Information, Calcul et Communication

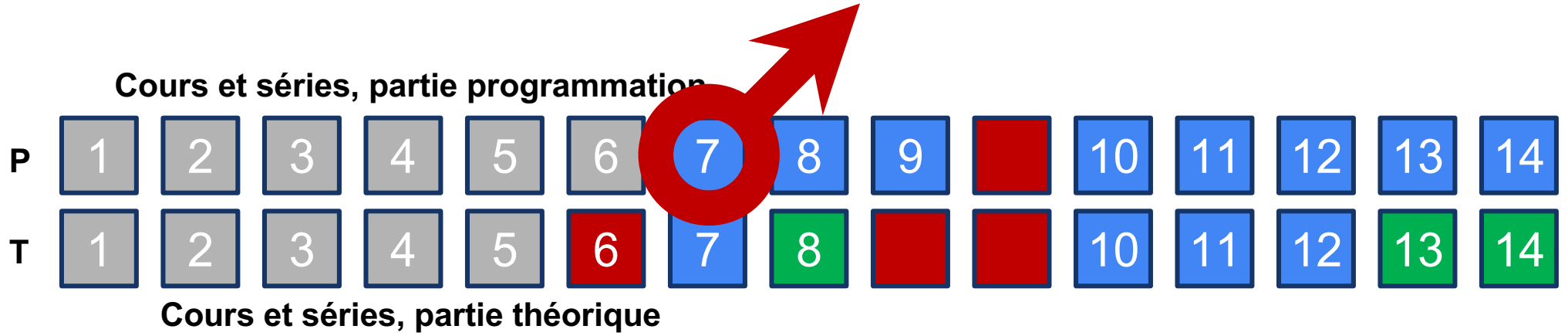
CS-119(k) ICC – Théorie
Semaine 6

Rafael Pires
rafael.pires@epfl.ch

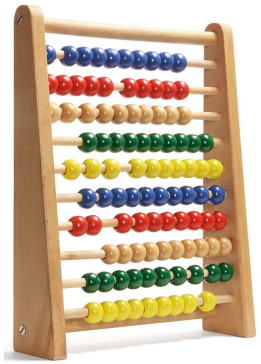
Lausanne, 28.03.2025

Annonces

31.03 ICC-P Cours par zoom (aussi diffusé en salle)
Séance d'exercices normale



Précédemment, dans ICC-T 05



- Dénombrement
- P : Résoudre en temps polynomial
- NP : Vérifier en temps polynomial
- Problèmes d’optimisation discrète

EPFL

Aujourd'hui

- Suite du cours et représentation de l'information
- Représentation binaire des nombres entiers
- Représentation binaire des nombres réels

ICC

Information

Calcul

Communication

- **Algorithmes**
- **Complexité**
- **Récursivité**
- **Algorithmes gloutons / prog. dynamique**
- **Théorie de la calculabilité**

Programme du cours

T | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | | | 10 | 11 | 12 | 13 | 14

Cours et séries, partie théorique

Calcul

Information

Communication

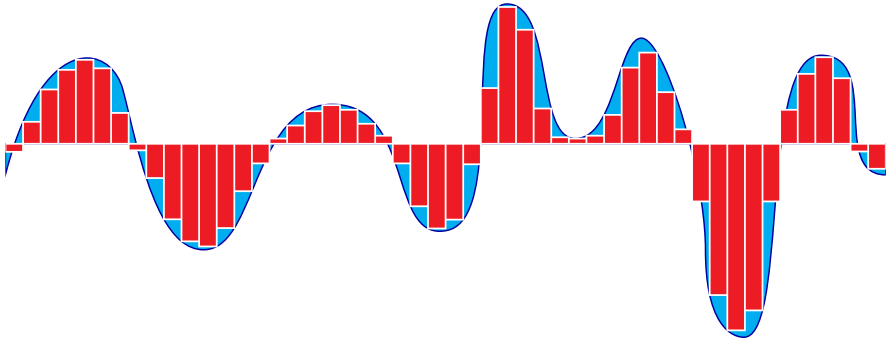
Suite du cours : Motivation

- Supposons que votre meilleur.e ami.e habite en Nouvelle-Zélande, et que vous désiriez lui jouer un sketch pour son anniversaire.
- Il y a 150 ans, vous auriez eu besoin de 80 jours…
- Il y a 50 ans, seuls 2-3 jours auraient suffi…
- Mais aujourd'hui, seules quelques minutes suffisent ! (si on excepte le temps qu'il vous faut pour préparer le sketch)
- Que se passe-t-il exactement pendant ces quelques minutes ?

