Cryptography
Week #12:
Quantum Promisses & post-Quantum Cryptography

Rogério Reis, rogerio.reis@fc.up.pt
MSI/MCC/MIERSI - 2025/2026
DCC FCUP

December, 12th 2025

Quantum Computation

All considerations made up to now, were based in a very well known model of computation: **the Von Neuman machine model**.

Although it should be called as **the Turing model!!**

In this model only one instruction (or step of program) is executed at each turn[1] and each instruction only acts over a very small number of memory bits. Although all the huge advances in computer performance and speed, that took place in the last 60 years, the considerations on tractability of computer solutions made in the beginning of the sixties (of the last century) are essentially valid today, needing only to consider dimention of problems one or two orders of magnitude larger.

[1] And parallel and matrix models of execution do not make a big difference to what computational complexity is concern.

The increase in speed in classical computers is directly related to the frequency at which these work. But this frequency at which they work is limited by the energy required to change the state of a "bit", which in turn is directly related to the degree of integration of referred computer.
In other words, **to have more efficient computers quickly we have to make them smaller**...
This size cannot be reduce indefinitely, and some say we are one or two decades beyond an reasonable limit, without starting to feel the serious instability problems that will force us to question the very model of computing.

Reversible Computation

One of the biggest problems with miniaturization of classical computers is that of energy dissipation.
R. Landauer showed, in 1961, that physical limitations resulting from the heat dissipation could be avoided for **almost all** operations performed by a CPU, making them **invertible**.
The exception is for the "delete" operation of one bit, all other operations can be made reversible.

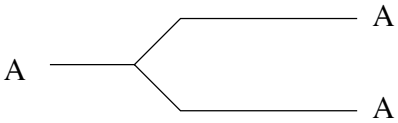
The first condition for a device to be reversible (**logical reversibility**) is that *input* and *output* can be obtained from the other. If Furthermore the device can effectively put to work in the opposite sense then it is said **physically reversible** and the second law of thermodynamics guarantees that does not dissipate energy.

In a quantum computer, programs are evaluated by unitary evolution of an *input* given by the state of the system. As all unitary operators are invertible, we can always "uncompute" the calculation performed.

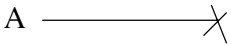
Universal (classical) machines and logic ports

Any calculation on a classical computer can be expressed as a circuit using a set of logic gates. That set of gates must be such that "any" computation can be expressed with it. A set of gates with such characteristic is called **universal set of gates**, as per example: $\{AND, OR, NOT\}$. In fact we can have universal sets of gates with just **two** gates: $\{AND, NOT\}$ or $\{OR, NOT\}$. We can use other ports like XOR, and then $\{XOR, AND\}$ constitutes a universal setof gates. The computational complexity is only affected by **a multiplicative factor** when the set of logic gates in which the programme is express is modified.

Although the previous logic gates are sufficient for any calculation, are not enough to build a computing machine. For a computer to be usable, it is necessary also:



FANOUT



ERASE

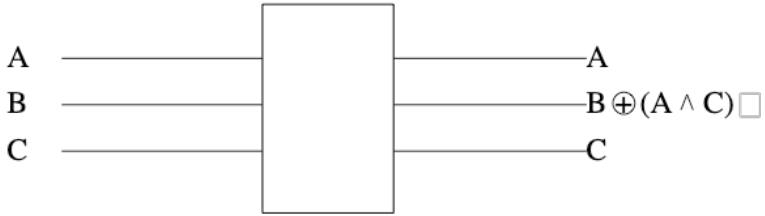
First let us consider the *FANOUT* port. Is it reversible? No information is destroyed, so it is **logically reversible**. *Landauer* showed that can be implemented in a **physically reversible** way.

As for *ERASE*, it is necessary to "clear" the memory of the computer, periodically. There is a type of *ERASE* that can be done in a (logically) reversible way, when there are multiple copies of the information to be erased, and *ERASE* can be computed by "uncomputing" a *FANOUT*. The difficulty appears when you want to delete the last copy of a given piece of information (what we usually call a primitive *ERASE*). In this case an *ERASE*, carried out at a temperature T dissipates $k_B T \ln 2$ (a result known as the *Landauer Principle*).

