INF 554: Using randomness in algorithms **Autumn 2014**

Lecture 1 — 15 September 2014

Lecturer: Frédéric Magniez *Scribe: Gurvan L'Hostis*

The goal of this course is to present a formal definition of randomized algorithms and some easy applications.

1.1 An introducing example : Freival's Algorithm

Decision problem:

- input: A, B and C, $n \times n$ matrices over an arbitrary ring
- output: decide if $A \times B = C$

Remarks: since 2011 with an improvement from Virginia Williams, an explicit matrix multiplication has an asymptotic complexity of $O(n^{2.3727})$.

Freivald's test:

- Choose $r \in \{0,1\}^n$ uniformly at random
- Evaluate $u = Cr$, $v = Br$ and $w = Av$
- Return $ACCEPT$ if $u = w$, else $REJECT$

This algorithm uses $3n^2$ additions and multiplications on the coefficients.

Theorem 1.1. *Freivald's algorithm has a one-sided error:*

- *If $AB = C$, $\mathbb{P}(algorithm\ accepts) = 1$*
- *If $AB \neq C$, $\mathbb{P}(algorithm\ rejects) \geq \frac{1}{2}$*

Remarks: If $AB \neq C$, since this algorithm has an one-sided error, by running k independent executions we have $\mathbb{P}$(algorithm accepts after k independent executions) $\leq 1/2^k$. In practice $k = 100$ is acceptable. For comparison, cosmic rays induce errors on computer with larger probability. In 1996, a studies by IBM revealed that they induced one error per 256 megabytes of RAM per month, which means a probability of 1.4×10^{-15} per byte per second, which is greater than 2^{-49}. Another comparison on large number, is that 2^{100} is far greater than the age of the universe in second, which is less than 2^{60} (for now...).

Proof:

- If $AB = C$ then $u = Cr = (AB)r = A(Br) = Av = w$ thus $\mathbb{P}$(algorithm accepts) $= 1$.

- If $AB \neq C$: particular case on $\mathbb{Z}_2$

  Let $F = \{r \in \{0,1\}^n : (A * B)r = Cr\} \subseteq 0, 1^n$. $F \neq \{0,1\}^n$ and F is a subspace of vector space $\{0,1\}^n$. Thus using Lagrange's Theorem we have $|F| \leq \frac{1}{2}|\{0,1\}^n|$

  Hence $\mathbb{P}(\text{algorithm accepts}) = \mathbb{P}(r \in F) \leq \frac{|F|}{|0,1^n|} \leq \frac{1}{2}$

- If $AB \neq C$: general case

  Assume there are two indices i and j such that $(AB)_{ij} \neq C_{ij}$. Let $D = C - AB$. Then $D_{ij} \neq 0$, $D \neq 0$. We want to prove $\underset{r \in \{0,1\}^n}{\mathbb{P}} [Dr = 0] \leq \frac{1}{2}$.

$$(Dr)_i = \sum_k D_{ik} r_k = D_{ij} r_j + f((r_k)_{k \neq j})$$

$$\mathbb{P}\left[Dr = 0\right] \leq \mathbb{P}\left[(Dr)_i = 0\right]$$

  Fix $r_1, \ldots, r_n$ excepts r_j. Then $v = f((r_k)_{k \neq j})$.

  - If $v = -D_{ij}$: if $r_j = 0$ then $(Dr)_i \neq 0$, if $r_j = 1$ then $(Dr)_i = D_{ij} - D_{ij} = 0$. Conditional probability of $(Dr)_i = 0$ is $\frac{1}{2}$.
  - If $v = 0$: if $r_j = 0$ then $(Dr)_i = 0$, if $r_j = 1$ then $(Dr)_i = D_{ij} \neq 0$. Conditional probability of $(Dr)_i = 0$ is $\frac{1}{2}$.
  - Otherwise: for $r_j = 0, 1$ $(Dr)_i \neq 0$.

$$\mathbb{P}\left[(Dr)_i = 0\right] \leq \frac{1}{2}$$

$\square$

1.2 Formal basis

1.2.1 Deterministic algorithm

Input: x $\longrightarrow$ $\boxed{\text{Algorithm}}$ $\longrightarrow$ Output $A(x)$

Goal:

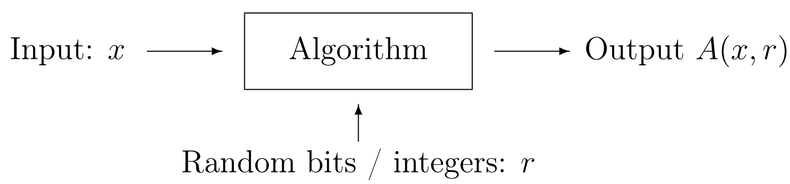
- correctly solve the problem on all inputs
- efficiency (wished): linear or polynomial time on input size in bytes

1.2.2 Randomized algorithm

A randomized algorithm, compared to a deterministic algorithm, has an additional input: the random variable r. We suppose that we have access to a source of uniform random bits or integers (which is basically equivalent).

Remarks:

- Behaviour depends on both x and r.
- Once r is fixed, the algorithm is deterministic.
- We do not know yet how to generate random numbers with computers, we have only access to pseudo-random generators.



Definition 1.2. *A solves a problem P with error δ if for all inputs x it verifies*

$$\mathbb{P}[A(x, r) \text{ is correct}] \geq 1 - \delta$$

A such problem is in the BPP class[1].

Let L be a language. A recognizes L with error δ on one side if for all inputs x :

- *if $x \in L$ then $A(x, r)$ accepts for all r ;*
- *if $x \notin L$ then $\mathbb{P}[A(x, r) \text{ accepts}] \leq \delta$.*

A such problem is in the RP class[2].

Remarks:

- One-sided error algorithm : if $\delta_0 < 1$ we can get to δ by iterating A $\lceil \frac{\log \delta}{\log \delta_0} \rceil$ times (in practice $\delta_0 \leq \frac{1}{2}$).
- Double-sided error algorithm : if $\delta_0 < \frac{1}{2}$ we can get to δ by iterating A $\lceil \frac{1}{(\frac{1}{2} - \delta_0)^2} \log \frac{1}{\delta} \rceil$ times (in practice $\delta_0 \leq \frac{1}{3}$).

[1] BPP: Bounded-error Probabilistic Polynomial time
[2] RP: Randomized Polynomial time

1.3 Primality Testing

Decision problem:

- input: an integer $n \geq 2$
- output: decide if n is prime

n is $k = \log_2(n)$ long. The sieve of Eratosthenes gives a result in $\sqrt{n}$ steps which is too long ($O(2^{k/2})$ operations). The best deterministic algorithm is Agrawal-Kayal-Saxena in $O((\log_2(n))^6)$ (2005).

1.3.1 Fermat's little theorem approach

Fermat's little theorem

Theorem 1.3. *$p \geq 2$ prime number $\Rightarrow \forall a \in [1, p-1], a^{p-1} = 1[p]$*

Tentative algorithm

Primality test algorithm:

- Input: $N \geq 2$
- Select a random $a \in [1, N-1]$
- If $a \wedge N \neq 1$ then reject (in this case N is not prime, because $(a \wedge N)|N$)
- Compute a^{N-1} with rapid exponentiation: $a^{2r} = (a^r)^2$, $a^{2r+1} = a(a^r)^2$
- Accept if $a^{N-1} = 1\,[N]$, otherwise reject

Remarks:

- Running time is $O(\log N)$.
- If N is prime then the algorithm accepts N with probability 1.

Algorithm's proof

Lemma 1.4. *Assume there is $1 \leq a < N$ such that $a \wedge N = 1$ and $a^{N-1} \neq 1\,[N]$. Then $\underset{1\leq a<N}{\mathbb{P}}[a^{N-1} = 1\,[N]\,|a \wedge N = 1] \leq \frac{1}{2}$*

Proof: Let $G = \{b \in \{1, ..., N-1\}|GCD(b, N) = 1\}$. G is an abelian group for the operation (X mod N). Let $F = \{b \in G|b^{N-1} = 1[N]\}$.
$F \neq G$ and F is a subgroup hence $|F| \leq 1/2|G|$ (Lagrange's Theorem) □

Corollary 1.5. *Assume there is $1 \leq a < N$ such that $a \wedge N = 1$ and $a^{N-1} \neq 1\,[N]$. Then $\underset{a}{\mathbb{P}}(\text{algorithm accepts } N) \leq \frac{1}{2}$*

