Appendix A

Basic number theory

A.1 Cyclic groups

not necessarily abelian group.

Notation: for a finite cyclic group $\mathbb{G}$ we let $\mathbb{G}^*$ denote the set of generators of $\mathbb{G}$.

↳ e.g. $(\mathbb{Z}; +)$

A.2 Arithmetic modulo primes

A.2.1 Basic concepts

We use the letters p and q to denote prime numbers. We will be using large primes, e.g. on the order of 300 digits (1024 bits).

1. For a prime p let $\mathbb{Z}_p = \{0, 1, 2, \ldots, p-1\}$.
   Elements of $\mathbb{Z}_p$ can be added modulo p and multiplied modulo p. For $x, y \in \mathbb{Z}_p$ we write $x + y$ and $x \cdot y$ to denote the sum and product of x and y modulo p.

2. Fermat's theorem: $g^{p-1} = 1$ for all $0 \neq g \in \mathbb{Z}_p$
   Example: $3^4 = 81 \equiv 1 \pmod 5$.

3. The *inverse* of $x \in \mathbb{Z}_p$ is an element $a \in \mathbb{Z}_p$ satisfying $a \cdot x = 1$ in $\mathbb{Z}_p$.
   The inverse of x in $\mathbb{Z}_p$ is denoted by x^{-1}.
   Example: 1. 3^{-1} in $\mathbb{Z}_5$ is 2 since $2 \cdot 3 \equiv 1 \pmod 5$.
   2. 2^{-1} in $\mathbb{Z}_p$ is $\frac{p+1}{2}$.

4. All elements $x \in \mathbb{Z}_p$ except for $x = 0$ are invertible.
   Simple (but inefficient) inversion algorithm: $x^{-1} = x^{p-2}$ in $\mathbb{Z}_p$.
   Indeed, $x^{p-2} \cdot x = x^{p-1} = 1$ in $\mathbb{Z}_p$.

5. We denote by $\mathbb{Z}_p^*$ the set of invertible elements in $\mathbb{Z}_p$. Then $\mathbb{Z}_p^* = \{1, 2, \ldots, p-1\}$.

6. We now have algorithm for solving linear equations in $\mathbb{Z}_p$: $a \cdot x = b$.
   Solution: $x = b \cdot a^{-1} = b \cdot a^{p-2}$.
   What about an algorithm for solving quadratic equations?

(2) *The following are equivalent:*
(i) *n is prime;*
(ii) *$\mathbb{Z}_n$ is an integral domain;*
(iii) *$\mathbb{Z}_n$ is a field.*

EEA is more efficient to compute x^{-1}.
(why inefficient?)

each is call a "unit"

$\mathbb{Z}_p$ v.s. F_p. (cyclic group v.s. finite field)

1 operation $(+) \bmod p$

addition & multiplication $(+, \cdot) \bmod p$.

!! field is only cyclic under $(\bmod p)$ if F_n. $n = p^d$ (power of prime)

A.2.2 Structure of $\mathbb{Z}_p^*$

1. $\mathbb{Z}_p^*$ is a *cyclic group*.
   In other words, there exists $g \in \mathbb{Z}_p^*$ such that $\mathbb{Z}_p^* = \{1, g, g^2, g^3, \ldots, g^{p-2}\}$.
   Such a g is called a *generator* of $\mathbb{Z}_p^*$.
   Example: in $\mathbb{Z}_7^*$: $\langle 3 \rangle = \{1, 3, 3^2, 3^3, 3^4, 3^5, 3^6\} \equiv \{1, 3, 2, 6, 4, 5\} \pmod 7 = \mathbb{Z}_7^*$.

2. Not every element of $\mathbb{Z}_p^*$ is a generator.
   Example: in $\mathbb{Z}_7^*$ we have $\langle 2 \rangle = \{1, 2, 4\} \neq \mathbb{Z}_7^*$.

3. The *order* of $g \in \mathbb{Z}_p^*$ is the smallest positive integer a such that $g^a = 1$.
   The order of $g \in \mathbb{Z}_p^*$ is denoted $\text{order}_p(g)$.
   Example: $\text{order}_7(3) = 6$ and $\text{order}_7(2) = 3$.

4. Lagrange's theorem: for all $g \in \mathbb{Z}_p^*$ we have that $\text{order}_p(g)$ divides $p-1$. Observe that Fermat's theorem is a simple corollary:
   for $g \in \mathbb{Z}_p^*$ we have $g^{p-1} = (g^{\text{order}(g)})^{(p-1)/\text{order}(g)} = (1)^{(p-1)/\text{order}(g)} = 1$.

