Randomized Complexity Classes

- How random bits, or coin flips, are useful in computation
- Randomized complexity classes: BPP, RP, co-RP

Polynomial Identity Testing

Given two multivariate polynomials P and Q, does P = Q?
Equivalently, is the polynomial P - Q identically 0?

$$\begin{vmatrix} 1 & 1 & 1 & \cdots & 1 \\ x_1 & x_2 & x_3 & \cdots & x_n \\ x_1^2 & x_2^2 & x_3^2 & \cdots & x_n^2 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ x_1^{n-1} & x_2^{n-1} & x_3^{n-1} & \cdots & x_n^{n-1} \end{vmatrix} = \prod_{i>j} (x_i - x_j)$$

Taken from Hastad's notes

Polynomial Identity Testing

Given two multivariate polynomials P and Q, does P = Q?
Equivalently, is the polynomial P - Q identically 0?

We'll assume that our polynomials are over the integers, but the algorithm we develop can be adapted to work also for polynomials over any field

To formulate this problem precisely, let's consider how multivariate polynomials can be represented

Polynomial Identity Testing

Representing multivariate polynomials

Polynomials may be represented implicitly, e.g. using arithmetic expressions or as “black boxes”

Example:

$p(x_1, x_2, x_3)$

$= (x_2 - x_1)(x_3 - x_1)(x_3 - x_2)$

Polynomial Identity Testing

Representing multivariate polynomials

Polynomials may be represented implicitly, e.g. using arithmetic expressions or as “black boxes”

Example:

$p(x_1, x_2, x_3)$

$= (x_2 - x_1)(x_3 - x_1)(x_3 - x_2)$

$= x_2x_3^2 - x_2^2x_3 - x_1x_2x_3 + x_1x_2^2 - x_1x_3^2 + x_1x_2x_3 + x_1^2x_3 - x_1^2x_2$

Polynomial Identity Testing

Representing multivariate polynomials

Polynomials may be represented implicitly, e.g. using arithmetic expressions or as “black boxes”

Example:

$p(x_1, x_2, x_3)$

$= (x_2 - x_1)(x_3 - x_1)(x_3 - x_2)$

$= x_2x_3^2 - x_2^2x_3 - x_1x_2x_3 + x_1x_2^2 - x_1x_3^2 + x_1x_2x_3 + x_1^2x_3 - x_1^2x_2$

The second line represents p implicitly, while the third represents p explicitly as a sum of distinct *monomials*

Polynomial Identity Testing

Representing multivariate polynomials

Polynomials may be represented implicitly, e.g. using arithmetic expressions or as “black boxes”

Example:

$p(x_1, x_2, x_3)$

$= (x_2 - x_1)(x_3 - x_1)(x_3 - x_2)$

$= x_2x_3^2 - x_2^2x_3 - x_1x_2x_3 + x_1x_2^2 - x_1x_3^2 + x_1x_2x_3 + x_1^2x_3 - x_1^2x_2$

The second line represents p implicitly, while the third represents p explicitly as a sum of distinct *monomials*

Polynomial Identity Testing

Representing multivariate polynomials

Polynomials may be represented implicitly, e.g. using arithmetic expressions or as “black boxes”

Example:

$p(x_1, x_2, x_3)$

$= (x_2 - x_1)(x_3 - x_1)(x_3 - x_2)$

$= x_2x_3^2 - x_2^2x_3 - x_1x_2x_3 + x_1x_2^2 - x_1x_3^2 + x_1x_2x_3 + x_1^2x_3 - x_1^2x_2$

Each monomial has a *coefficient* and a *degree*

Polynomial Identity Testing

Representing multivariate polynomials

Polynomials may be represented implicitly, e.g. using arithmetic expressions or as “black boxes”

Example:

$p(x_1, x_2, x_3)$

$= (x_2 - x_1)(x_3 - x_1)(x_3 - x_2)$

$= x_2x_3^2 - x_2^2x_3 - x_1x_2x_3 + x_1x_2^2 - x_1x_3^2 + x_1x_2x_3 + x_1^2x_3 - x_1^2x_2$

