IA 101

Chapitre 2
Information algorithmique en pratique

Complexité

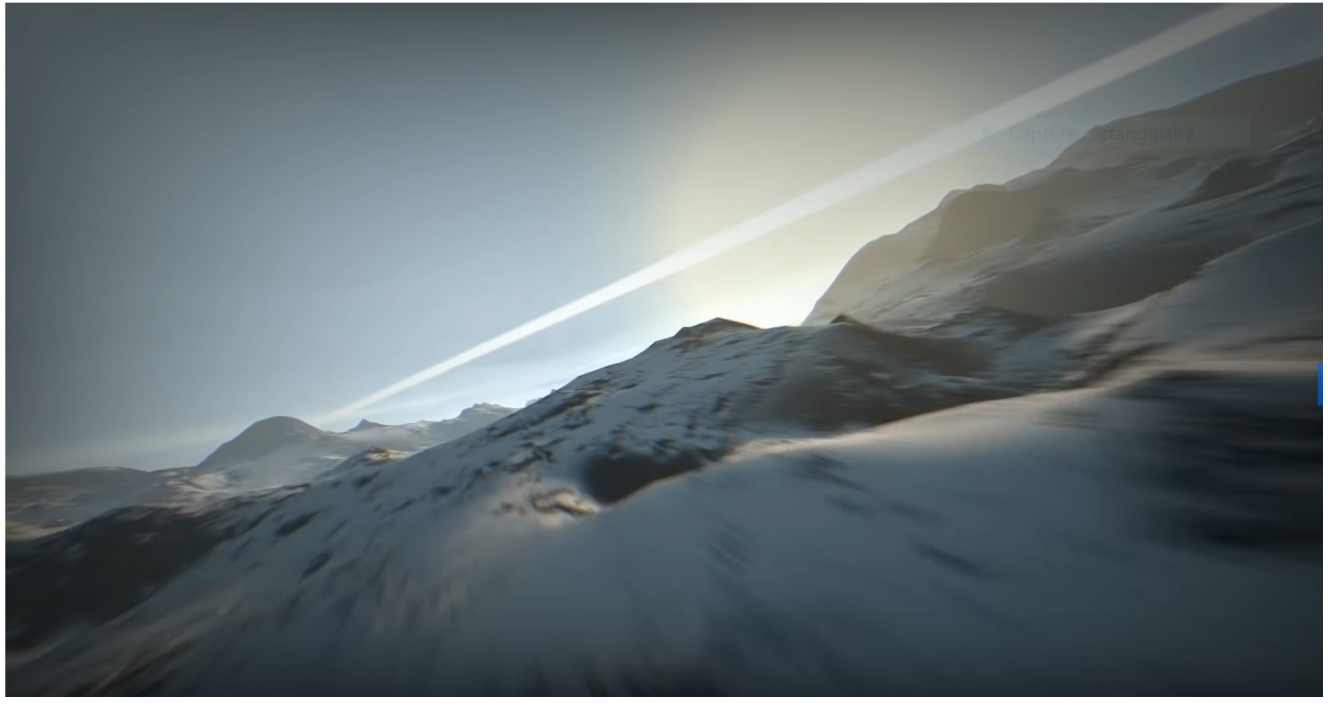
- *La complexité de Kolmogorov n'est pas calculable* (Théorème, preuve dans le chapitre 3)
- En pratique, on peut l'approximer
- L'information algorithme c'est ce qui reste quand toute redondance est factorisée → **compression**

Compression

Quelle taille minimum fait une video de 3min30 en 1080p avec des paysages de montagne réalistes ?

- 4 GB
- 40 MB
- 4 MB
- 400 kB
- 40 kB
- 4 kB

Compression



1st place in pc 4k compo at Breakpoint 2009

https://www.youtube.com/watch?v=jB0vBmiTr6o

Compression de séquences binaires

- Certaines séquences binaires sont très compressibles
  - 0000000...000
  - 010101010101...010101
- Quelle proportion de séquences binaires de taille 100 est compressible de 10 bits ?
  - 10%
  - 1%
  - 0.1%
  - 0.01%
  - 0.001%

Théorème d'incompressibilité

- Quelle est la proportion de séquences binaires de *n* bits qui peuvent être compressées de *k* bits ?
- Soit un compresseur qui compresse de *k* bits
- Une séquence compressée est représentée par *n-k* bits
- Il y en a au plus 2^{n-k}
- La proportion est d'au plus $\frac{2^{n-k}}{2^n} = 2^{-k}$