Suite du cours : Motivation

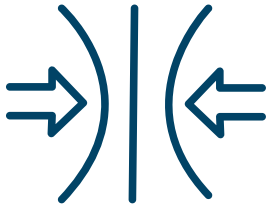
- Avec des amis, vous préparez une vidéo amusante.

- Le son et l'image sont enregistrés au format vidéo :
  - un signal analogique est converti en sa représentation numérique à l'aide d'un algorithme sophistiqué.



Suite du cours : Motivation

- Vous **enregistrez** le fichier en mémoire.
- Vous réduisez sa taille grâce à un algorithme de **compression**, afin que sa **transmission** ne pose pas de problème.

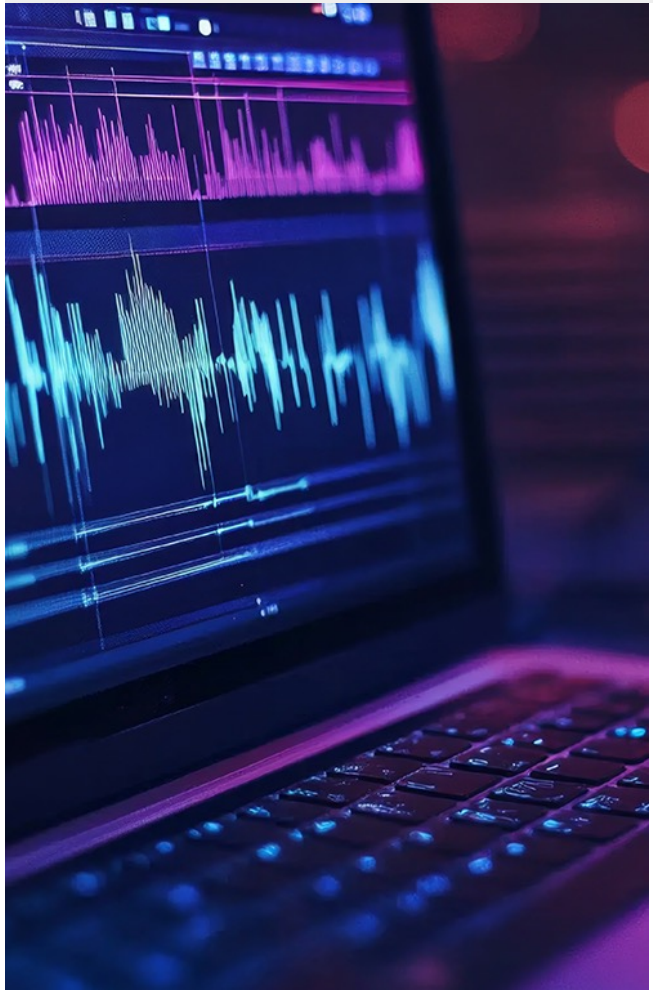


- Vous téléchargez ensuite la vidéo sur votre plateforme préférée :
  - Un algorithme de **chiffrement** est également utilisé pour transmettre les données.

Finalement, votre ami·e découvre la vidéo…

- Les données sont d'abord déchiffrées et décompressées à l'aide d'algorithmes adaptés, puis le signal est reconstruit à partir de leur représentation numérique.

Pour résumer



- Dans nos gestes quotidiens, nous utilisons désormais un grand nombre d'algorithmes sophistiqués, souvent sans nous en rendre compte.
- L'omniprésence de ces algorithmes a changé notre manière de communiquer, de voyager, de voir le monde...
- Plusieurs contributions fondamentales, remontant pour la plupart à plus d'une septantaine d'années, ont permis la réalisation de ces moyens de communication et l'avènement de notre ère digitale.
- Ce sont ces contributions que nous vous proposons de découvrir plus en détail dans les cours qui suivent.

