Computational Complexity

Lecture 10: Probabilistic Algorithms

Ronald de Haan
me@ronalddehaan.eu

University of Amsterdam

March 5, 2021

Recap

- Non-uniform complexity
- Circuit complexity
- TMs that take advice
- The Karp-Lipton Theorem: if $\mathsf{NP} \subseteq \mathsf{P}/\mathsf{poly}$, then $\Sigma_2^p = \Pi_2^p$

What will we do today?

- Probabilistic algorithms
- Complexity classes BPP, RP, coRP, ZPP

Randomized algorithms

- Randomized (or probabilistic) algorithms are a realistic extension of deterministic algorithms
- They have access to a random number generator (or random coin flips)
- The outcome of such algorithms is a random variable
- The running time of such algorithms is a random variable

Example problem

- *Input:* you're given $m \in \mathbb{N}$ and you have access to an oracle O that can give you a value $O(i) \in \{a, b\}$, for each $i \in \{1, \ldots, 2^m\}$
- *Promise:* m is even and for exactly half of the i's it holds that $O(i) = a$, and so for the other half, $O(i) = b$
- *Task:* output some $i \in \{1, \ldots, 2^m\}$ such that $O(i) = a$

- When we consider deterministic (non-randomized) algorithms, what worst-case running time (and # of oracle queries) can we achieve for this problem?
  - We need ${}^{2^m}\!/_2 + 1 = 2^{m-1} + 1$ queries in the worst case, and $\Theta(2^m)$ time

Monte Carlo algorithm

```
i := 0;
while i < k do
    randomly pick j ∈ {1, ..., 2^m};
    query the oracle: o_j := O(j);
    if o_j = a then
        return j;
    else
        i := i + 1;
    end
end
randomly pick j ∈ {1, ..., 2^m};
return j;
```

- Runs for k rounds, so takes time $O(k \cdot m)$
- Probability of a correct answer: $1 - (1/2)^{k+1}$
- Works for any value of k
- The running time does not vary randomly
- Non-zero error probability

Las Vegas algorithm

while *True* **do**
  randomly pick $j \in \{1, \ldots, 2^m\}$;
  query the oracle: $o_j := O(j)$;
  if $o_j = a$ **then**
    return j;
  end
end

- The running time varies randomly (and is polynomial in expectation)
- Zero error probability

- Probability of a correct answer (given that it halted): 1
- Expected running time $O(m)$:

$$O(m) \cdot \left[1 \cdot 1/2 + 2 \cdot (1/2)^2 + 3 \cdot (1/2)^3 + \cdots\right] = O(m) \qquad \text{because} \quad \lim_{n \to \infty} \sum_{i=1}^{n} \frac{i}{2^i} = 2$$

Probabilistic Turing machines

Definition

Probabilistic Turing machines (PTM) are variants of (deterministic) TMs, where:

- There are two transition functions δ_1, δ_2.
- At each step, one of δ_1, δ_2 is chosen randomly, both with probability $1/2$. (Each such choice is made independently.)
- (As halting states, it has an accept state q_{acc} and a reject state q_{rej}.)

- $\mathbb{M}(x)$ denotes the random variable corresponding to the output of $\mathbb{M}$ on input x.
- $\mathbb{M}$ runs in time $T(n)$ if for every input x and every sequence of nondeterministic choices, $\mathbb{M}$ halts within $T(|x|)$ steps, regardless of the random choices made.

BPTIME and BPP

Definition (BPTIME)

Let $T : \mathbb{N} \to \mathbb{N}$ be a function. A problem $L \subseteq \{0,1\}^*$ is in $\mathsf{BPTIME}(T(n))$ if there exists a PTM $\mathbb{M}$ that runs in time $O(T(n))$, such that for each $x \in \{0,1\}^*$:

$$\mathbb{P}[\ \mathbb{M}(x) = L(x)\ ] \geq 2/3,$$

where $L(x) = 1$ if $x \in L$, and $L(x) = 0$ if $x \notin L$.

- BP: **B**ounded-error **P**robabilistic
- These are *Monte Carlo algorithms with two-sided (bounded) error*

Definition (BPP)

$$\mathsf{BPP} = \bigcup_{c \geq 1} \mathsf{BPTIME}(n^c).$$

Characterization of BPP

Theorem

A problem $L \subseteq \{0,1\}^*$ if and only if there exists a polynomial-time deterministic TM $\mathbb{M}$ and a polynomial $p : \mathbb{N} \to \mathbb{N}$ such that for each $x \in \{0,1\}^*$:

$$\mathop{\mathbb{P}}_{r \in_R \{0,1\}^{p(|x|)}} [\ \mathbb{M}(x, r) = L(x)\ ] \geq 2/3.$$

(Here $\in_R$ denotes (sampling from) the uniform distribution.)