- Par exemple je veux compresser les séquences de *n-1* bits, je peux le faire pour au plus $2^{n-(n-1)} = 2$

Théorème d’incompressibilité

- On ne peut pas tout compresser ; pas de compression universelle (paradoxe du compresseur)
- Cf la compression d’entiers : facile de compresser un ensemble fini, pas facile de compresser un ensemble infini
- Si on compresse un objet, il y en a un autre qui est dilaté

Théorème d’incompressibilité

- La compression iterative mène à
  - Une séquence périodique, ou
  - Une séquence de longueur non bornée
- On peut compresser 1 fois 1000 objets, mais on ne peut pas compresser 1000 fois 1000 objets. Non c’est pas ça...

Complexité — similarité

- Pas de compression universelle
- Mais une bonne compression exploite les répétitions
- Soit Z un compresseur (e.g. zip), comparer:
  - Z(A)
  - Z(B)
  - Z(A + B)
- Si A et B similaires, $Z(A) \approx Z(A + B)$

Complexité — similarité

- On veut
  - $C(x|y) \approx 0$ si *x* et *y* similaires
  - $C(x|y) \approx C(x)$ sinon
- $\max(C(x|y), C(y|x))$ mesure la distance *x* — *y*
- $\dfrac{\max(C(x|y), C(y|x))}{\max(C(x), C(y))}$ normalisé entre 0 et 1

Complexité — similarité

- $\frac{\max(C(x|y), C(y|x))}{\max(C(x), C(y))}$ normalisé entre 0 et 1

- $C(x|y)$ difficile à approximer
- $C(x, y) \leq C(y) + C(x|y) + O(1)$
  $\rightarrow C(x, y) - C(y)$

- $\mathrm{NID}(x, y) = \frac{C(x, y) - \min(C(x), C(y))}{\max(C(x), C(y))}$

Normalized Information Distance

Exercice : vérifier que c'est bien une distance

Normalized compression distance

$$\mathrm{NID}(x,y) = \frac{C(x,y) - \min(C(x), C(y))}{\max(C(x), C(y))}$$

- Problème : calcul de *C*
- Solution:

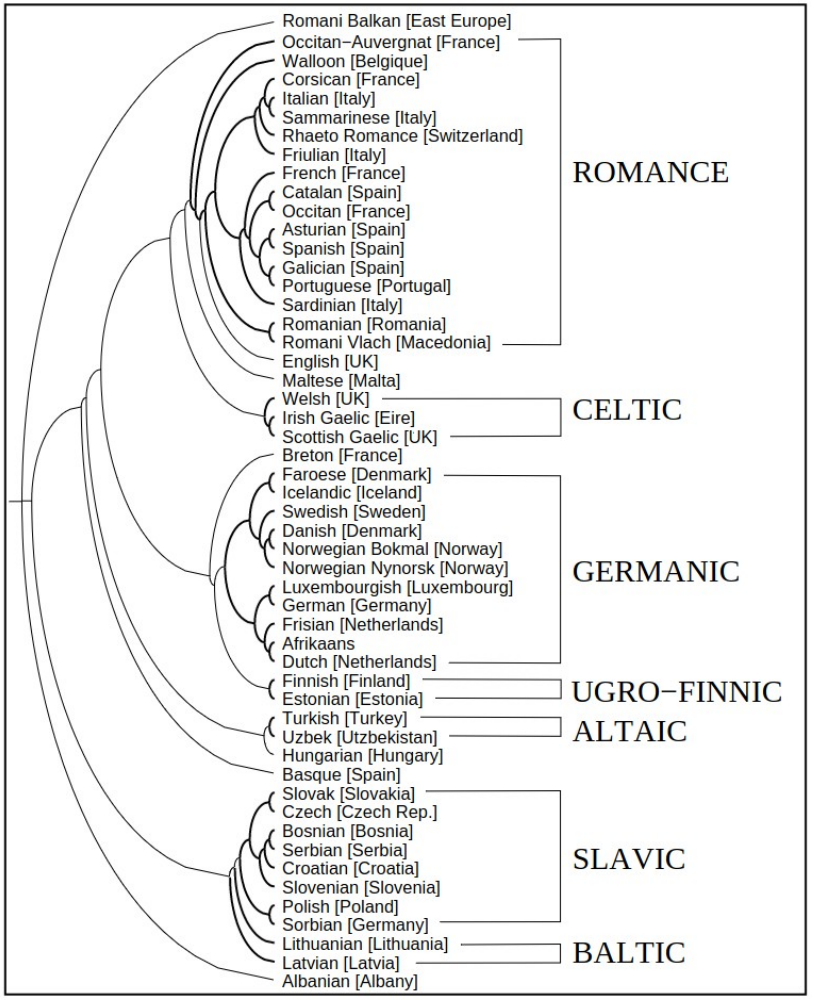
$$\mathrm{NCD}(x,y) = \frac{Z(x,y) - \min(Z(x), Z(y))}{\max(Z(x), Z(y))}$$