ICC

Information

- Représentation de l'information
  - Nombres entiers et réels
  - Circuits logiques et transistors
- Echantillonnage et reconstruction de signaux
- Entropie et compression de données

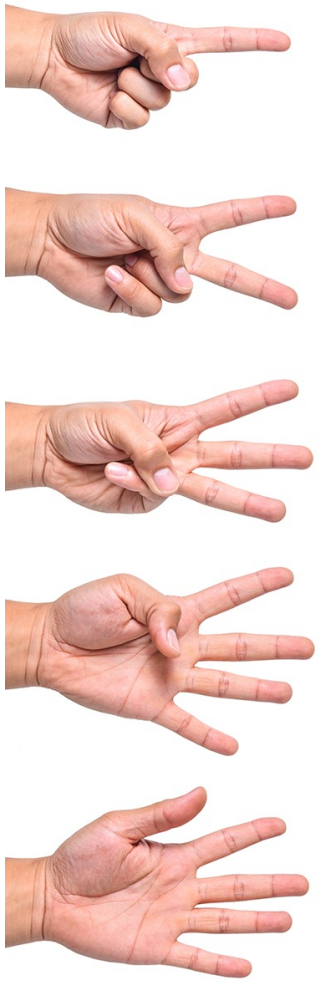
Calcul

- Algorithmes
- Complexité
- Récursivité
- Algorithmes gloutons / prog. dynamique
- Théorie de la calculabilité

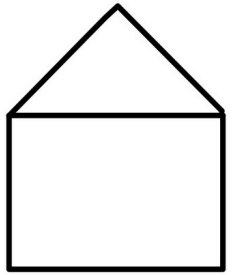
Communication

- Réseaux
- Cryptographie

Représentation de l'information

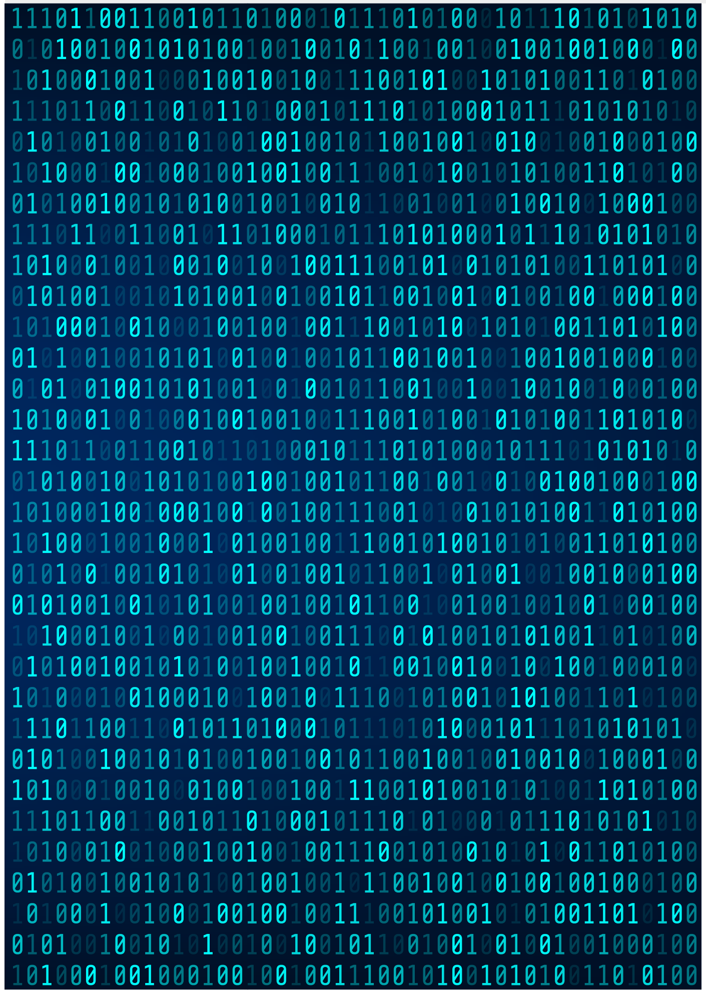


- Il existe plusieurs façons de représenter une information.
  - maison
  - Haus
  - casa
  - chasa
  - house
  - domus



- Mais encore:
- Code Morse :
  - `__ ._ .. ... ___ _.`
- Code ASCII décimal :
  - `77 65 73 83 79 78`
- Code ASCII hexadécimal
  - `4D 41 49 53 4F 4E`
- Code ASCII binaire :
  - `01001101 01000001 …`

Pourquoi choisir la représentation binaire ?