Fortunately, *ERASE* primitives are not absolutely essential to computation.
To see how, let's consider what we need to evaluate any function using *reversible logic* (hence where primitive *ERASE* are prohibited).
It is possible to calculate the function $f(a)$ by keeping a copy of the *input*:

$$f : a \rightarrow (a, f(a))$$

Let us consider the following port (**Toffoli port**)



where:

$$B \oplus (A \wedge C) = \begin{cases} A \wedge C, & \text{for } B = 0 \qquad (AND) \\ A \oplus B, & \text{for } C = 1 \qquad (XOR) \\ \overline{A}, & \text{for } B = C = 1 \qquad (NOT) \\ A, & \text{to } B \neq C = 1 \quad (FANOUT) \end{cases}$$

this port constitutes, by itself, a **universal set**. It is evidently invertible because applied to itself results in the identity:

$$(B \oplus (A \wedge C)) \oplus (A \wedge C) = B.$$

But since we can't use *ERASE* primitives, the more ports are used, more "useless" bits are accumulated, because in each port we must preserve the *input* bits to preserve its reversibility. So a computer built at the using only reversible logic (instead of conventional one) behaves like:

$$f : a \rightarrow (a, j(a), f(a))$$

with lots of "superfluous" bits $j(a)$.
Bennet showed that these extra bits can be reversibly erased" in intermediate stages with a minimum of time costs and memory.

The solution can be seen as:

$$\begin{aligned} f : a &\rightarrow (a, j(a), f(a)) \\ \mathit{FANOUT} : (a, j(a), f(a)) &\rightarrow (a, j(a), f(a), f(a)) \\ f^* : (a, j(a), f(a), f(a)) &\rightarrow (a, f(a)) \end{aligned}$$

where f^* denotes the "uncomputation" of function f.

Elementary quantum notation

A simple quantum system is that of a particle with its two spin-$\frac{1}{2}$. Its base states are, *spin down* $|\downarrow\rangle$ and *spin to up* $|\uparrow\rangle$, which can without represented as the binaries 0 and 1, that is, $|0\rangle$ and $|1\rangle$, respectively. The state of such a particle is described by

$$\phi = \alpha|0\rangle + \beta|1\rangle \text{ , with } \alpha, \beta \in \mathbb{C}$$

that is, a linear **superposition** of the two "base" states of the system. The squares of the complex coefficients $|\alpha|^2$ and $|\beta|^2$ represent the probabilities of finding the particle in the respective states. Generalizing for k spin-$\frac{1}{2}$ particles, we have a base with 2^k states (vectors that generate the Hilbert space) corresponding to the 2^k possible binary *strings* of size k. Therefore the dimension of Hilbert space grows exponentially with k.

Logical gates for quantum bits (qbits)

Let's start with just one **qubit**. If we represent the states $|\downarrow\rangle$ and $|\uparrow\rangle$ (i.e. $|0\rangle$ and $|1\rangle$) by the vectors $\binom{1}{0}$ and $\binom{0}{1}$, respectively, any transformation unit corresponds to a matrix of the form

$$U_\theta \equiv \begin{pmatrix} e^{i(\delta+\frac{\sigma}{2}+\frac{\tau}{2})}\cos(\frac{\theta}{2}) & e^{i(\delta+\frac{\sigma}{2}-\frac{\tau}{2})}\sin(\frac{\theta}{2}) \\ -e^{i(\delta-\frac{\sigma}{2}+\frac{\tau}{2})}\sin(\frac{\theta}{2}) & e^{i(\delta-\frac{\sigma}{2}-\frac{\tau}{2})}\cos(\frac{\theta}{2}) \end{pmatrix}$$

where normally $\delta = \sigma = \tau = 0$. Using this operator we can swap the bits:

$$U_\pi|0\rangle = -|1\rangle \text{ e } U_\pi|1\rangle = |0\rangle$$

The extra signal represents a phase factor that does not affect the ogical operation of the gates.

Another important one-bit gate is $U_{-\frac{\pi}{2}}$ (**Hadamard operator**) that transforms a particle with *spin* down into one with equiprobable overlap of *spin* down and spin up:

$$U_{\frac{\pi}{2}}|0\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle).$$

Let us consider a *string* of k particles of spin-$\frac{1}{2}$ initially with the spin down. If we apply this gate, independently, for each particle we obtain the superposition of all possible binary *strings* of size k:

$$|0\rangle \rightarrow \frac{1}{\sqrt{q}} \sum_{a=0}^{q-1} |a\rangle,$$

with $q = 2^k$. This register formed by these k particles contains the overlap of all integers from 0 to $2^k - 1$.

Suppose we can construct a unitary operator that transforms a pair of strings $|a;0\rangle$ into the pair $|a;f(a)\rangle$, for a given function f. Such operator applied to the previous one superposition of states

$$\frac{1}{\sqrt{q}}\sum_{a=0}^{q-1}|a;0\rangle \rightarrow \frac{1}{\sqrt{q}}\sum_{a=0}^{q-1}|a;f(a)\rangle$$

calculates the value of the function $(f(a)$ in a parallel for **all** values of a. It is this phenomenon that we will call **quantum parallelism**.