- Le compresseur nous donne le programme minimal d’un objet

Normalized compression distance

$$\mathrm{NCD}(x,y) = \frac{Z(x,y) - \mathrm{m}}{\max(Z}$$

- Ca fonctionne très bien
- Cilibrasi and Vitányi, *Fast Phy*
  CoV-2 by Compression, Entro
- Benedetto et al, *Language tre*
  Physical Review Letters 2002



Normalized compression distance

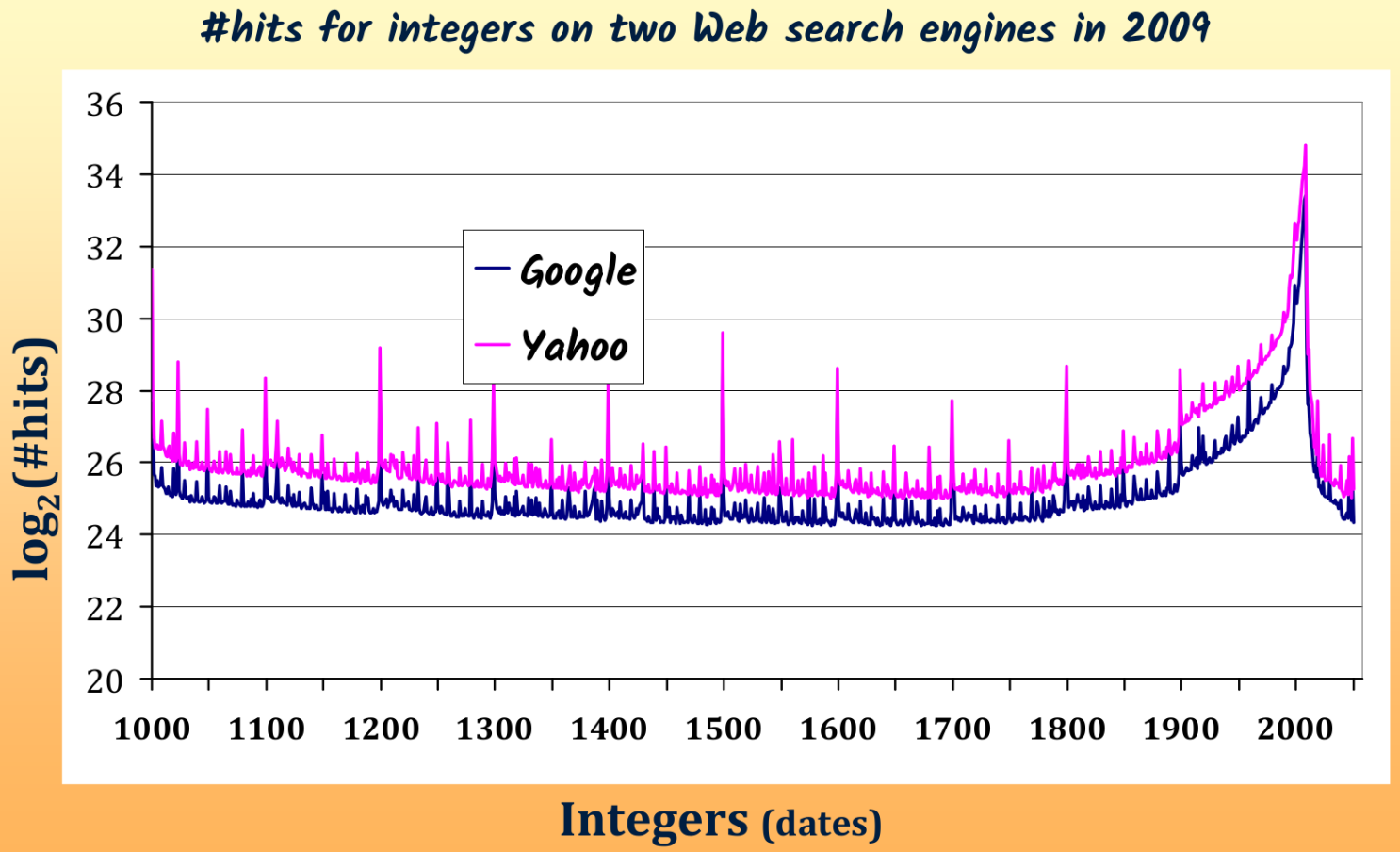
$$\mathrm{NCD}(x,y) = \frac{Z(x,y) - \min(Z(x), Z(y))}{\max(Z(x), Z(y))}$$