- Réduction à deux symboles (0 et 1) facile à implémenter :

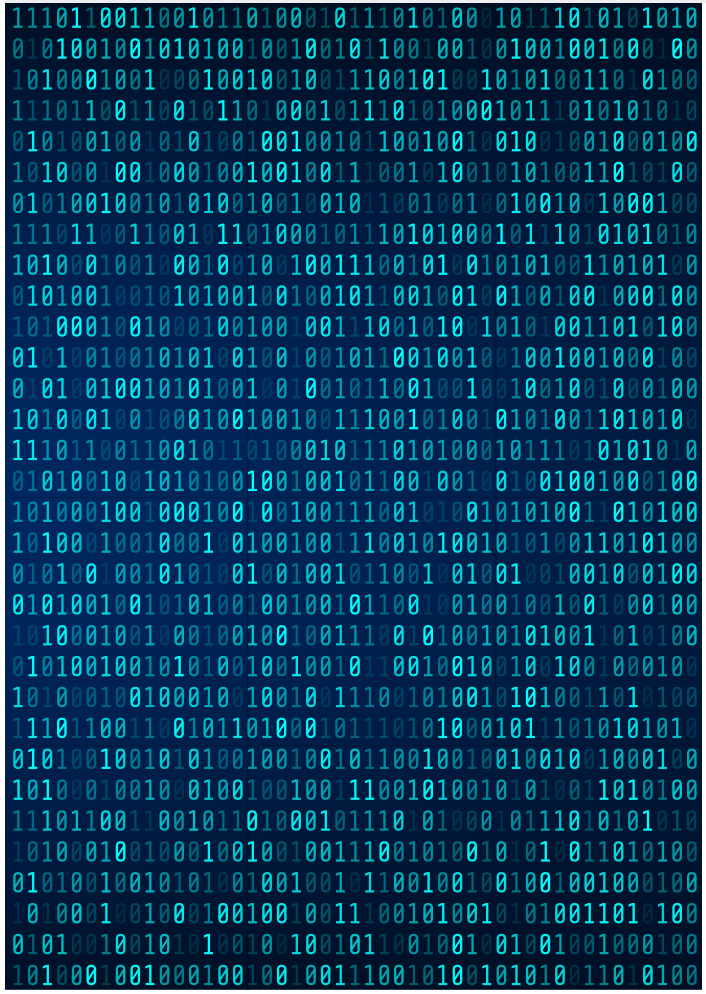
0	1
Circuit ouvert	Circuit fermé
Tension de 0V	Tension de 5V

- Et au fait, pourquoi ne pas choisir un seul symbole, « | » par exemple ?

1 = | 2 = || 3 = ||| 4 = ||||

1’000 = |||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||| …

Pourquoi choisir la représentation binaire ?



- Combien d'éléments peut-on représenter avec n bits ?

$$2^n$$

- Exemples :
  - 1 bit : « noir » (`0`), « blanc » (`1`)
  - 2 bits : est (`00`), ouest (`01`), nord (`10`), sud (`11`)
  - 8 bits : 256 carctères différents (code ASCII étendu)
  - 32 bits : plus de 4 milliards de caractères différents (code UTF-8)

Aujourd'hui

- Suite du cours et représentation de l'information
- Représentation binaire des nombres entiers
- Représentation binaire des nombres réels

Représentation des nombres entiers positifs

- Prenons un exemple de nombre entier positif : 1'984
- Ceci est une représentation ! (la représentation décimale)

$$1'984 = 1'000 + 900 + 80 + 4$$
$$= 1 . 10^3 + 9 . 10^2 + 8 . 10^1 + 4 . 10^0$$

D'autres avant nous auraient écrit : M C M L X X X I V

- Mais on peut aussi écrire :

$$1'984 = 1'024 + 512 + 256 + 128 + 64$$
$$= 1 . 2^{10} + 1 . 2^9 + 1 . 2^8 + 1 . 2^7 + 1 . 2^6 + 0 . 2^5$$
$$+ 0 . 2^4 + 0 . 2^3 + 0 . 2^2 + 0 . 2^1 + 0 . 2^0$$

→ 11111000000 en binaire

Représentation des nombres entiers positifs

- Représentation décimale :

$$\sum_{j=0}^{m-1} c_j \cdot 10^j, m \text{ chiffres } c_j \in \{0,1,\cdots,9\}$$

