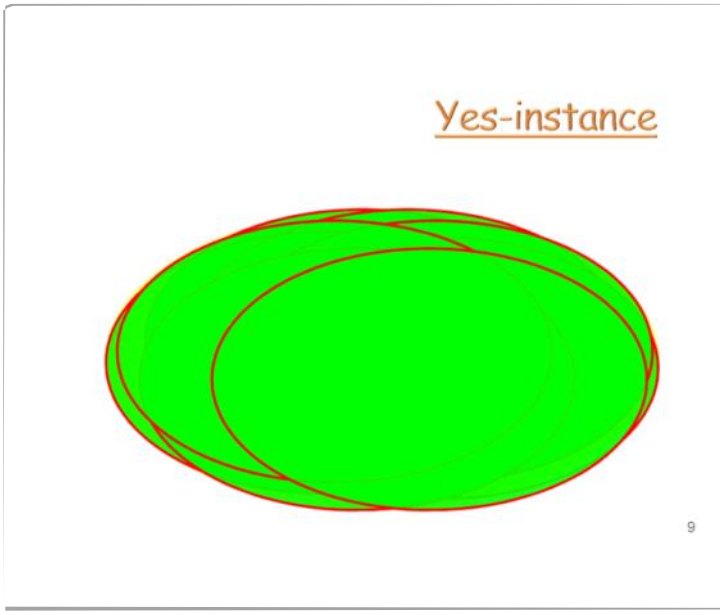


One can AMPLIFY a BPP TM to err with a very small probability, in particular exponentially small. To do that, a TM M' runs M many times on independently selected random strings and returns the majority of the answers returned by these runs. Apply Chernoff bound to see that the probability of error becomes exponentially small in the number of repetitions.
For the purpose of the next theorem we prove, we are interested in slightly different parameters, and would like to get the probability of error small in terms of the number of bits the TM uses.
It is not hard to see that, starting with a TM that uses m random bits, applying the above repetition technique m polylog(m) times ensures the probability of error is less than $1/3^{m'}$ where m' is the number of bits M' uses (m^2 polylog(m)).

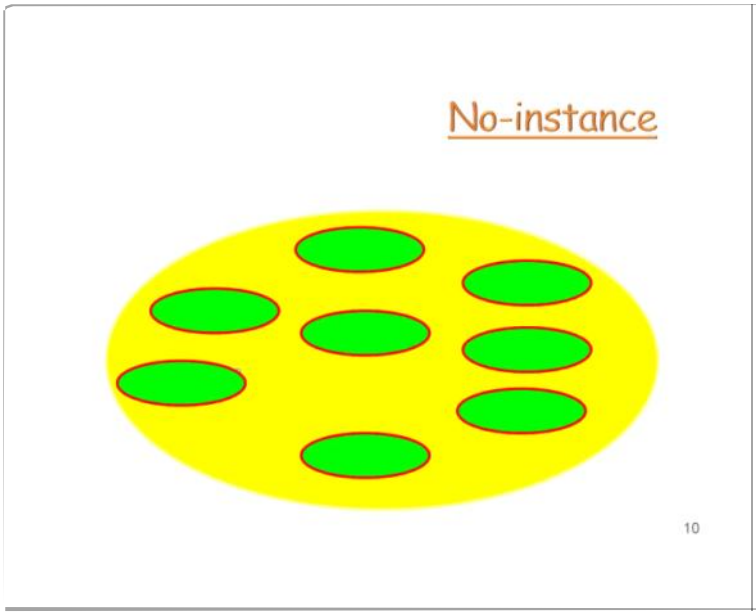


Now, does randomness really help in time-bounded computations?
It is quite possible that P=BPP but no one can prove that so far.
It is again possible BPP is contained in NP, but that's also not proved.
What we can prove is that BPP is in fact in PH, in particular in S2.
The proof is by a reduction: given a language L in BPP there is a TM M whose error on any input is limited to 1/3m where m is the random bits M uses (we've just shown that); per M and x construct a formula that is true if and only if there are m strings S1, .., Sm so that for every string r, applying M on r XORed with one of the Si's makes M accept (one is enough).
This formula is clearly in S2 – we now need to show that formula is true if and only if x∈L.

BPP

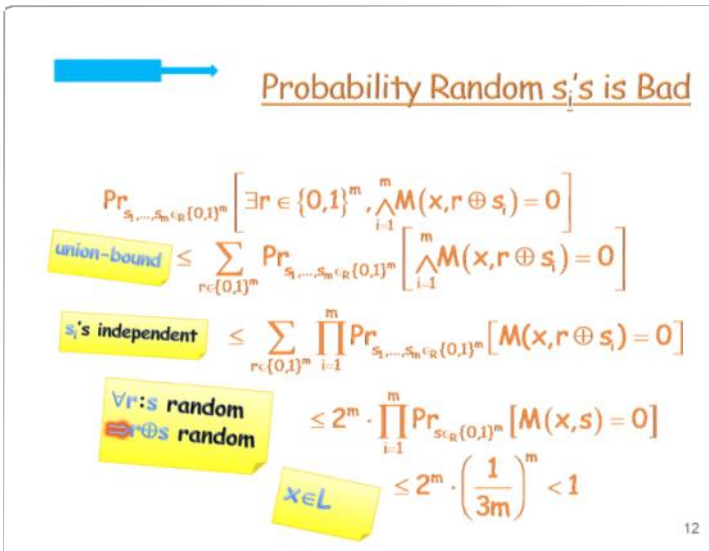


Intuitively, we need to show that in case x∈L (M rejects w.p. <1/3m) there are S1, .., Sm so that every r has at least one of them XOR it to become accepting.