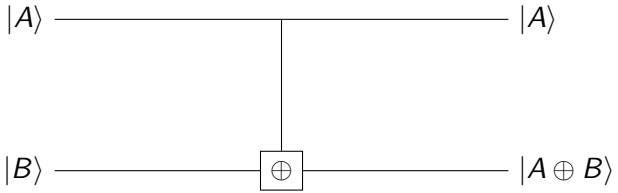
To see how such a unitary operator could be constructed, at the using elementary gates, let's start by constructing the operator *XOR*. For a two-particle system, we have a basis

$$|00\rangle = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \end{pmatrix} \quad |01\rangle = \begin{pmatrix} 0 \\ 1 \\ 0 \\ 0 \end{pmatrix} \quad |10\rangle = \begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \end{pmatrix} \quad |11\rangle = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \end{pmatrix}$$

We can represent the unitary operator corresponding to XOR as

$$U_{XOR} \equiv \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}$$

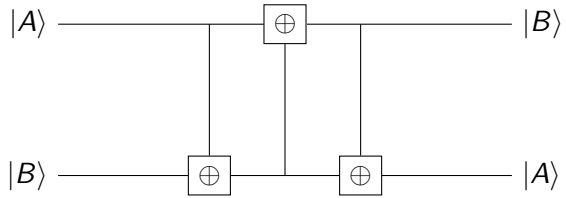
which corresponds to the following quantum circuit:



The first particle acts as a conditional to change the state of the second. The circuit is equivalent to

$$\text{if } (|A\rangle = 1)|B\rangle = \text{not}|B\rangle.$$

The *XOR* gate allows you to move information:



In particular, we show that the *XOR* gate along with some one-bit ports are enough to express any transformation of a finite set of bits.

Quantum Cryptanalysis

Use of quantum parallelism to determine the period of a sequence

Let us consider the sequence

$$f(0), f(1), ..., f(q-1),$$

where $q = 2^k$. We want to use quantum parallelism to determine the period of the function. Let's start with a set of particles with spin down, divided into two groups (two registers formed by **qubits**):

$$|0;0\rangle = |\downarrow,\downarrow,...,\downarrow;\downarrow,\downarrow,...,\downarrow\rangle$$

where the first group has k **qubits** and the second has "the sufficient number of qubits".

To each **qubit** of the first register we apply the Hadamard ($U_{-\frac{\pi}{2}}$) obtaining the **superposition** of all possible integers from 0 to $2^k - 1$:

$$\rightarrow \frac{1}{\sqrt{q}} \sum_{a=0}^{q-1} |a; 0\rangle.$$

The next step consists of decomposition of the calculation of $f(a)$ into a set of unitary 1 bit or two bits operators. This sequence will correspond to $|a; 0\rangle$ the state $|a; f(a)\rangle$ for any *input* a. The number of bits in the second register has be sufficiently large to store the largest value of $f(a)$.

When we apply this sequence of unitary operators to superposition of states from our records, rather than to a *input* simple, we get

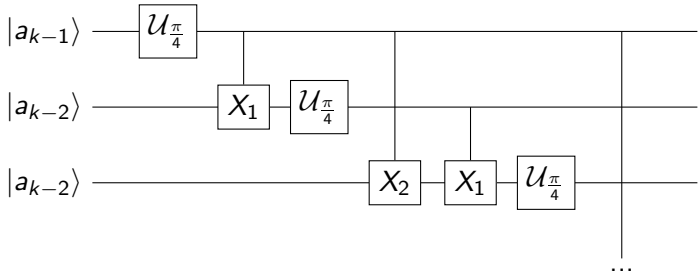
$$\rightarrow \frac{1}{\sqrt{q}} \sum_{a=0}^{q-1} |a; f(a)\rangle.$$

With the same "cost", we compute, at once, the value of the function f for **all** the values of a.

Now let us consider the "quantum" version of the discrete Fourier transformation

$$|a\rangle \rightarrow \frac{1}{\sqrt{q}} \sum_{a=0}^{q-1} e^{2\pi i a c} |c\rangle$$

which is trivially unitary. There exists an efficient way to calculate such a transform with 1 and 2 bit unitary operators:


$|a_{k-1}\rangle$
$|a_{k-2}\rangle$
$|a_{k-2}\rangle$
$\mathcal{U}_{\frac{\pi}{4}}$
X_1
$\mathcal{U}_{\frac{\pi}{4}}$
X_2
X_1
$\mathcal{U}_{\frac{\pi}{4}}$
...