- Nombre de chiffres nécessaires :

$$m = \lceil \log_{10}(N+1) \rceil$$

- Représentation binaire :

$$\sum_{i=0}^{n-1} b_i \cdot 2^i, n \text{ bits } b_i \in \{0,1\}$$

- Nombre de bits nécessaires :

$$n = \lceil \log_2(N+1) \rceil$$

- Attention ! Avec n bits, on peut representer 2^n nombres entiers différents : de 0 à $2^n - 1$, et donc pas 2^n lui-même !
- Exemple: Avec n = 8 bits, intervalle de 0 à $2^8 - 1 = 255$.

Opérations binaires : multiplication et division

- En binaire, multiplier et diviser par 2 est très facile :

(de même que multiplier et diviser par 10 est facile en décimal)

	0110	6
×	0010	2
	1100	12

	0110	6
÷	0010	2
	0011	3

- Décalage à gauche
- Décalage à droite

Opérations binaires : addition et soustraction

- **Addition :**

$$\begin{array}{rr} & 1\,1\,1 \\ & 0101 \\ + & 0011 \\ \hline & 1000 \end{array} \qquad \begin{array}{r} 5 \\ 3 \\ \hline 8 \end{array}$$

- **Soustraction :**

$$\begin{array}{rr} & 0\,{}^{10} \\ & 0\cancel{1}01 \\ - & 0011 \\ \hline & 0010 \end{array} \qquad \begin{array}{r} 5 \\ 3 \\ \hline 2 \end{array}$$

0+0=0 0+1=1

1+1=10 1+1+1=11

Opérations binaires : dépassement de capacité

- Etant donné la limite imposée par le nombre de bits utilisés, des problèmes de dépassement de capacité (overflow) surviennent lorsqu'on effectue des opérations binaires et que le résultat attendu se trouve en dehors de l'intervalle des nombres représentables.
- n = 4 bits

	Binaire	Décimal
retenues	~~1~~ 111	
	1011	11
+	0111	7
	0010	2

	Binaire	Décimal
retenues	~~1~~ 111	
	1111	15
+	0001	1
	0000	0

- 11 + 7 = 18, en dehors de {0 ... 15}

Nombres négatifs

- Une solution possible :

0 0011 +3
1 0011 -3

↑

bit de signe

```
   11
  0 0011   +3
+ 1 0011   -3
-------   ---
  1 0110   -6
```

Représentation binaire des nombres entiers relatifs

complément à deux

- Avec n bits, on utilise la convention suivante pour représenter les nombres entiers relatifs (positifs et négatifs) :

$$N = -b_{n-1} \cdot 2^{n-1} + \sum_{i=0}^{n-2} b_i \cdot 2^i$$

- Exemple avec 8 bits : N = -43 est représenté par 11010101, car

$$\begin{aligned} -43 &= -128 + 64 + 0 + 16 + 0 + 4 + 0 + 1 \\ &= \mathbf{-1} \cdot 2^7 + \mathbf{1} \cdot 2^6 + \mathbf{0} \cdot 2^5 + \mathbf{1} \cdot 2^4 + \mathbf{0} \cdot 2^3 + \mathbf{1} \cdot 2^2 + \mathbf{0} \cdot 2^1 + \mathbf{1} \cdot 2^0 \end{aligned}$$

Représentation binaire des nombres entiers relatifs

complément à deux

- Le premier bit est le bit de signe
  - `0` : nombre positif
  - `1` : nombre négatif
- Avec 8 bits, les nombres représentables vont de -128 à +127 :
  - -128 = `10000000`
  - +127 = `01111111`
- -1 = `11111111` avec cette représentation
- +128 n'est pas représentable avec 8 bits!

Opérations binaires : opposés

complément à deux

- Comment représenter -17 avec 8 bits ?