Coefficient: -1
Degree 3

Each monomial has a *coefficient* and a *degree*

Polynomial Identity Testing

Representing multivariate polynomials

Polynomials may be represented implicitly, e.g. using arithmetic expressions or as “black boxes”

Example:

$p(x_1, x_2, x_3)$

$= (x_2 - x_1)(x_3 - x_1)(x_3 - x_2)$

$= x_2x_3^2 - x_2^2x_3 - x_1x_2x_3 + x_1x_2^2 - x_1x_3^2 + x_1x_2x_3 + x_1^2x_3 - x_1^2x_2$

A polynomial is *identically 0* if all monomials have coefficient equal to 0

Polynomial Identity Testing

Representing multivariate polynomials

Polynomials may be represented implicitly, e.g. using arithmetic expressions or as “black boxes”

Example:

$p(x_1, x_2, x_3)$

$= (x_2 - x_1)(x_3 - x_1)(x_3 - x_2)$

$= x_2x_3^2 - x_2^2x_3 - x_1x_2x_3 + x_1x_2^2 - x_1x_3^2 + x_1x_2x_3 + x_1^2x_3 - x_1^2x_2$

How many monomials can there be in a polynomial with n variables and degree at most d? (Rough guess?)

Polynomial Identity Testing

Representing multivariate polynomials

A “black box” example

$$\begin{vmatrix} 1 & 1 & 1 & \cdots & 1 \\ x_1 & x_2 & x_3 & \cdots & x_n \\ x_1^2 & x_2^2 & x_3^2 & \cdots & x_n^2 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ x_1^{n-1} & x_2^{n-1} & x_3^{n-1} & \cdots & x_n^{n-1} \end{vmatrix} = \prod_{i>j} (x_i - x_j)$$

Taken from Hastad’s notes

Polynomial Identity Testing

Representing multivariate polynomials

A “black box” example

$$\begin{vmatrix} 1 & 1 & 1 & \cdots & 1 \\ x_1 & x_2 & x_3 & \cdots & x_n \\ x_1^2 & x_2^2 & x_3^2 & \cdots & x_n^2 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ x_1^{n-1} & x_2^{n-1} & x_3^{n-1} & \cdots & x_n^{n-1} \end{vmatrix} = \prod_{i>j} (x_i - x_j)$$

This determinant is a multivariate polynomial over n variables, with degree $O(n^2)$, since

$$\det A = \sum_{\text{permutations } \pi \text{ of } 1..n} \operatorname{sign}(\pi) \prod_{1 \le u \le n} A_{u,\pi(u)}$$

(Leibniz formula)

Polynomial Identity Testing

Representing multivariate polynomials

A “black box” example

$$\begin{vmatrix} 1 & 1 & 1 & \cdots & 1 \\ x_1 & x_2 & x_3 & \cdots & x_n \\ x_1^2 & x_2^2 & x_3^2 & \cdots & x_n^2 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ x_1^{n-1} & x_2^{n-1} & x_3^{n-1} & \cdots & x_n^{n-1} \end{vmatrix} = \prod_{i>j} (x_i - x_j)$$

There are efficient algorithms (“black boxes”) to compute the determinant for fixed values of the variables

Polynomial Identity Testing (PIT)

Is a multivariate polynomial p identically 0?

$$\begin{vmatrix} 1 & 1 & 1 & \cdots & 1 \\ x_1 & x_2 & x_3 & \cdots & x_n \\ x_1^2 & x_2^2 & x_3^2 & \cdots & x_n^2 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ x_1^{n-1} & x_2^{n-1} & x_3^{n-1} & \cdots & x_n^{n-1} \end{vmatrix} = \prod_{i>j}(x_i - x_j) \; ?$$

Polynomial Identity Testing (PIT)

Is a multivariate polynomial p identically 0?

- p may be represented implicitly, e.g. using arithmetic expressions or as “black boxes”
- p has *n* variables and degree *d*
- p can be evaluated in time polynomial in *n* and *d* for any fixed values of the variables
- An explicit representation pf p may have size exponential in *n*, since it may have $\binom{n+d}{d}$ monomials

Polynomial Identity Testing (PIT)

Is a multivariate polynomial p identically 0?