- Ca fonctionne très bien
- ...mais ça n’utilise que l’information textuelle d’un objet
- Pas le sens des mots
- Pas les similarités conceptuelles

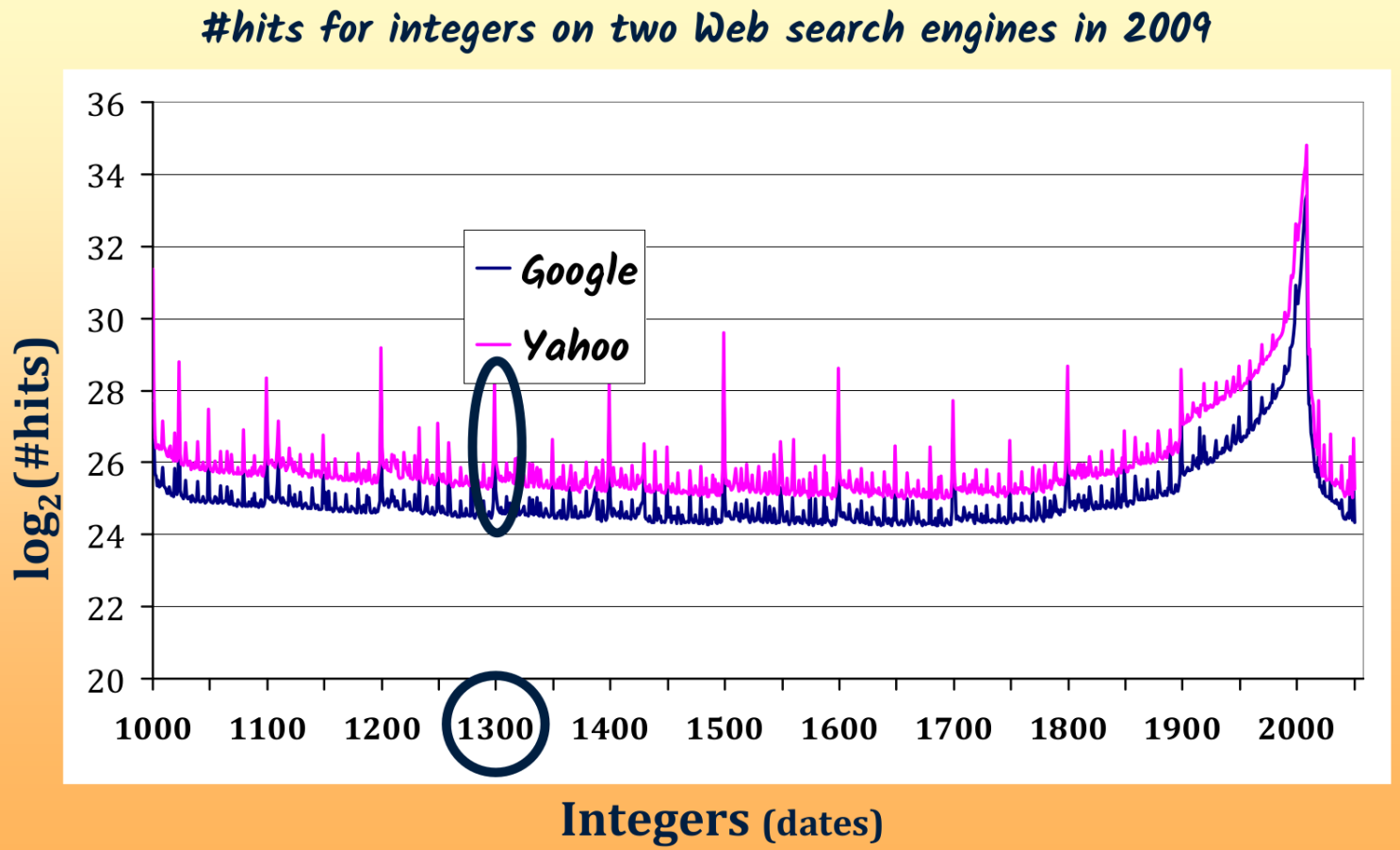
Complexité et fréquence

- Les objets simples sont
  - Plus fréquents ?
  - Moins fréquents ?