1. On part avec 17 = `00010001`
2. On calcule le complément à 1 de ce nombre :

```
  11111111
- 00010001
----------
  11101110
```

3. On ajoute 1 à ce nombre (complément à 2) :

```
  11101111
```

-128 + 64 + 32 + 0 + 8 + 4 + 2 + 1 = -17

Opérations binaires : addition et soustraction

complément à deux

~~1~~ 1 1	
0 101	+5
+ 1 101	-3
0 010	+2 ✅

~~1~~ 1 11	
1 011	-5
+ 1 101	-3
1 000	-8 ✅

~~1~~ 11	
1 011	-5
+ 1 011	-5
0 110	+6 ⚠

Overflow : n=4 : {-8, -7, …, +7}

Aujourd’hui

- Suite du cours et représentation de l’information
- Représentation binaire des nombres entiers
- **Représentation binaire des nombres réels**

Représentation binaire des nombres réels

- Il est impossible de lister tous les nombres réels un par un (ils ne forment pas un ensemble dénombrable).
- Même en se limitant à [0,1], seule une partie de ces nombres peut être représentée : on procède donc à des approximations.
- On doit donc se résigner à accepter la présence d'erreurs. Soit x_{rep} la valeur représentée en binaire du nombre x, on définit:
  - L'erreur absolue :

$$|\Delta x| = |x - x_{rep}|$$

  - L'erreur relative (« précision ») :

$$\frac{|\Delta x|}{|x|}$$

Représentation binaire des nombres réels

- Prenons un nombre entre 0 et 1, $x = 0{,}375$

$$x = 0{,}375 = 0{,}25 + 0{,}125 = \mathbf{0} \cdot 2^{-1} + \mathbf{1} \cdot 2^{-2} + \mathbf{1} \cdot 3^{-3} \rightarrow \mathbf{011}$$

Représentation exacte !

- Autre exemple :

$$y = \pi - 3 = 0{,}141592\cdots = 0.125 + \cdots$$

$$= \mathbf{0} \cdot 2^{-1} + \mathbf{0} \cdot 2^{-2} + \mathbf{1} \cdot 3^{-3} \rightarrow \mathbf{0010}\cdots$$

Représentation approximative !

Représentation binaire des nombres réels

$0{,}375 \times 2 = 0{,}75$, reste $0{,}75$

$0{,}75 \times 2 = 1{,}50$, reste $0{,}50$

$0{,}50 \times 2 = 1{,}00$, reste 0

$\rightarrow 011$

Représentation binaire des nombres réels

$0{,}1 \times 2 = 0{,}2$, reste $0{,}2$

$0{,}2 \times 2 = 0{,}4$, reste $0{,}4$

$0{,}4 \times 2 = 0{,}8$, reste $0{,}8$

$0{,}8 \times 2 = 1{,}6$, reste $0{,}6$

$0{,}6 \times 2 = 1{,}2$, reste $0{,}2$

$0{,}2 \times 2 = 0{,}4$, reste $0{,}4$

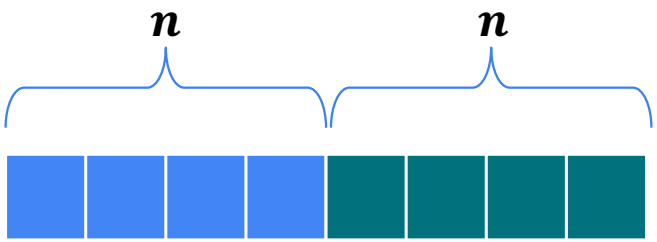
$\rightarrow 0001100110011001100110011\ldots$

Représentation binaire des nombres réels

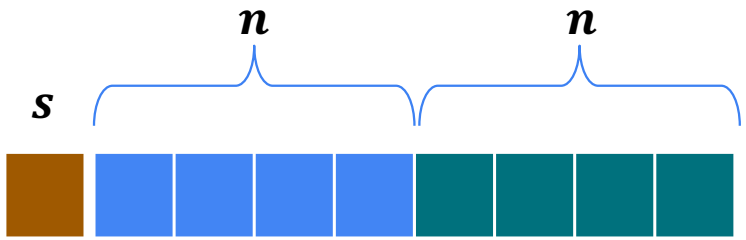
virgule fixe

- Nombres réels plus grand que 0 :

  n bits pour la partie entière, n bits pour la partie décimale



- Nombres réels négatifs : rajouter un bit de signe (s)



Représentation binaire des nombres réels

virgule fixe

- **L'erreur absolue :**

$$|\Delta x| = \left|x - x_{rep}\right| < 2^{-n}$$

- **L'erreur relative (« précision ») :**

$$\frac{|\Delta x|}{|x|}$$

Peut être arbitrairement grande si n est proche de 0.