- This is analogous to the verifier definition of NP
  - Using a probabilistic interpretation of the certificates, rather than existentially quantifying over them

One-sided error: RP and coRP

Definition (RTIME)

Let $T : \mathbb{N} \to \mathbb{N}$ be a function. A problem $L \subseteq \{0,1\}^*$ is in RTIME($T(n)$) if there exists a PTM $\mathbb{M}$ that runs in time $O(T(n))$, such that for each $x \in \{0,1\}^*$:

$$\text{if } x \in L, \text{ then } \mathbb{P}[\,\mathbb{M}(x) = 1\,] \geq 2/3,$$

$$\text{if } x \notin L, \text{ then } \mathbb{P}[\,\mathbb{M}(x) = 0\,] = 1.$$

- These are *Monte Carlo algorithms with one-sided (bounded) error*

Definition (RP)

$$\mathrm{RP} = \bigcup_{c \geq 1} \mathrm{RTIME}(n^c).$$

One-sided error: RP and coRP (ct'd)

Definition (coRTIME)

Let $T : \mathbb{N} \to \mathbb{N}$ be a function. A problem $L \subseteq \{0,1\}^*$ is in coRTIME$(T(n))$ if there exists a PTM $\mathbb{M}$ that runs in time $O(T(n))$, such that for each $x \in \{0,1\}^*$:

$$\text{if } x \in L, \text{ then } \mathbb{P}[\,\mathbb{M}(x) = 1\,] = 1,$$
$$\text{if } x \notin L, \text{ then } \mathbb{P}[\,\mathbb{M}(x) = 0\,] \geq 2/3.$$

- These are also *Monte Carlo algorithms with one-sided (bounded) error*

Definition (coRP)

$$\text{coRP} = \bigcup_{c \geq 1} \text{coRTIME}(n^c), \qquad \text{or equivalently: } \text{coRP} = \{\, \overline{L} \mid L \in \text{RP} \,\}.$$

Two-sided error: ZPP

Definition (expected running time)

Let $T : \mathbb{N} \to \mathbb{N}$ be a function and let $\mathbb{M}$ be a PTM. Then $\mathbb{M}$ runs in *expected time* $T(n)$, if for each $x \in \{0,1\}^*$ it holds that $\mathbb{E}[\ \mathrm{time}_{\mathbb{M}}(x)\ ] \leq T(|x|)$.

Definition (ZPTIME)

Let $T : \mathbb{N} \to \mathbb{N}$ be a function. A problem $L \subseteq \{0,1\}^*$ is in ZPTIME$(T(n))$ if there exists a PTM $\mathbb{M}$ that runs in expected time $O(T(n))$, such that for each $x \in \{0,1\}^*$, whenever $\mathbb{M}$ halts on x then $\mathbb{M}(x) = L(x)$.

- These are *Las Vegas algorithms*

Definition (ZPP)

$$\mathrm{ZPP} = \bigcup_{c \geq 1} \mathrm{ZPTIME}(n^c).$$

Error reduction

- We used the constant $2/3$ in the definitions of BPP, etc.
- In fact, each constant $> 1/2$ would work, and even $> 1/2 + |x|^{-c}$.
- We can make the error probability very small

Theorem (Error reduction for BPP)

Let $L \subseteq \{0,1\}^*$ be a decision problem, and suppose that there exists a polynomial-time PTM $\mathbb{M}$ such that for each $x \in \{0,1\}^*$, $\mathbb{P}[\mathbb{M}(x) = L(x)] \geq 1/2 + 1/|x|^c$.

Then for every constant $d > 0$, there exists a polynomial-time PTM $\mathbb{M}'$ such that for each $x \in \{0,1\}^*$, $\mathbb{P}[\mathbb{M}'(x) = L(x)] \geq 1 - 1/2^{(|x|^d)} = 1 - 2^{-|x|^d}$.

- Idea: run $\mathbb{M}$ many times and output the majority answer

Some relations

- RP $\subseteq$ BPP, coRP $\subseteq$ BPP
- RP $\subseteq$ NP, coRP $\subseteq$ coNP
  - Homework!
- ZPP $=$ RP $\cap$ coRP
  - Homework!
- BPP $\subseteq$ P/poly
  - Idea: by using error reduction, you can find some $r \in \{0,1\}^{p(n)}$ for each n that can be used as "certificate" to give the correct answer for each $x \in \{0,1\}^n$.
- BPP $\subseteq \Sigma_2^{\mathsf{p}}$, BPP $\subseteq \Pi_2^{\mathsf{p}}$

Recap

- Probabilistic algorithms
- Complexity classes BPP, RP, coRP, ZPP

Next time

- Approximation algorithms
- The PCP Theorem