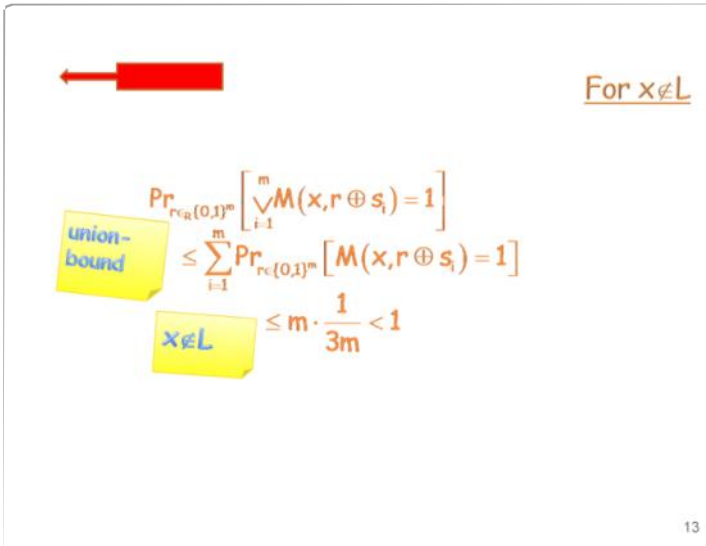


On the other hand, in case $x \notin L$, as M accepts w.p. <1/3m, XORing r with m distinct strings, can enlarge the set of accepting strings by at most a factor m, which would still imply that at most 1/3 of the strings r are good, namely, not all are good and the formula is FALSE.

The proof is as follows:
First, assume M that errs on L w.p. <1/3m where m is the number of random bits it uses (we just proved one can assume that).
Now, apply the probabilistic method, which would allow us to conclude that in case $x \in L$ there are S1,..,Sm that cause all random strings r to be accepting XORed with one of them.
The probabilistic method proves there exists some structure satisfying a given property, by showing the probability of a structure chosen randomly, according to some distribution, to satisfy the property is positive.
Let us then prove completeness next.
Soundness follows by a simple application of the union-bound (the probability of a union of events is bounded from above by the sum of the events` probabilities). A formal proof would follow.



Completeness:
The probability S1,..,Sm is bad (namely there exists an r that stays rejecting even if XORed with all Si's) is bounded from above
by the sum over all r's of the probability r is bad, which in turn is bounded from above
by the sum over all r's of the product of the probability for each Si (r XORed with a random Si are independent events), which can be limited from above
By the number of r's, times the m'th power of the probability for a random string s to be bad (the probability of all independent events to hold is the product of their probability)
which in fact tends to 0, but is certainly smaller than 1.
To conclude, the probability of S1,..,Sm to be good is very high and certainly positive.



Soundness:
Follows by a simple application of the union bound.

Q.E.D!

It follows that:

- $L \in BPP \Rightarrow$ there's a poly. prob. TM M, s.t for any x there is $m=poly(|x|)$ s.t $x \in L \Leftrightarrow \exists s_1,\ldots,s_m \; \forall r \vee_{1 \le i \le m} M(x, r \oplus s_i)=1$

Hence

- $L \in \Sigma_2 \Rightarrow BPP \subseteq \Sigma_2$ □

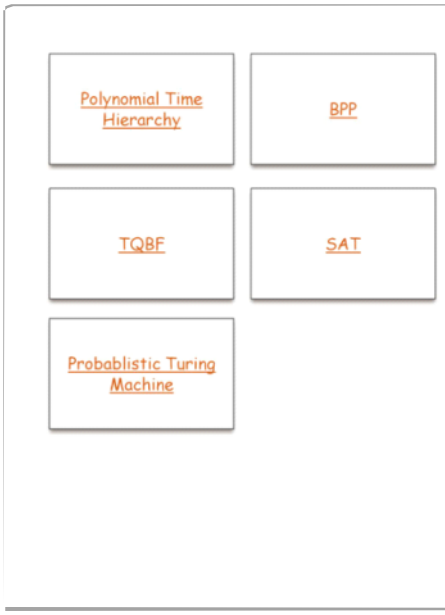
14

In summary, any L in BPP can be reduced to the above formula, which is in S2.
Q.E.D.

- the polynomial-time hierarchy
- Saw $NP \subseteq PH \subseteq PSPACE$
- $NP=coNP \Rightarrow PH=NP$ ("the hierarchy collapses")

- probabilistic TMs
  - Defined the complexity class BPP
  - How to amplify randomized computations
  - We proved $P \subseteq BPP \subseteq \Sigma_2$

15



- Polynomial Time Hierarchy
- BPP
- TQBF
- SAT
- Probablistic Turing Machine