5. If the factorization of $p-1$ is known then there is a simple and efficient algorithm to determine $\text{order}_p(g)$ for any $g \in \mathbb{Z}_p^*$.

A.2.3 Quadratic residues

1. The *square root* of $x \in \mathbb{Z}_p$ is a number $y \in \mathbb{Z}_p$ such that $y^2 = x \bmod p$.
   Example: 1. $\sqrt{2}$ in $\mathbb{Z}_7$ is 3 since $3^2 = 2 \bmod 7$.
   2. $\sqrt{3}$ in $\mathbb{Z}_7$ does not exist.

2. An element $x \in \mathbb{Z}_p^*$ is called a *Quadratic Residue* (QR for short) if it has a square root in $\mathbb{Z}_p^*$.

3. How many square roots does $x \in \mathbb{Z}_p$ have?
   If $x^2 = y^2$ in $\mathbb{Z}_p$ then $0 = x^2 - y^2 = (x-y)(x+y)$.
   $\mathbb{Z}_p$ is an "integral domain" which implies that $x - y = 0$ or $x + y = 0$, namely $x = \pm y$.
   Hence, elements in $\mathbb{Z}_p$ have either zero square roots or two square roots.
   If a is the square root of x then $-a$ is also a square root of x in $\mathbb{Z}_p$.

4. Euler's theorem: $x \in \mathbb{Z}_p$ is a QR if and only if $x^{(p-1)/2} = 1$.
   Example: $2^{(7-1)/2} = 1$ in $\mathbb{Z}_7$ but $3^{(7-1)/2} = -1$ in $\mathbb{Z}_7$.

5. Let $g \in \mathbb{Z}_p^*$. Then $a = g^{(p-1)/2}$ is a square root of 1. Indeed, $a^2 = g^{p-1} = 1$ in $\mathbb{Z}_p$.
   Square roots of 1 in $\mathbb{Z}_p$ are 1 and -1.
   Hence, for $g \in \mathbb{Z}_p^*$ we know that $g^{(p-1)/2}$ is 1 or -1.

6. Legendre symbol: for $x \in Z_p$ define $\left(\frac{x}{p}\right) := \begin{cases} 1 & \text{if } x \text{ is a QR in } \mathbb{Z}_p \\ -1 & \text{if } x \text{ is not a QR in } \mathbb{Z}_p \\ 0 & \text{if } x = 0 \bmod p \end{cases}$.

7. By Euler's theorem we know that $\left(\frac{x}{p}\right) = x^{(p-1)/2}$ in $\mathbb{Z}_p$.
   $\Longrightarrow$ the Legendre symbol can be efficiently computed.

from Shoup

We then see that $\mathbb{Z}_p^*$ consists of the residue classes

$$[\pm 1], \ldots, [\pm(p-1)/2],$$

and so $(\mathbb{Z}_p^*)^2$ consists of the residue classes

$$[1]^2, \ldots, [(p-1)/2]^2,$$

NOTE: ① in Z_p, $\left(\frac{g}{p}\right) = -1$, i.e. generator of a finite cyclic group is a non-quadratic residue.

because if $x^2 = g \bmod p$.
then g won't be a generator
(e.g. $x^3 \neq g^\alpha \bmod p$)

② $\left(\frac{u \cdot v}{p}\right) = \left(\frac{u}{p}\right) \cdot \left(\frac{v}{p}\right)$ in Legendre Symbol.

i.e. $\begin{cases} QR \cdot QR = QR \\ \overline{QR} \cdot \overline{QR} = QR \\ QR \cdot \overline{QR} = \overline{QR} \end{cases}$

② ⇒ ③ DDH in Z_p: $g^\alpha, g^\beta, g^{\alpha\beta} \Leftrightarrow g^\gamma$ ($\gamma \xleftarrow{R} Z_p$)

$\left(\frac{g^{\alpha\beta}}{p}\right) = 1$ if either α or β is even $\Pr\left[\left(\frac{g^{\alpha\beta}}{p}\right) = 1\right] = \frac{3}{4}$

∵ γ is chosen at random, $\Pr\left[\left(\frac{g^\gamma}{p}\right) = 1\right] = \frac{1}{2}$

↳ then distinguish DDH w/ non-negligible prob.