PIT is an interesting problem for several reasons:

- Algorithms for PIT can be applied to solve other problems, such as determining whether a bipartite graph has a perfect matching
- PIT has an efficient (i.e., poly-time) randomized algorithm *but no known efficient deterministic algorithm*
- If an efficient deterministic algorithm is found for PIT, there are other very interesting consequences in complexity theory (circuit lower bounds)

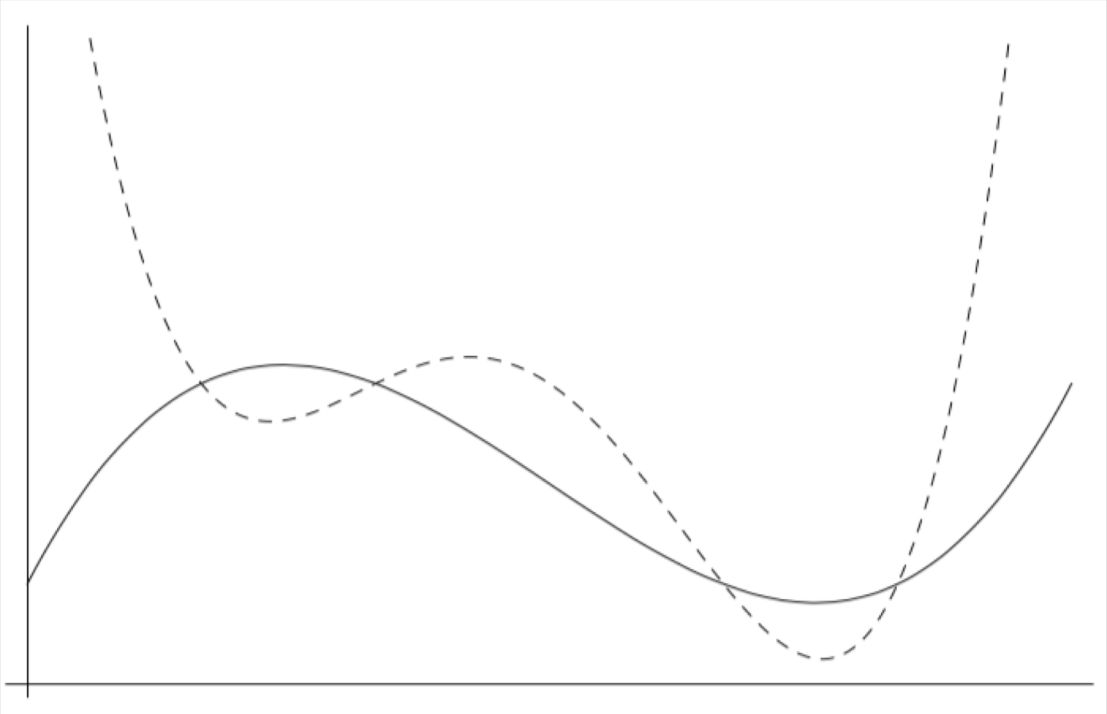
Polynomial Identity Testing (PIT)

Is a multivariate polynomial p identically 0?

Polynomial Identity Testing (PIT)

Is a multivariate polynomial p identically 0?

Univariate case: A univariate polynomial of degree d that is not identically 0 has at most d zero's



From Moore

Polynomial Identity Testing (PIT)

Is a multivariate polynomial p identically 0?

PIT-Alg(p) // p has n variables and degree ≤ d

Let R = $2nd$ // larger R reduces the error

Choose $\boldsymbol{z} = (z_1, z_2, \ldots, z_n)$ randomly and uniformly in the range $[1, R]^n$ // each z_i *is a positive integer*

If p(**z**) ≠ 0 Return false

Else Return true

Polynomial Identity Testing (PIT)

Correctness analysis of PIT-Alg

Claim: Let p be an input to PIT-Alg

a) If p is identically 0 then Pr[PIT-Alg(p) is true] = 1

b) If p is not identically 0 then

$$\Pr[\text{PIT-Alg}(p) \text{ is true}] \leq dn/R \leq \tfrac{1}{2}$$

The proof uses the following claim:

Claim (Schwartz-Zippel): Let p be a polynomial that is not identically 0, with n variables and degree $\leq d$. The set

$$Z = \{ (\boldsymbol{z} \mid 1 \leq z_i \leq R, 1 \leq i \leq n) \wedge (p(\boldsymbol{z}) = 0)\}$$

has size at most dnR^{n-1}.