When this transformation is applied to the superposition of states saved in our register of **qubits**, we get

$$\rightarrow \frac{1}{q} \sum_{a=0}^{q-1} \sum_{c=0}^{q-1} e^{\frac{2\pi i a c}{q}} |c; f(a)\rangle$$

The computation is finished, and we can "measure" the spins of the particles that constitute the first register. What will be the result? Suppose that $f(a)$ has period r, that is $f(a+r) = f(a)$. The sumation over a will contain the constructive interference of the coefficients $e^{\frac{2\pi i a c}{q}}$ only when $\frac{c}{q}$ is a multiple of $\frac{1}{r}$.

Therefore, the measured result will be a value of $\frac{c}{q}$. In order to get the period it is enough to repeat this quantum calculation $\log \log \frac{r}{k}$ times to have, with a high probability of one of the multiples be coprime of r, determining it. The algorithm is probabilistic, but this failure probability can be made as small as we want.
All this work seems a bit pointless, just to calculate the period of a function, but this process will support a very strong cryptographic attack.

Shor's algorithm for factoring integers

Suppose we want to factor the number n. If we can find a factor, we reduce the size of the problem and thus the time needed with your calculation. Let's first choose a number x such that $\gcd(x, n) = 1$. Let us consider the sequence

$$x^i \pmod{n}$$

i.e. consider the function

$$f(a) = x^a \pmod{n}.$$

We can use the previous quantum algorithm to calculate the period of such a function.

Let r be the period of f. If r is odd, we just choose a new x and start again.[2]. Then let r be even.

$$\begin{aligned} x^r &\equiv 1 \pmod{n} \\ (x^{\frac{r}{2}})^2 - 1 &\equiv 0 \pmod{n} \\ (x^{\frac{r}{2}} + 1)(x^{\frac{r}{2}} - 1) &\equiv 0 \pmod{n} \end{aligned}$$

i.e. $(x^{\frac{r}{2}} + 1)$ or $(x^{\frac{r}{2}} - 1)$ have a common factor with n, and therefore using the Euclid's algorithm we obtain this factor, and we have n factored.

[2] For a x random the probability of r being even is $\frac{1}{2}$ and therefore, few attempts will be necessary.

This algorithm has a complexity

$$O((\log n)^3)$$

which is much lower than

$$O(e^{\frac{64}{9}^{\frac{1}{3}}(\ln n)^{\frac{1}{3}}(\ln \ln n)^{\frac{2}{3}}})$$

which is the best result for conventional algorithms.

Quantum cryptography
- Bennet & Brassard's protocol

Quantum computing is, for now, nothing more than a theoretical model of a possible cryptographic attack, or at most of a promise (or a menace) in this field. Until now it has not been possible to build quantum computing with much more than a few (hundreds) *qubits* stable for just a few minutes, whereas we need to keep millions of qubits stable for weeks to break any crypto. Also, qubits must be kept at extremely low temperatures (close to absolute zero) to remain stable. Even at freezing temperatures, the state of qubits decays, and they eventually become useless. As of this writing, we don't yet know how to make qubits that last for more than a couple of seconds (their coherence time). Another challenge is that the environment, such as heat and magnetic fields, can affect the qubits' states and lead to computation errors. In theory, it's possible to correct these errors, but it's difficult to do so. Correcting qubits' errors requires quantum error-correcting codes, which in turn require many additional qubits and a low enough rate of error."
The results in cryptography are already a reality, with practical applications and comercial existence.

One way of transmitting information can be through the orientation of a photon's wave. We can transmit a 1 whit a photon with a wave axis $\uparrow$ and to transmit a 0 a photon with wave axis $\rightarrow$. For the receiver to "read" this information, he only needs to use a polarizer filter. If he chooses to set this filter in a $\uparrow$ orientation, and the photon passes the polarizer it means that it was a 1 that was supposed to be transmited, otherwise he can conclude that was a 0. In exactly the same way **Alice** could encode her message transmitting a $\nearrow$ photon for a 1 and a $\nwarrow$ photon for a 0.

What happens if **Alice** sends a 1 in this new base $\times$ (a photon $\nearrow$) and **Bob** make a mistake and try to "read" it with a base polarizer $+$?
If **Bob** uses a polarizer $\uparrow$ (or $\rightarrow$) has a probability of $\frac{1}{2}$ of "reading" a 1 and the same probability for a 0, and has no way to realize that he is not receiving the information that **Alice** is transmiting.

This is the basis of the **BB** protocol

- **Alice** generates two pseudo-random sequences of bits r_1 and r_2.