The question then remains: in what groups is the DDH assumed to be hard (as hard as a complete DL)? One example is the subgroup of k residues modulo a prime p, where $(p-1)/k$ is also a large prime. The reason ties back to our assessment above. When $k = 2$, the group of quadratic residues modulo a safe prime will contain elements where for each one the Legendre symbol is 1. In other words, the above trick doesn't work.

↓ ideally, may all elements share the same Legendre Symbol.

$[k]_p$ and $\frac{p-1}{k}$ is a large prime.

e.g. $[2]_p$ & $\frac{p-1}{2}$ is a prime

$\alpha \equiv g^{\frac{p-1}{2}}$ for some $g \in Z_p^*$

8. Easy fact: let g be a generator of $\mathbb{Z}_p^*$. Let $x = g^r$ for some integer r.
   Then x is a QR in $\mathbb{Z}_p$ if and only if r is even.
   $\Longrightarrow$ **the Legendre symbol reveals the parity of r.**

9. Since $x = g^r$ is a QR if and only if r is even it follows that exactly half the elements of $\mathbb{Z}_p$ are QR's.

10. When $p = 3 \bmod 4$ computing square roots of $x \in \mathbb{Z}_p$ is easy.
    Simply compute $a = x^{(p+1)/4}$ in $\mathbb{Z}_p$.
    $a = \sqrt{x}$ since $a^2 = x^{(p+1)/2} = x \cdot x^{(p-1)/2} = x \cdot 1 = x$ in $\mathbb{Z}_p$.

11. When $p = 1 \bmod 4$ computing square roots in $\mathbb{Z}_p$ is possible but somewhat more complicated; it requires a randomized algorithm.

12. We now have an algorithm for solving quadratic equations in $\mathbb{Z}_p$.
    We know that if a solution to $ax^2 + bx + c = 0 \bmod p$ exists then it is given by:

$$x_{1,2} = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

    in $\mathbb{Z}_p$. Hence, the equation has a solution in $\mathbb{Z}_p$ if and only if $\Delta = b^2 - 4ac$ is a QR in $\mathbb{Z}_p$. Using our algorithm for taking square roots in $\mathbb{Z}_p$ we can find $\sqrt{\Delta} \bmod p$ and recover x_1 and x_2.

13. What about cubic equations in $\mathbb{Z}_p$? There exists an efficient randomized algorithm that solves any equation of degree d in time polynomial in d.

A.2.4 Computing in $\mathbb{Z}_p$

1. Since p is a huge prime (e.g. 1024 bits long) it cannot be stored in a single register.

2. Elements of $\mathbb{Z}_p$ are stored in buckets where each bucket is 32 or 64 bits long depending on the processor's chip size.

3. Adding two elements $x, y \in \mathbb{Z}_p$ can be done in linear time in the *length* of p.

4. Multiplying two elements $x, y \in \mathbb{Z}_p$ can be done in quadratic time in the *length* of p. If p is n bits long, better algorithms work in time $O(n^{1.7})$ (rather than $O(n^2)$).

5. Inverting an element $x \in \mathbb{Z}_p$ can be done in quadratic time in the length of p.

6. Using the repeated squaring algorithm, $x^r \bmod p$ can be computed in time $(\log_2 r)O(n^2)$ where p is n bits long. Note, the algorithm takes linear time in the length of r.

A.2.5 Summary: arithmetic modulo primes

Let p be a 1024 bit prime. Easy problems in $\mathbb{Z}_p$:

1. Generating a random element. Adding and multiplying elements.

2. Computing $g^r \bmod p$ is easy even if r is very large.

3. Inverting an element. Solving linear systems.
4. Testing if an element is a QR and computing its square root if it is a QR.
5. Solving polynomial equations of degree d can be done in polynomial time in d.

Problems that are believed to be hard in $\mathbb{Z}_p$:

1. Let g be a generator of $\mathbb{Z}_p^*$. Given $x \in \mathbb{Z}_p^*$ find an r such that $x = g^r \bmod p$. This is known as the *discrete log problem.*
2. Let g be a generator of $\mathbb{Z}_p^*$. Given $x, y \in \mathbb{Z}_p^*$ where $x = g^{r_1}$ and $y = g^{r_2}$. Find $z = g^{r_1 r_2}$. This is known as the *Diffie-Hellman problem.*

A.3 Arithmetic modulo composites

We are dealing with integers n on the order of 300 digits long, (1024 bits). Unless otherwise stated, we assume that n is the product of two equal size primes, e.g. on the order of 150 digits each (512 bits).