Proof: Take N non prime such that there is $1 \leq a < N$ such that $a \wedge N = 1$ and $a^{N-1} \neq 1\,[N]$

$$\begin{aligned}
\mathbb{P}_a(algorithm\ accepts\ N) &= \mathbb{P}_a(a \wedge N = 1 \ and\ a^{N-1} = 1\,[N]) \\
&= \underbrace{\mathbb{P}_a(a^{N-1} = 1\,[N]\,|a \wedge N = 1)}_{\leq \frac{1}{2}} \times \underbrace{\mathbb{P}_a(a \wedge N = 1)}_{\leq 1} \\
&\leq \frac{1}{2}
\end{aligned}$$

□

Carmichael number

Definition 1.6. *An non-prime integer N is a* Carmichael number *if all $1 \leq a < N$ such that $a \wedge N = 1$ satisfy $a^{N-1} \neq 1\,[N]$.*

The smallest Carmichael number is $561 = 3 \times 11 \times 17$.
There are 255 Carmichael number $\leq 10^8$

1.3.2 Miller-Rabin test

Lemma 1.7. *If p is prime then the only solution of $x^2 = 1\,[p]$ are $\pm 1 \mod p$.*

Algorithm

- Input: $N \geq 2$
- If $N = 2$, ACCEPT. Otherwise if $2|N$, REJECT.
- Take $a \in [2, N-1]$ uniformly at random.
- If $a \wedge N \neq 1$, REJECT
- Let $N - 1 = 2^t u$ ($t \geq 1$ since N is odd). Compute $b = a^u$. Let $i \leq t$ be the smallest integer such that $b^{2^i} = 1$.
- If i does not exist, REJECT (since $b^{2^t} \neq 1\,[N]$, Fermat's test fails)
- If $i = 0$ or $b^{2^{i-1}} = -1$, ACCEPT
- Otherwise, REJECT

Remark: Running time is $O(\log N)$.

1.4 Reminder on Probabilities

1.4.1 Definitions

- **Discrete random variable** X (finite) from Ω (finite)

  Example: random bit B on $\Omega = \{0, 1\}$

- **Stochastic process**: $(X_t)_{t \in T}$ with $T \in \mathbb{N}$

- **Halting time** τ such as $\tau = t$ depends only from $X_1, ..., X_t$

  Example: τ: time to get a 0 from a random bit stream, $\mathbb{E}(\tau) = 2$

  - $\mathbb{P}[\tau = 1] = 1/2$
  - $\mathbb{P}[\tau = 2] = 1/4$
  - $\mathbb{P}[\tau = k] = 1/2^k$
  - $\mathbb{E}(\tau) = \sum_k k\mathbb{P}[\tau = k] = 2$

1.4.2 Bernoulli

Theorem 1.8. *If* $\mathbb{P}[B_t = 0] = p$*, then* $\mathbb{E}(\tau) = 1/p$

Application

Let $\Omega = \{1, 2, ..., n\}$, X a discrete random value from ω, $X_1, ..., X_t$ a stochastic process
Let τ be the smallest t such as $\{X_1, ..., X_t\} = \Omega$

Then $\mathbb{E}(\tau) \approx n \log(n)$

Proof

$\tau = \sum_{i=1}^{n} \tau_i$ with τ_i time to get a new value knowing we already have $i - 1$ different values.
Then $\mathbb{P}(\tau_i) = \frac{n-i+1}{n}$ and using the theorem we have $\mathbb{E}(\tau_i) = \frac{n}{n-i+1}$
Hence $\mathbb{E}(\tau) = \sum_{i=1}^{n} \mathbb{E}(\tau_i) = \sum_{i=1}^{n} \frac{n}{n-i+1} \approx n \log(n)$

1.4.3 Markov inequality

Theorem 1.9. $X \geq 0$ *a discrete random variable,* $\mu = \mathbb{E}(X)$*. Then* $\forall a > 0, \mathbb{P}(X > a\mu) \leq \frac{1}{a}$

1.4.4 Chernoff bound

Theorem 1.10. $X_1, ..., X_n$ *independent random variables from* $\{0, 1\}$ *such as* $\forall i, \mathbb{P}[X_i = 1] = \mu_i = \mathbb{E}(X_i)$*. Let* $X = \frac{1}{n} \sum X_i$ *and* $\mu = \frac{1}{n} \sum \mu_i = \mathbb{E}(X)$
Then $\forall \delta > 0, \mathbb{P}[|X - \mu| \geq \delta\mu] \leq 2^{-\mu\delta^2 n/3}$