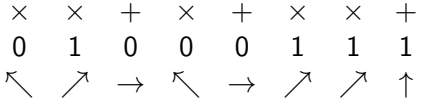
$$r_1 = 11010110$$

$$r_2 = 01000111$$

- Interprets the first sequence as the sequence of bases to be use.

1	1	0	1	0	1	1	0
$\times$	$\times$	$+$	$\times$	$+$	$\times$	$\times$	$+$

- Encodes the second sequence using the orientation bases obtained by first.



- **Alice** then sends **Bob** the message like this encoded.

- **Bob** cannot read **Alice**'s message because he doesn't know r_1 and therefore does not know which bases to use to "read" each one of the photons. It then also generates a pseudo-random sequence bit r'

$$r' = 00010101$$

and "try to read" the photons with a sequence of bases given by this sequence

$$\begin{array}{cccccccc} 0 & 0 & 0 & 1 & 0 & 1 & 0 & 1 \\ + & + & + & \times & + & \times & + & \times \end{array}$$

- As the probability of each of the bits of r' being equal to respective bit of r_1 is just $\frac{1}{2}$, this is proportion of bits read correctly. **Bob** could "read" something like

written	$\times$	$\times$	$+$	$\times$	$+$	$\times$	$\times$	$+$
	0	1	0	0	0	1	1	1
	$\nwarrow$	$\nearrow$	$\rightarrow$	$\nwarrow$	$\rightarrow$	$\nearrow$	$\nearrow$	$\uparrow$
read	$+$	$+$	$+$	$\times$	$+$	$\times$	$+$	$\times$
	0	0	**0**	**0**	**0**	**1**	0	1

- **Alice** now sends r_1 over an open channel.
- **Bob** can know which bits of r_1 he had "achieved to guess" and which correspond to a correct reading.
- **Bob** sends r' to **Alice** via the same channel, who finds out which bits were read successfully.
- The sequence of bits read successfully is therefore known to two intervening

$$0001$$

and can be used to determine the polarization bases to use in future communication

$$+++\times$$

If **Eve** was observing the entire process, it would know which bits contain information about the new communication base, but as it was not able to guess neither r_1 nor r' cannot guess which bits of r_2 correspond to the coincidences of the two random first ones. She cannot therefore read the information that will be transmitted coded in this fashion from now on.

Post-Quantum Cryptography

The coined term of **Post-Quantum Cryptography** denotes the cryptogtaphic primitives that are not menaced by the speedup possible to achieve by the quantum parallelism. Several different kinds of cryptography primitives are today proposed with this aim:

- Code-Based Cryptography (e.g. McEliece cryptosystem (1978) based in error-corrected codes)
- Lattice-Based Cryptography (e.g NTRU (1988) based on the "shortest vector problem")
- Non-Comutative Algebra Cryptography (e.g Anshel–Anshel–Goldfeld protocol (1999) based on non-abelian group algebra)
- Linear Automata Cryptography (e.g. R.Tao and S.Chen (1985) based on Linear Finite Automata)
- Multivariate Cryptography (e.g. T.Matsumoto and H.Imai (1988) based on multivariate polinomials)

Lattice-Based Cryptography

Definition (Vector Space)

A real *vector space* V is a subset of $\mathbb{R}^m$ with the property that

$$(\forall v_1, v_2 \in V)(\forall \alpha_1, \alpha_2 \in \mathbb{R})\ \alpha_1 v_1 + \alpha_2 v_2 \in V.$$

Definition (Linear Combinations)

Let $v_1, v_2, \ldots, v_k \in V$. A *linear combination* of this vectors is a vector of the form

$$w = \alpha_1 v_1 + \alpha_2 v_2 + \cdots + \alpha_n v_n \text{ with } \alpha_1, \alpha_2, \ldots, \alpha_n \in \mathbb{R}.$$

The colection of all such linear combinations

$$\{\alpha_1 v_1 + \alpha_2 v_2 + \cdots + \alpha_n v_n \mid \alpha_1, \alpha_2, \ldots, \alpha_n \in \mathbb{R}\},$$

is called the *span* of $\{v_1, v_2, \ldots, v_n\}$.

Definition (Linear Independence)

A set of vectors $v_1, v_2, \ldots, v_k \in V$ is *(linearly) independent* if

$$\alpha_1 v_1 + \alpha_2 v_2 + \cdots + \alpha_n v_n = 0 \implies \alpha_1, \alpha_2, \ldots, \alpha_n = 0.$$

Otherwise it is said to *linearly dependent.*

Definition (Bases)

A *basis* of V is a set o linearly indpendent vectors $v_1, v_2, \ldots, v_k$ with span V.