1. For a composite n let $\mathbb{Z}_n = \{0, 1, 2, \ldots, n-1\}$.
   Elements of $\mathbb{Z}_n$ can be added and multiplied modulo n.
2. The inverse of $x \in \mathbb{Z}_n$ is an element $y \in \mathbb{Z}_n$ such that $x \cdot y = 1 \bmod n$.
   An element $x \in \mathbb{Z}_n$ has an inverse if and only if x and n are relatively prime. In other words, $\gcd(x, n) = 1$.
3. Elements of $\mathbb{Z}_n$ can be efficiently inverted using Euclid's algorithm. If $\gcd(x, n) = 1$ then using Euclid's algorithm it is possible to efficiently construct two integers $a, b \in \mathbb{Z}$ such that $ax + bn = 1$. Reducing this relation modulo n leads to $ax = 1 \bmod n$. Hence $a = x^{-1} \bmod n$.
   note: this inversion algorithm also works in $\mathbb{Z}_p$ for a prime p and is more efficient than inverting x by computing $x^{p-2} \bmod p$.
4. We let $\mathbb{Z}_n^*$ denote the set of invertible elements in $\mathbb{Z}_n$.
5. We now have an algorithm for solving linear equations: $a \cdot x = b \bmod n$.
   Solution: $x = b \cdot a^{-1}$ where a^{-1} is computed using Euclid's algorithm.
6. How many elements are in $\mathbb{Z}_n^*$? We denote by $\varphi(n)$ the number of elements in $\mathbb{Z}_n^*$. We already know that $\varphi(p) = p - 1$ for a prime p.
7. One can show that if $n = p_1^{e_1} \cdots p_m^{e_m}$ then $\varphi(n) = n \cdot \prod_{i=1}^{m} \left(1 - \frac{1}{p_i}\right)$.
   In particular, when $n = pq$ we have that $\varphi(n) = (p-1)(q-1) = n - p - q + 1$.
   Example: $\varphi(15) = \left|\{1, 2, 4, 7, 8, 11, 13, 14\}\right| = 8 = 2 * 4$.
8. Euler's theorem: all $a \in \mathbb{Z}_n^*$ satisfy $a^{\varphi(n)} = 1$ in $\mathbb{Z}_n$.
   note: For primes p Euler's theorem implies that $a^{\varphi(p)} = a^{p-1} = 1$ for all $a \in \mathbb{Z}_p^*$. Hence, Euler's theorem is a generalization of Fermat's theorem.

Theorem 2.11. *If $n = p_1^{e_1} \cdots p_r^{e_r}$ is the factorization of n into primes, then*

$$\varphi(n) = \prod_{i=1}^{r} p_i^{e_i - 1}(p_i - 1) = n \prod_{i=1}^{r} (1 - 1/p_i).$$

Theorem 2.6 (Chinese remainder theorem). *Let $\{n_i\}_{i=1}^k$ be a pairwise relatively prime family of positive integers, and let $a_1, \ldots, a_k$ be arbitrary integers. Then there exists a solution $a \in \mathbb{Z}$ to the system of congruences*

$$a \equiv a_i \pmod{n_i} \quad (i = 1, \ldots, k).$$

Moreover, any $a' \in \mathbb{Z}$ is a solution to this system of congruences if and only if $a \equiv a' \pmod{n}$, where $n := \prod_{i=1}^k n_i$.

Structure of $\mathbb{Z}_n$

Theorem A.1 (Chinese Remainder Theorem (CRT)). *state theorem*

Summary

Let n be a 1024 bit integer which is a product of two 512 bit primes. Easy problems in $\mathbb{Z}_n$:

1. Generating a random element. Adding and multiplying elements.
2. Computing $g^r \bmod n$ is easy even if r is very large.
3. Inverting an element. Solving linear systems.