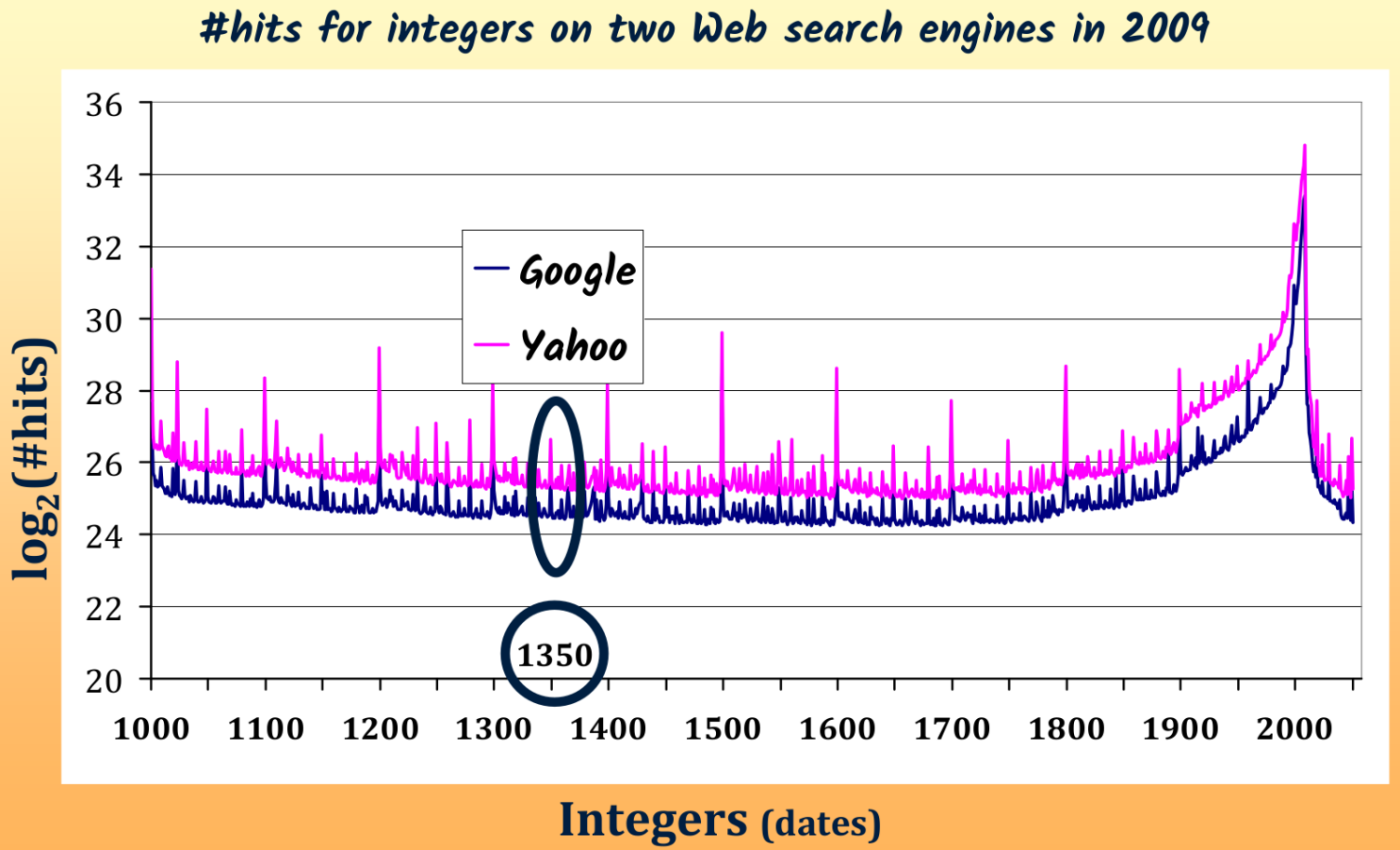
Complexity and Frequency