Proposition

Let $V \subseteq \mathbb{R}^m$ be a vector space.

1. There existes a basis for V.
2. Any two basis for V have the same number of elements. The number of elements in a basis for V is called the *dimension of* V.
3. Let $v_1, v_2, \ldots, v_k$ be a basis for V and let $w_1, w_2, \ldots, w_k \in V$ be another set of vectors in V. Write each w_j as a linear combination of the v_i,

$$\begin{aligned} w_1 &= \alpha_{11} v_1 + \alpha_{12} v_2 + \cdots + \alpha_{1n} v_n, \\ &\vdots \\ w_n &= \alpha_{n1} v_1 + \alpha_{n2} v_2 + \cdots + \alpha_{nn}. \end{aligned}$$

Then $w_1, w_2, \ldots, w_k$ is also a basis for V if and only if the determinant of the corresponding matrix is not null.

Definition

Let $v, w \in V \subseteq \mathbb{R}^m$ and write v and w using coordinates as

$$v = (x_1, x_2, \ldots, x_m) \qquad w = (y_1, y_2, \ldots, y_m).$$

The *dot product of* v and w is the quantity

$$v \cdot w = x_1 y_1 + x_2 y_2 + \cdots + x_m y_m.$$

We say that v and w are orthogonal to one another if $v \cdot w = 0$.
The *length*, or *Euclidean norm*, of v is

$$||v|| = \sqrt{x_1^2 + x_2^2 + \cdots + x_m^2}.$$

Proposition

1. Let θ be the angle between the vectors v and w, where we place the starting point of v and w at the origin. Then
$$v \cdot w = ||v||\,||w|| \cos(\theta),$$
2. (Cauchy-Schwartz inequality)
$$|v \cdot w| \leq ||v||\,||w||.$$

Definition (Orthogonal Basis)

An *orthogonal* basis for a vector space V is a basis $v_1, \ldots, v_n$ such that

$$(\forall i, j)(i \neq j) \implies (v_i \cdot v_j = 0).$$

The basis is *ortonormal* if in addition, $||v_i|| = 1$, for all i.

Theorem (Gram-Scmhidt algorithm)

Let $v_1, \ldots, v_n$ be a basis for a vector space $V \subseteq \mathbb{R}^m$. The following algorithm creates an orthogonal basis $v_1^*, \ldots, v_n^*$ for V:

1: $v_1^* \leftarrow v_1$
2: **for** $i \leftarrow 2 \ldots n$ **do**
3: **for** $j \leftarrow 1 \ldots i-1$ **do**
4: $\mu_j = v_i \cdot v_j^* / ||v_j^*||^2$
5: $v_i^* \leftarrow v_i - \sum_{j=1}^{i-1} \mu_j v_j^*$.

The two bases are such that

$$(\forall i)\ Span(v_1, \ldots, v_n) = Span((v_1^*, \ldots, v_n^*).$$

Lattices: definition and properties

Definition (Lattice)

Let $v_1, \ldots, v_n \in \mathbb{R}^m$ be a set of lineatly independent vectors. The *Lattice L generated by* $v_1, \ldots, v_n$ is

$$L = \{a_1 v_1 + a_2 v_2 + \cdots + a_n v_n \mid a_1, a_2, \ldots, a_n \in \mathbb{Z}\}.$$

A basis for L is any set of independent vectors that generates L.

Proposition

Any two bases for a lattice L are related by a matrix having integer coefficients and determinant equal to ± 1.

Definition (Integral Lattice)

An integral (or integer) lattice is a lattice all of whose vectors have integer coordinates, thus a subgroup of $\mathbb{Z}^m$ for some $m \geq 1$.

Definition

A subset $L \subseteq \mathbb{R}^m$ is an *addictive subgroup* if it is closed under addition and subtraction. It is called a *discrete addictive subgroup* if there is a positive constant $\varepsilon > 0$ s.t.

$$L \cap \{w \in \mathbb{R}^m \mid ||v - w|| < \varepsilon\} = \{v\}.$$

Theorem

A subset of $\mathbb{R}^m$ is a lattice if and only iff it is a discrete addictive subgroup.