Problems that are believed to be hard if the factorization of n is unknown, but become easy if the factorization of n is known:

1. Finding the prime factors of n.
2. Testing if an element is a QR in $\mathbb{Z}_n$.
3. Computing the square root of a QR in $\mathbb{Z}_n$. This is provably as hard as factoring n. When the factorization of $n = pq$ is known one computes the square root of $x \in \mathbb{Z}_n^*$ by first computing the square root in $\mathbb{Z}_p$ of $x \bmod p$ and the square root in $\mathbb{Z}_q$ of $x \bmod q$ and then using the CRT to obtain the square root of x in $\mathbb{Z}_n$.
4. Computing e'th roots modulo n when $\gcd(e, \varphi(n)) = 1$. $x^{\frac{1}{e}}$ when $\gcd(e, \varphi(n)) = 1$, $x^{\varphi(n)} = 1$
5. More generally, solving polynomial equations of degree $d > 1$. This problem is easy if the factorization of n is known: one first finds the roots of the polynomial equation modulo the prime factors of n and then uses the CRT to obtain the roots in $\mathbb{Z}_n$.

Problems that are believed to be hard in $\mathbb{Z}_n$:

1. Let g be a generator of $\mathbb{Z}_n^*$. Given $x \in \mathbb{Z}_n^*$ find an r such that $x = g^r \bmod n$. This is known as the *discrete log problem*.
2. Let g be a generator of $\mathbb{Z}_n^*$. Given $x, y \in \mathbb{Z}_n^*$ where $x = g^{r_1}$ and $y = g^{r_2}$. Find $z = g^{r_1 r_2}$. This is known as the *Diffie-Hellman problem*.

Q: when do we need group $\mathbb{Z}_n^*$ instead of $\mathbb{Z}_n$?
↳ when "inverse" is needed, $r \in \mathbb{Z}$, thus could be g^r where $r < 0$.
↳ $\mathbb{Z}_n^*$ is a group, whereas $\mathbb{Z}_n$ is either a commutative ring

Appendix B

Basic probability theory

Includes a description of statistical distance.

B.1 Birthday Paradox

Theorem B.1. *Let $\mathcal{M}$ be a set of size n and let $X_1, \ldots, X_k$ be k independent random variables uniform in $\mathcal{M}$. Let C be the event that for some distinct $i, j \in \{1, \ldots, k\}$ we have that $X_i = X_j$. Then*

↳ i.e. collision

(i) $\Pr[C] \geq 1 - e^{-k(k-1)/2n} \geq \min\left\{\frac{k(k-1)}{4n}, 0.63\right\}$, *and*

(ii) $\Pr[C] \leq 1 - e^{-k(k-1)/n}$ *when* $k < n/2$.

Proof. These all follow easily from the inequality

$$1 - x \leq e^{-x} \leq 1 - x/2,$$

which holds for all $x \in [0, 1]$. □

Most frequently we will use the lower bound to say that a collision happens with *at least* a certain probability. But occasionally we will use the upper bound to argue that collisions do not happen.

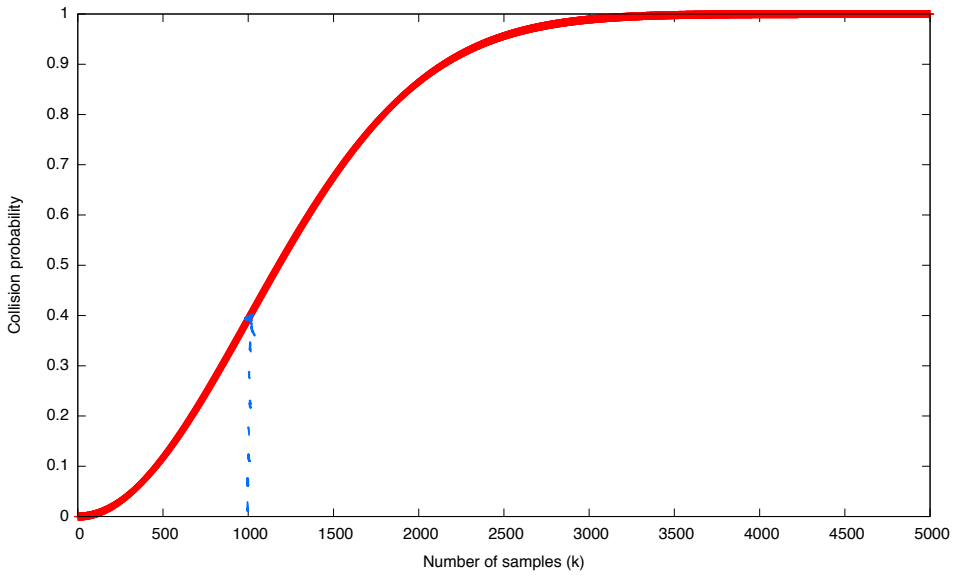
It is well documented that birthdays are not really uniform throughout the year. For example, in the U.S. the percentage of births in September is higher than in any other month. We show next that this non-uniformity only increases the probability of collision.

