Probabilistic Computation

Lecture 14
BPP, ZPP
Zoo
Zoo

EXP \rightarrow \text{NEXP} \rightarrow \text{NP} \rightarrow \Sigma_2^P \rightarrow \text{NPSPACE} \rightarrow \text{PSPACE}

P \rightarrow \text{RP} \rightarrow \text{NP} \rightarrow \Sigma_2^P \rightarrow \text{NPSPACE} \rightarrow \text{PSPACE}

\text{L} \rightarrow \text{NL} \rightarrow \text{NPSPACE} \rightarrow \text{PSPACE}

\text{L} \rightarrow \text{NL} \rightarrow \text{NPSPACE} \rightarrow \text{PSPACE}

\text{L} \rightarrow \text{NL} \rightarrow \text{NPSPACE} \rightarrow \text{PSPACE}
Zoo
Zoo
Zoo
Some Probabilistic Algorithmic Concepts
Some Probabilistic Algorithmic Concepts

- Sampling to determine some probability
- Checking if determinant of a symbolic matrix is zero: Substitute random values for the variables and evaluate
- Polynomial Identity Testing: polynomial given as an arithmetic circuit. Like above, but values can be too large. So work over a random modulus.
- Random Walks (for sampling)
- Monte Carlo algorithms for calculations
- Reachability tests
Random Walks
Random Walks

- Which nodes does the walk touch and with what probability?
- How do these probabilities vary with number of steps?

Analyzing a random walk

- Probability Vector: \( p \)
- Transition probability matrix: \( M \)
- One step of the walk: \( p' = Mp \)
- After \( t \) steps: \( p^{(t)} = M^t p \)
Space-Bounded Probabilistic Computation
Space-Bounded Probabilistic Computation

- PL, RL, BPL
- Logspace analogues of PP, RP, BPP
- Note: $RL \subseteq NL$, $RL \subseteq BPL$
- Recall $NL \subseteq P$ (because PATH $\in P$)
- So $RL \subseteq P$
- In fact $BPL \subseteq P$
\[ \text{BPL} \subseteq \text{P} \]
BPL $\subseteq P$

- Consider the BPL algorithm, on input $x$, as a random walk over states
  - Construct the transition matrix $M$
    - Size of graph is $\text{poly}(n)$, probability values are 0, 0.5 and 1
  - Calculate $M^t$ for $t = \text{max running time} = \text{poly}(n)$
  - Accept if $(M^t p_{\text{start}})_{\text{accept}} > 2/3$
Zoo
Zoo

EXP → NEXP → NPSPACE → PSPACE
P → NP → Σ²P → NPSPACE
P → RP
P → NL
P → RL
P → L
Zoo
Zoo
Zoo
Zoo
Expected Running Time
Expected Running Time

- Running time is a random variable too
  - As is the outcome of yes/no
- May ask for running time to be polynomial only in expectation, or with high probability
- Las Vegas algorithms: only expected running time is polynomial; but when it terminates, it produces the correct answer
  - Zero error probability
Zero-Error Computation
Zero-Error Computation

- e.g. A simple algorithm for finding median in expected linear time

- (There are non-trivial algorithms to do it in deterministic linear time. Simple sorting takes $O(n \log n)$ time.)

- Procedure Find-element(L,k) to find $k^{th}$ smallest element in list L

- Pick random element $x$ in L. Scan L; divide it into $L_{>x}$ (elements $> x$) and $L_{<x}$ (elements $< x$); also determine position $m$ of $x$ in L.

- If $m = k$, return $x$. If $m > k$, call Find-element($L_{<x}$,k), else call Find-element($L_{>x}$,k-m)

- Correctness obvious. Expected running time?
Zero-Error Computation
Zero-Error Computation

- Expected running time (worst case over all lists of size n, and all k) be $T(n)$

- Time for non-recursive operations is linear: say bounded by $cn$. Will show inductively $T(n)$ at most $4cn$ (base case $n=1$).

- $T(n) \leq cn + 1/n \left[ \sum_{n>j>k} T(j) + \sum_{0<j<k} T(n-j) \right]$

- $T(n) \leq cn + 1/n.4c[\sum_{j>k} j + \sum_{j<k} (n-j)]$ by inductive hypothesis

- $\sum_{j>k} j + \sum_{j<k} (n-j) = \sum_{j>k} j + (k-1)n - \sum_{j<k} j \leq \sum_{j} j + (k-1)n - 2 \sum_{j<k} j$

  - $\leq n^2/2 + (k-1)n - k(k-1) < n^2/2 + k(n-k) \leq 3/4 \cdot n^2$

- $T(n) \leq cn + 3cn$ as required
Zero-Error Computation
Zero-Error Computation

Las-Vegas Algorithms: Probabilistic algorithms with deterministic outcome (but probabilistic run time)

$\text{ZPTIME}(T)$: class of languages decided by a zero-error probabilistic TM, with expected running time at most $T$

$\text{ZPP} = \text{ZPTIME}(\text{poly})$

$\text{ZPP} = \text{RP} \cap \text{co-RP}$
$\text{ZPP} \subseteq \text{RP}$
ZPP \subseteq \text{RP}

- Truncate after "long enough," and say "no"
- Do we still have bounded (one-sided) error?
- Will run for "too long" only with small probability
  - Because expected running time short
  - With high probability the running time does not exceed the expected running time by much
  - \( \Pr[ x > a \ E[X] ] < 1/a \) (non-negative \( X \))
- Markov's inequality
- \( \Pr[\text{error}] \) changes by at most \( 1/a \) if truncated after \( a \) times expected running time
\( \text{RP} \cap \text{co-RP} \subseteq \text{ZPP} \)
\( \mathsf{RP} \cap \mathsf{co-RP} \subseteq \mathsf{ZPP} \)

If \( L \in \mathsf{RP} \cap \mathsf{co-RP} \) a \( \mathsf{ZPP} \) algorithm for \( L \):

- Run both \( \mathsf{RP} \) and \( \mathsf{co-RP} \) algorithms

- If former says yes or latter says no, output that answer

- Else, i.e., if former says no and latter yes, repeat

- Expected number of repeats = \( O(1) \)
Today
Today

- Zoo
  - BPL $\subseteq$ P
- Expected running time
- Zero-Error probabilistic computation
- $ZPP = RP \cap \text{co-RP}$