Représentation binaire des nombres réels

virgule fixe

- L'erreur absolue :
  $|\Delta x| = |0.18 - 0.125| = 0.055$
- L'erreur relative (« précision ») :
  $$\frac{|\Delta x|}{|x|} = \frac{0.055}{0.18} = 30{,}5\%$$

- L'erreur absolue :
  $|\Delta x| = |15.18 - 15.125| = 0.055$
- L'erreur relative (« précision ») :
  $$\frac{|\Delta x|}{|x|} = \frac{0.055}{15.18} = 0{,}36\%$$

Représentation binaire des nombres réels

virgule flottante

- Pour pallier ce problème de précision, on choisit plutôt la représentation suivante :

1. On garde la représentation en virgule fixe avec n bits pour les nombres réels dans l'intervalle [1,2]
   - ❖ L'erreur relative (« précision ») :

$$\frac{|\Delta x|}{|x|} < 2^{-n}, \text{ car } |x| \geq 1$$

2. On réplique cette représentation à toutes les échelles en la multipliant ou en la divisant par des puissances de 2.

Représentation binaire des nombres réels

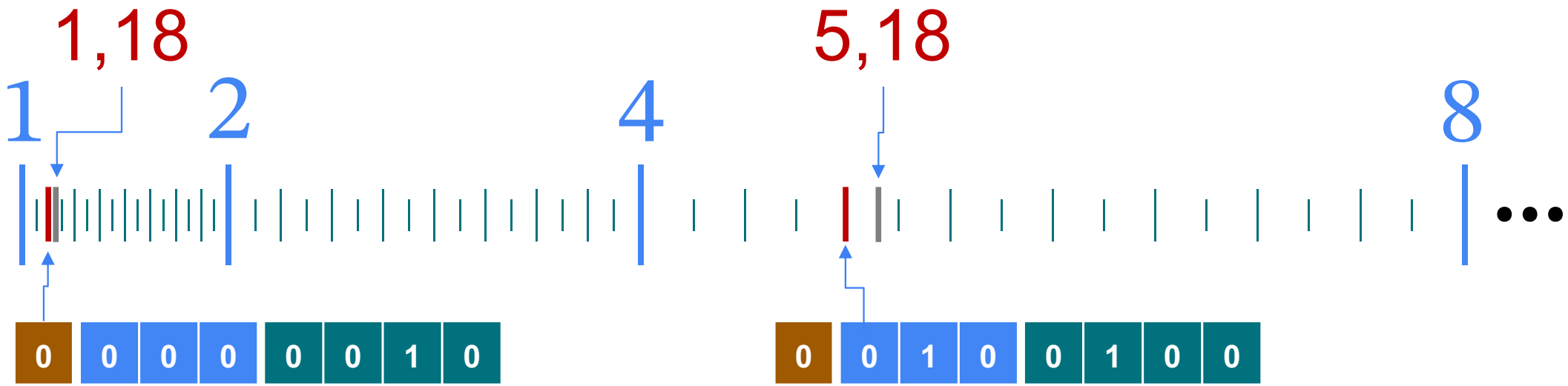
virgule flottante

- Inspiration : Notation scientifique en base 10 avec un nombre fixe de chiffres significatifs.
- Exemples avec 3 chiffres significatifs :
  - 3.1415 s'écrit $3.14 \cdot 10^{0}$
  - 0.0125 s'écrit $1.25 \cdot 10^{-2}$
  - 7354 s'écrit $7.35 \cdot 10^{3}$

$$\text{signe} \cdot 2^{\text{exposant}} \cdot 1,\text{mantisse}$$

Représentation binaire des nombres réels

virgule flottante



- L'erreur absolue :
  $|\Delta x| = |1.18 - 1.125| = 0.055$
- L'erreur relative (« précision ») :
  $\frac{|\Delta x|}{|x|} = \frac{0.055}{1.18} = 4{,}66\%$

- L'erreur absolue :
  $|\Delta x| = |5.18 - 5| = 0.18$
- L'erreur relative (« précision ») :
  $\frac{|\Delta x|}{|x|} = \frac{0.18}{5.18} = 3{,}47\%$

Et le 0 ?

virgule flottante

$$+1 \cdot 2^{0} \cdot 1{,}0$$

0	0	0	0	0	0	0	0

Représentation binaire des nombres réels : Et le 0 ?