Definition (Fundamental domain)

Let L be a lattice of dimention n and let $v_1, \ldots, v_n$ be a basis of L. The *fundamental domain* for L corresponding to this basis is the set

$$\mathcal{F}(v_1, \ldots, v_n) = \{t_1 v_1 + t_2 v_2 + \cdots + t_n v_n \mid 0 \leq t_i < 1\}.$$

Proposition

Let $L \subseteq \mathbb{R}^n$ be a lattice of dimension n and let $\mathcal{F}$ be a fundamental domain for L. Then every vector $w \in \mathbb{R}^n$ can be written in the form

$$w = t + v \text{ for a unique } t \in \mathcal{F} \text{ and a unique } v \in L.$$

Definition (Determinant of a fundamental domain)

Let L be a lattice of dimension n and let $\mathcal{F}$ be a fundamental domain of L. Then the n-dimention volume of $\mathcal{F}$ is called the *determinant* of L and denoted by $\det L$.

Proposition (Hadamard's Inequality)

Let L be a lattice, take any basis $v_1, \ldots, v_n$ for L, and let $\mathcal{F}$ be a fundamental domain for L. Then

$$\det L = Vol(\mathcal{F}) \leq ||v_1||\,||v_2|| \cdots ||v_n||.$$

Proposition

Let $L \in \mathbb{R}^n$ be a lattice of dimension n, let $v_1, \ldots, v_n$ be a basis for L, and let $\mathcal{F} = \mathcal{F}(v_1, \ldots, v_n)$ be the associated fundamental domain. Write the coordinates of the i^{th} basis vector as

$$v_i = (r_{i1}, r_{i2}, \ldots, r_{in})$$

and use the coordinates of the v_i as rows of a matrix.

$$F = F(v_1, \ldots, v_n) = \begin{pmatrix} r_{11} & r_{12} & \cdots & r_{1n} \\ r_{21} & r_{22} & \cdots & r_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ r_{n1} & r_{n2} & \cdots & r_{nn} \end{pmatrix}.$$

Then the volume of $\mathcal{F}$ is given by

$$Vol(\mathcal{F}(v_1, \ldots, v_n)) = |\det F(v_1, \ldots, v_n)|.$$

Corollary

Let $L \subseteq \mathbb{R}^n$ be a lattice of dimention n. Then every fundamental domain for L has the same volume. Hence $\det L$ *is an invariant of the lattice L, indeendent of the particular fundamental domain used to compute it.*

Two fundamental lattice problems

The Shortest Vector Problem (SVP): Find a shortest nonzero vector in a lattice L, i.e., find a nonzero vector $v \in L$ that minimises the euclidean norm $||v||$.

The Closest Vector Problem (CVP): Given a vector $w \in \mathbb{R}^m$ that **ins not** in L, find a vector $v \in L$ that is the closest to w, i.e., find a vector $v \in L$ that minimises the euclidean norm $||w - v||$.

Theorem (Hermite's Theorem)

Every lattice L of dimension n contains a nonzero vector $v \in L$ satisfying

$$||v|| \leq \sqrt{n}(\det L)^{\frac{1}{n}}.$$

For a given dimension n, Hermite's constant γ_n is the smallest value such that every lattice L of dimension n contains a nonzero vector $v \in L$ satisfying

$$||v||^2 \leq \gamma_n(\det L)^{\frac{2}{n}}.$$

The previous theorem states that $\gamma_n \leq n$. The exact value of γ_n is on known for $n = 1, 2, 3, 4, 5, 6, 7, 8, 24$.

Definition (Hadamard ratio)

We define the *Hadamard ratio of the basis* $\mathcal{B} = \{v_1, \ldots, v_n\}$ of a lattice L to be

$$\mathcal{H} = \left(\frac{\det L}{||v_1||\,||v_2|| \cdots ||v_n||} \right)^{\frac{1}{n}}.$$

Definition (Closed Ball)

For any $a \in \mathbb{R}^n$ and any $R > 0$, the (closed) *ball of radius R centered in a* is the set

$$\mathbb{B}_R(a) = \{x \in \mathbb{R}^n \mid ||x - a|| \leq R\}.$$

Definition

Let S be a subset of $\mathbb{R}^n$.

1. S is *bounded* if there is R such that $S \subseteq \mathbb{B}_R(0)$.
2. S is *symmetric* if for every point $a \in S$, $-a \in S$.
3. S is *convex* if for all $a, b \in S$ the entire segment connecting a and b lies in S.
4. S is *closed* if $a \in \mathbb{R}^n$ is such that every $\mathbb{B}_R(a) \cap S \neq \emptyset$, then $a \in S$.

Theorem (Minkowski's Theorem)

Let $L \subseteq \mathbb{R}^n$ be a lattice of dimension n and let $S \subseteq \mathbb{R}^n$ be a symmetric convex set whose volume satisfies

$$Vol(S) > 2^n \det L.$$

Then S contains a nonzero lattice vector.
If S is also closed, then it suffices to take $Vol(S) \geq 2^n \det L$.