Polynomial Identity Testing (PIT)

Correctness analysis of PIT-Alg

Claim: Let p be an input to PIT-Alg

a) If p is identically 0 then Pr[PIT-Alg(p) is true] = 1

b) If p is not identically 0 then

$$\Pr[\text{PIT-Alg}(p) \text{ is true}] \leq dn/R \leq \tfrac{1}{2}$$

Polynomial Identity Testing (PIT)

Correctness analysis of PIT-Alg

Claim: Let p be an input to PIT-Alg

a) If p is identically 0 then Pr[PIT-Alg(p) is true] = 1

b) If p is not identically 0 then

$$\Pr[\text{PIT-Alg}(p) \text{ is true}] \leq dn/R \leq \tfrac{1}{2}$$

We can reduce the error by increasing R, or by repeating the algorithm.

For example, choose $R = dn^{c+1}$ to get error $1/n^c$

Summary

- Finding a deterministic polynomial-time algorithm for PIT would be a major advance (but not easy since it would also answer hard questions about circuit lower bounds)

- One approach: Derandomize PIT-Alg by efficiently constructing a set of points $\{z(1),...,z(k)\}$ such that for all degree-d polynomials p in n variables that are not identically 0, there exists an $i \in [1..k]$ with $p(z(i)) \neq 0$

Probabilistic Turing Machines

- A Probabilistic TM (PTM) has both deterministic and *coin-flipping* states. In such states there are transitions to two new states that encode two outcomes of a coin flip: either 0 (heads) or 1 (tails). Each state is reached with probability 1/2.

- The possible executions of a PTM on an input can be represented as a tree of configurations. The probability of reaching a leaf of the tree is $1/2^k$, where k is the number of coin flipping states on the path from the root to the leaf

BPP: Bounded Error Poly Time

- PTM M decides L in time t(n) if on all inputs x, M halts in t(|x|) steps regardless of its random choices, and
  - x is in L $\Rightarrow$ Pr[M accepts x] $\geq 2/3$
  - x is not in L $\Rightarrow$ Pr[M accepts x] $\leq 1/3$
- We say that L is in BPTIME(t(n))

- BPP = $\bigcup_c$ BPTIME(n^c).

RP, co-RP: One-sided Error Poly Time

- PTM M decides L in time t(n) if on all inputs x, M halts in t(|x|) steps regardless of its random choices, and

  x is in L $\Rightarrow$ Pr[M accepts x] $\geq 2/3$

  x is not in L $\Rightarrow$ Pr[M accepts x] = 0
- We say that L is in RTIME(t(n))

- RP = $\bigcup_c$ RTIME(n^c)
- co-RP is the set of languages whose complement is in RP

Reducing Error

- Error of 1/3 in the definitions isn't great. Can we reduce the error? How?

Reducing Error

- Error of 1/3 in the definitions isn't great. Can we reduce the error? How?
- Let L be in BPP. There is a polynomial-time PTM M′ such that for every input x,