We present a stronger version of the birthday paradox that applies to independent random variables that are not necessarily uniform in $\mathcal{M}$. We do, however, require that all random variables are identically distributed. Such random variables are called i.i.d (independent and identically distributed). This version of the birthday paradox is due to Blom [Blom, D. (1973), "A birthday problem", American Mathematical Monthly, vol. 80, pp. 1141-1142].

Corollary B.2. *Let $\mathcal{M}$ be a set of size n and let $X_1, \ldots, X_k$ be k i.i.d random variables over $\mathcal{M}$ where $k \geq 2$. Let C be the event that for some distinct $i, j \in \{1, \ldots, k\}$ we have that $X_i = X_j$. Then*

$$\Pr[C] \geq 1 - e^{-k(k-1)/2n} \geq \min\left\{\frac{k(k-1)}{4n}, 0.63\right\}.$$

NOTE: NOT necessarily uniformly distributed, but independent and identically distributed.



The graph shows that collision probability for $n = 10^6$ elements and k ranging from one sample to 5000 samples. It illustrates the threshold phenomenon around the square root. At the square root, $\sqrt{n} = 1000$, the collision probability is about 0.4. Already at $4\sqrt{n} = 4000$ the collision probability is almost 1. At $0.5\sqrt{n} = 500$ the collision probability is small.

Figure B.1: Birthday Paradox

Proof. Let X be a random variable distributed as X_1. Let $\mathcal{M} = \{a_1, \ldots, a_n\}$ and let $p_i = \Pr[X = a_i]$. Let I be the set of all k-tuples over $\mathcal{M}$ containing distinct elements. Then I contains $\binom{n}{k}k!$ tuples. Since the variables are independent we have that:

$$\Pr[\neg C] = \sum_{(b_1,\ldots,b_k)\in I} \Pr[X_1 = b_1 \wedge \ldots \wedge X_k = b_k] = \sum_{(b_1,\ldots,b_k)\in I} \prod_{j=1}^{k} p_{b_j} \tag{B.1}$$

We show that this sum is maximized when $p_1 = p_2 = \ldots = p_n = 1/n$. This will mean that the probability of collision is minimized when all the variables are uniform. The Corollary will then follow from Theorem B.1.

Suppose some p_i is not $1/n$, say $p_i < 1/n$. Since $\sum_{j=1}^{n} p_i = 1$ there must be another p_j such that $p_j > 1/n$. Let $\epsilon = \min((1/n) - p_i,\ p_j - 1/n)$ and note that $p_j - p_i > \epsilon$. We show that replacing p_i by $p_i + \epsilon$ and p_j by $p_j - \epsilon$ increases the value of the sum in (B.1). Clearly, the resulting $p_1, \ldots, p_n$ still sum to 1. Hence, the resulting $p_1, \ldots, p_n$ form a distribution over $\mathcal{M}$ in which there is one less value that is not $1/n$. Furthermore, the probability of no collision in this distribution is greater than in the unmodified distribution. Repeating this replacement process at most n times will show that the sum is maximized when all the p_i's are equal to $1/n$. Again, this means that the probability of not getting a collision is maximized when the variables are uniform.

Now, consider the sum in (B.1). There are four types of terms. First, there are terms that do not contain either p_i or p_j. These terms are unaffected by the change to p_i and p_j. Second, there are terms that contain exactly one of p_i or p_j. These terms pair up. For every k-tuple that contains i but not j there is a corresponding tuple that contains j but not i. Then the sum of the corresponding two terms in (B.1) looks like $A(p_i + \epsilon) + A(p_j - \epsilon)$ for some $A \in [0, 1]$. Since this equals $Ap_i + Ap_j$ the sum of these two terms is not affected by the change to p_i and p_j. Finally, there are terms in (B.1) that contain both p_i and p_j. These terms change by

$$B(p_i + \epsilon)(p_j - \epsilon) - Bp_ip_j = B[\epsilon(p_j - p_i) - \epsilon^2]$$

for some $B \in [0, 1]$. By definition of ϵ we know that $p_j - p_i > \epsilon$ and therefore $\epsilon(p_j - p_i) - \epsilon^2 > 0$. Hence, the sum with modified p_i and p_j is larger than the sum with the unmodified values.