virgule flottante

- Les subnormaux (Quand exposant = 0)

$$\text{signe} \cdot 2^{1 - \text{biais}} \cdot 0,\text{mantisse}$$

- Nombres normalisés (Quand exposant ≠ 0)

$$\text{signe} \cdot 2^{\text{exposant} - \text{biais}} \cdot 1,\text{mantisse}$$

Avec biais = $2^{n_e - 1} - 1$ et n_e le nombre de bits qui représentent l'exposant.
Dans notre exemple, biais = $2^{3-1} - 1 = 3$

0 = | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

1 = | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 |

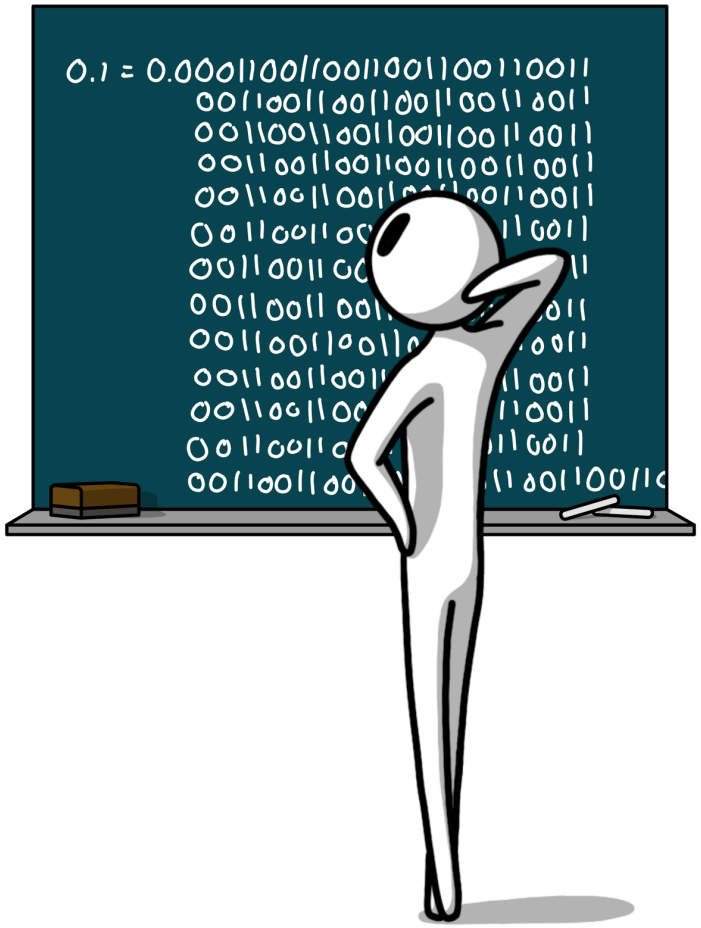
Représentation binaire des nombres réels : Et le 0 ?

virgule flottante

$-\infty$ = 1 1 1 1 0 0 0 0

NaN = 0 1 1 1 0 0 0 1

Conséquence des erreurs d'arrondi



- Tester l'égalité de résultats en virgule flottante est une absurdité.
- Le test d'égalité de valeur de résultats en virgule flottante doit être fait avec tolérance autour de la valeur théorique.

IEEE 754 les plus courantes :

	32 bits	64 bits
Signe	1	1
Exposant	8	11
Mantisse	23	52

- 0.1, par exemple, n'a pas de représentation exacte d'après cette convention
(comme 1/3 n'a pas de représentation exacte en décimal)

Aujourd’hui

- Suite du cours et représentation de l’information
- Représentation binaire des nombres entiers
- Représentation binaire des nombres réels

Résumé Cours 6 – ICC-T

- Perspective du cours : du codage de l'information (nombres, signaux, compression, réseaux, cryptographie) aux circuits physiques (transistors).
- Représentation binaire des entiers :
  - Nombres positifs, négatifs (complément à deux)
  - Opérations (addition, soustraction, multiplication, division)
  - Dépassement de capacité
- Représentation binaire des réels :
  - Virgule fixe : erreur relative inévitable
  - Virgule flottante : signe, exposant, mantisse
  - Astuces pratiques : éviter la comparaison stricte en égalité (ex. 0.1 non représentable)

rafael.pires@epfl.ch

Merci