  x is in L $\Rightarrow$ Pr[M accepts x] $\geq 1 - 2^{-|x|-2}$

  x is not in L $\Rightarrow$ Pr[M accepts x] $\leq 2^{-|x|-2}$

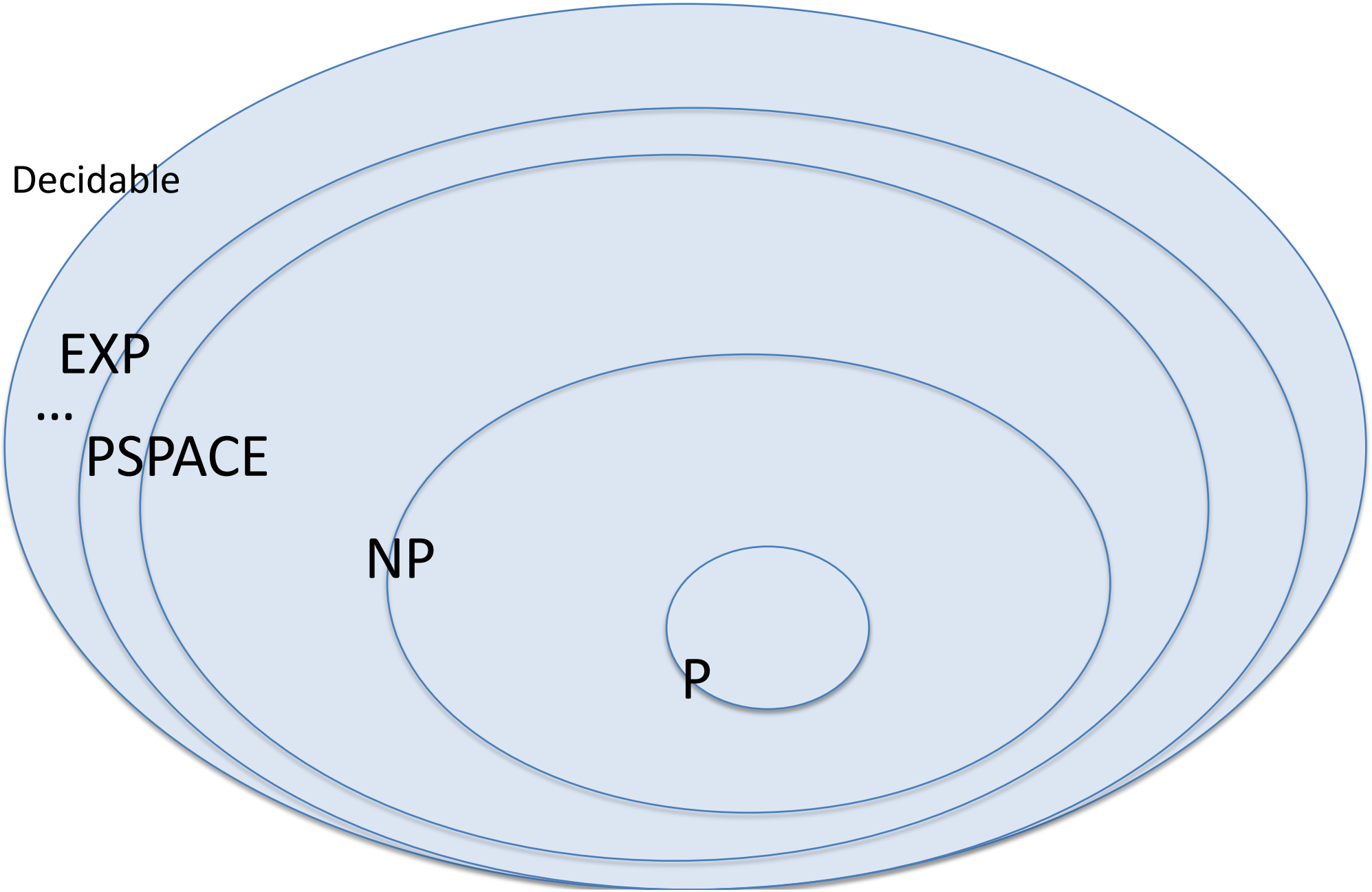
- The error can be further reduced if desired