Overall, we proved that the modification to p_i and p_j increases the value of the sum in (B.1), as required. This completes the proof of the Corollary. □

B.1.1 More collision bounds

Consider the sequence $x_i \leftarrow f(x_{i-1})$ for a random function $f : \mathcal{X} \to \mathcal{X}$. Analyze the cycle time of this walk (needed for Pollard). Now, consider the same sequence for a permutation $\pi : \mathcal{X} \to \mathcal{X}$. Analyze the cycle time (needed for analysis of SecurID identification).

B.1.2 A simple distinguisher

We describe a simple algorithm that distinguishes two distributions on strings in $\{0, 1\}^n$. Let $X_1, \ldots, X_n$ and $Y_1, \ldots, Y_n$ be independent random variables taking values in $\{0, 1\}$. Then

$$X := (X_1, \ldots, X_n) \qquad \text{and} \qquad Y := (Y_1, \ldots, Y_n)$$

are elements of $\{0, 1\}^n$. Suppose, that for $i = 1, \ldots, n$ we have

$$\Pr[X_i = 1] = p \qquad \text{and} \qquad \Pr[Y_i = 1] = (1 + 2\epsilon) \cdot p$$

for some $p \in [0,1]$ and some small $\epsilon > 0$ so that $(1+2\epsilon) \cdot p \leq 1$. Then X and Y induce two distinct distributions on $\{0,1\}^n$.

We are given an n-bit string T and are told that it is either sampled according to the distribution X or the distribution Y, so that both p and ϵ are known to us. Our goal is to decide which distribution T was sampled from. Consider the following simple algorithm $\mathcal{A}$:

input: $T = (t_1, \ldots, t_n) \in \{0,1\}^n$
output: 1 if T is sampled from X and 0 otherwise
$s \leftarrow (1/n) \cdot \sum_{i=1}^{n} t_i$
if $s > p \cdot (1+\epsilon)$ output 0 else output 1

We are primarily interested in the quantity

$$\Delta := \left| \Pr[\mathcal{A}(T_x) = 1] - \Pr[\mathcal{A}(T_y) = 1] \right| \quad \in [0,1]$$

where $T_x \xleftarrow{R} X$ and $T_y \xleftarrow{R} Y$. This quantity captures how well $\mathcal{A}$ distinguishes the distributions X and Y. For a good distinguisher Δ will be close to 1. For a weak distinguisher Δ will be close to 0. The following theorem shows that when n is about $1/(p\epsilon^2)$ then Δ is about $1/2$.

Theorem B.3. *For all $p \in [0,1]$ and $\epsilon < 0.3$, if $n = 4\lceil 1/(p\epsilon^2) \rceil$ then $\Delta > 0.5$*

Proof. The proof follows directly from the Chernoff bound. When T is sampled from X the Chernoff bound implies that

$$\Pr[\mathcal{A}(T_x) = 1] = \Pr[s > p(1+\epsilon)] \leq e^{-n \cdot (p\epsilon^2/2)} \leq e^{-2} \leq 0.135$$

When T is sampled from Y then the Chernoff bound implies that

$$\Pr[\mathcal{A}(T_y) = 0] = \Pr[s < p(1+\epsilon)] \leq e^{-n \cdot (p\epsilon^2/4)} \leq e^{-1} \leq 0.368$$

Hence, $\Delta > |(1 - 0.368) - 0.135| = 0.503$ and the bound follows. □

$$\Pr[\exists i \neq j,\ r_i = r_j] = 1 - \Pr[\forall i \neq j,\ r_i \neq r_j] = 1 - \frac{B-1}{B} \cdot \frac{B-2}{B} \cdots \frac{B-n+1}{B}$$

$$= 1 - \prod_{i=1}^{n-1} \left(1 - \frac{i}{B}\right)$$

$$\geq 1 - \prod_{i=1}^{n-1} e^{-i/B}$$

$$= 1 - e^{-\frac{1}{B}\sum_{i=1}^{n-1} i}$$

$$= 1 - e^{-\frac{n(n-1)}{2B}}$$

Taylor expansion:

$$1 - x \leq e^{-x} = 1 - x + \underbrace{\left(\frac{x^2}{2!} - \frac{x^3}{3!} + \cdots\right)}_{>0}$$

$\Rightarrow$ when $n \simeq 1.2\, B^{1/2}$, Prob. $\geq 1 - e^{-0.72} \simeq 0.513 \geq 0.5$