Babai's algorithm

Theorem (Babai's Closest Vertex Algorithm)

Let $L \subseteq \mathbb{R}^n$ be a lattice with basis $v_1, \ldots, v_n$, and let $w \in \mathbb{R}^n$ an arbitrary vector. If the vectors in the basis are "sufficiently orthogonal" to another, then the following algorithm solves CVP.

1: $w = t_1 v_1 + t_2 v_2 + \cdots + t_n v_n$ *with* $t_1, t_2, \ldots, v_n \in \mathbb{R}$.
2: $a_i \leftarrow \lfloor t_i \rceil$ *for* $i = 1, 2, \ldots, n$.
3: **return** $a_1 v_1 + a_2 v_2 + \cdots + a_n v_n$.

If the basis vectors are highly nonorthogonal, then the vector returned by the algorithm is generally far from the closest lattice vector to w.

The Goldreich, Goldwasser and Halevi (GGH) cryptosystem

The scheme is based in the CVP.

Key generation: **Alice** chooses a "good basis" $v_1, \ldots, v_n$ and a integer matrix U s.t. $\det U = \pm 1$. Computes a public "bad" basis $w_1, \ldots, w_n$ resulting from the rows of $W = Uv$.

Encryption: **Bob** having a small plaintext m as a vector with coordinates $(x_i)_i$, chooses a random small vector r. Using the public key computes

$$c = x_1 w_1 + x_2 w_2 + \cdots + x_n w_n + r.$$

Sends e to **Alice**.

Decryption: Alice, uses gets $m' = U^{-1} w$ and applying Babai's algorithm, recovers m.

An example I

For a lattice with the "toy" dimension 3, let the "good" basis be

$$v_1 = (-97, 19, 19), \quad v_2 = (-36, 30, 86), \quad v_3 = (-184, -64, 78).$$

The lattice L spanned by v_1, v_2, v_3 has determinant $\det L = 859516$, and the Hademard ratio of the basis is

$$\mathcal{H}(v_1, v_2, v_3) = \left(\frac{\det L}{||v_1|| \, ||v_2|| \, ||v_3||} \right)^{\frac{1}{3}} \approx 0.74620.$$

Alice multiplies her private basis by

$$U = \begin{pmatrix} 4327 & -15447 & 23454 \\ 3297 & -11770 & 17871 \\ 5464 & -19506 & 29617 \end{pmatrix},$$

An example II

which has determinant $\det U = -1$, to create her public basis

$$\begin{aligned} w_1 &= (-4179163, -1882253, 583183), \\ w_2 &= (-3184353, -1434201, 444361), \\ w_3 &= (-5277320, -2376852, 736426). \end{aligned}$$

The Hadamard ratio of the public basis is

$$\mathcal{H}(w_1, w_2, w_3) \approx 0.0000208.$$

If **Bob** decides to send to **Alice** the message $m = (86, -35, -32)$ using the perturbation $r = (-4, -3, 2)$, the corresponding cyphertext is

$$\begin{aligned} c &= (86, -35, -32) \begin{pmatrix} 4327 & -15447 & 23454 \\ 3297 & -11770 & 17871 \\ 5464 & -19506 & 29617 \end{pmatrix} + (-4, -3, 2) \\ &= (-79081427, -35617462, 11035473). \end{aligned}$$

An example III

Alice, first, rewrites the message/vector to expressed in her own private basis.

$$v^{-1} = \begin{pmatrix} -97 & 19 & 19 \\ -36 & 30 & 86 \\ -184 & -64 & 78 \end{pmatrix}^{-1} = \begin{pmatrix} -0.009126066298 & 0.003138975889 & -0.001237905984 \\ 0.01514340629 & 0.004735223079 & -0.008909665440 \\ -0.009102797388 & 0.01129007488 & 0.002589829625 \end{pmatrix}$$

And then obtains the encrypted vector computing

$$cv^{-1} = (81878.9728684515, -292300.004181423, 443815.036860280).$$

Taking the approximation of these values we get

$$z = 81879v_1 - 292300v_2 + 443815v_3 = (-79081423, -35617459, 11035471),$$

that expressed in the public base comes

$$z \begin{pmatrix} -4179163 & -1882253 & 583183 \\ -3184353 & -1434201 & 444361 \\ -5277320 & -2376852 & 736426 \end{pmatrix}^{-1} = 86w_1 - 35w_2 - 32w_3.$$

An example IV

If **Eve** tries to do the same but using the public key and applying Babai's algorithm she would obtain a "plaintext" $(76, -35, -24)$. This is because the approximation to the point for **Alice**, using the private key is 5.3852, while for **Eve**, using the public key, is 472000.


The End