Reducing Error to $2^{-|x|-2}$

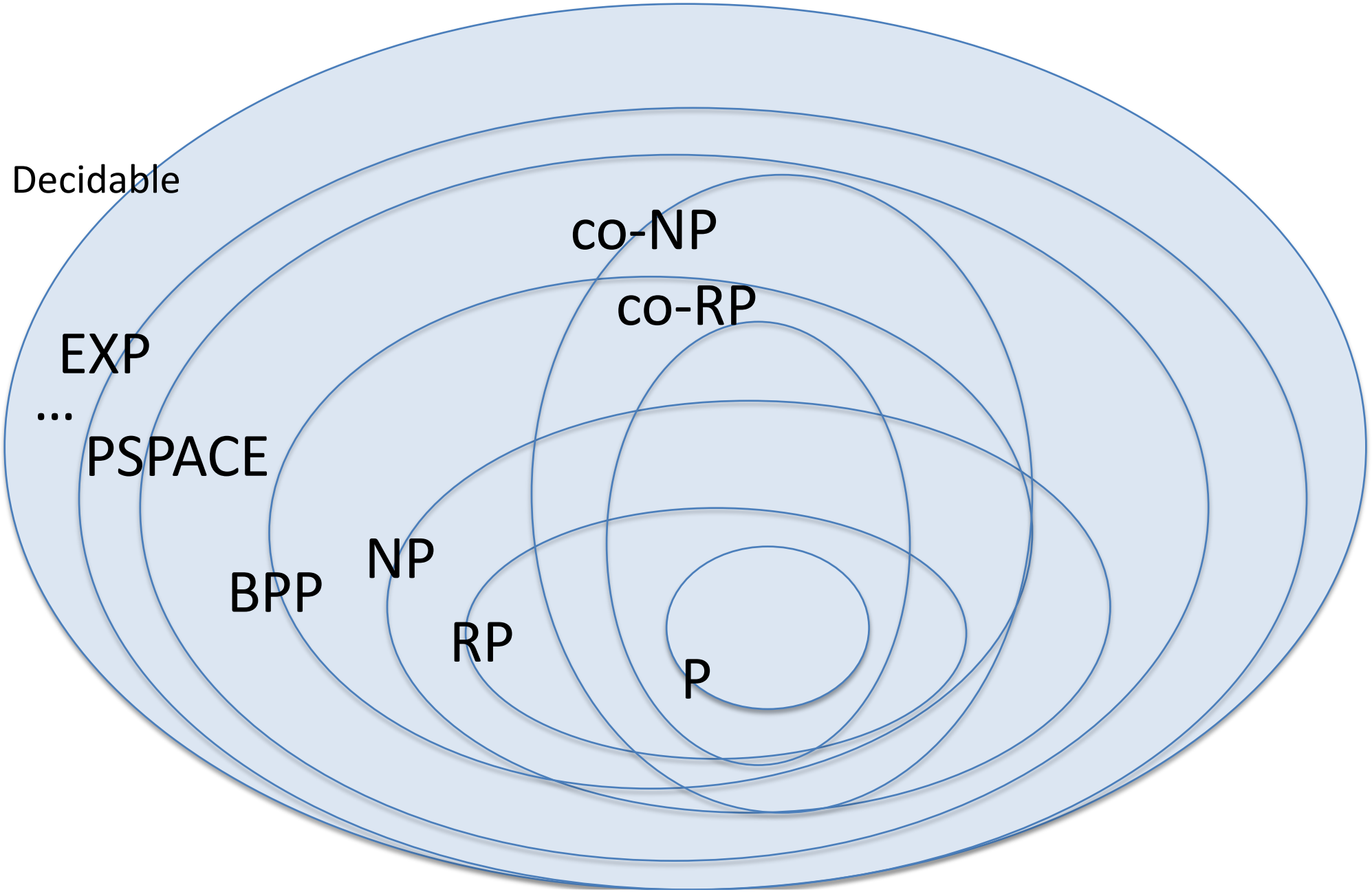
- Idea: Let M be a TM with error 1/3. Run M $\Theta(n)$ times, and take the majority outcome
- Analysis: apply a simple variant of Chernoff's Bound: Let $0 < p < 1/2$, $q=1-p$ and let k be an even integer. Then

$$\sum_{i=k/2}^{k} \binom{k}{i} p^i q^{k-i} \leq (4pq)^{k/2}.$$

Add BPP, RP, co-RP to the picture…



Add BPP, RP, co-RP to the picture…



Next Class

- BPP is in P/poly
- BPP and the polynomial time hierarchy

- Reading for next class: Arora-Barak 5.1, 6.2