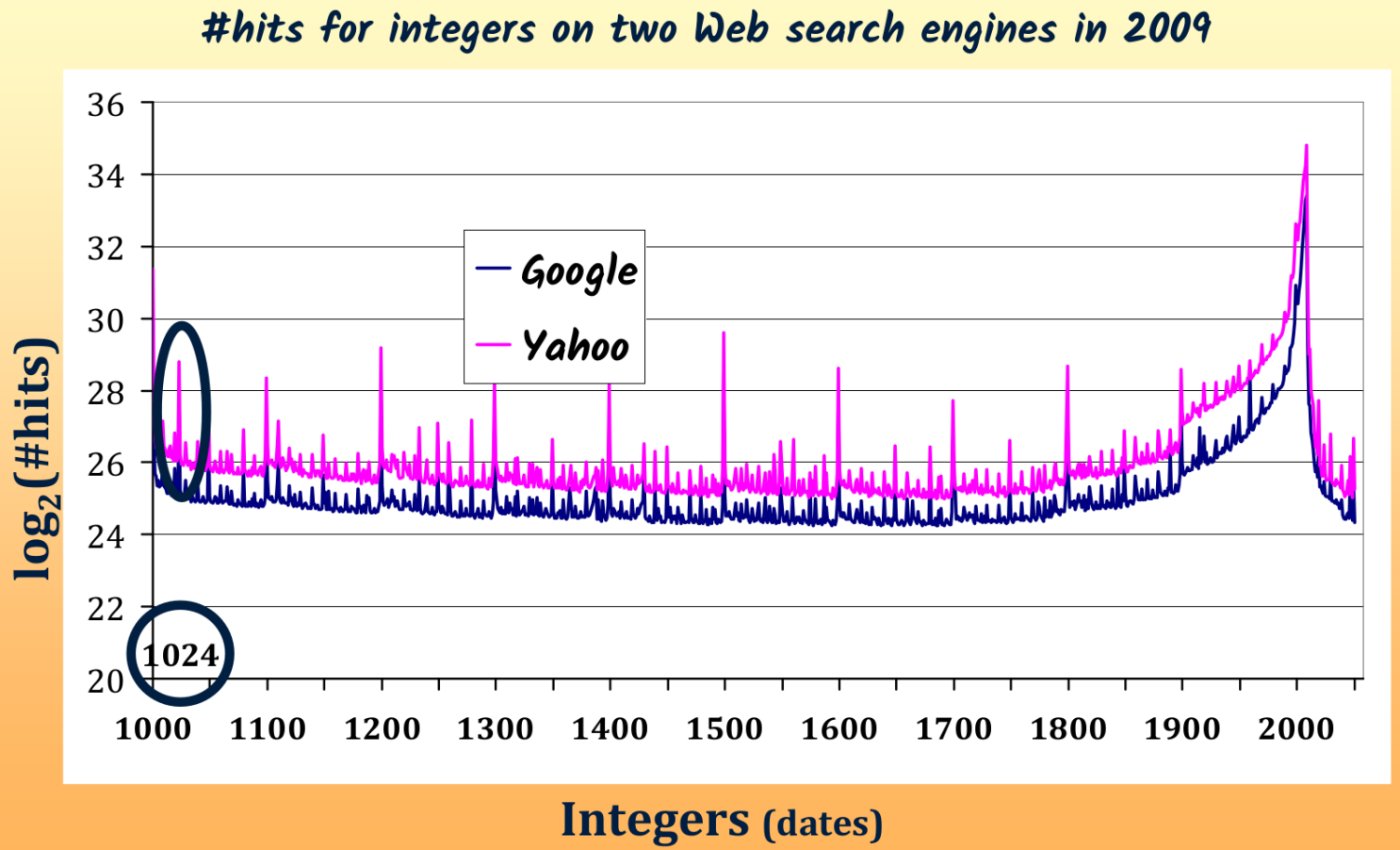
Complexity and Frequency



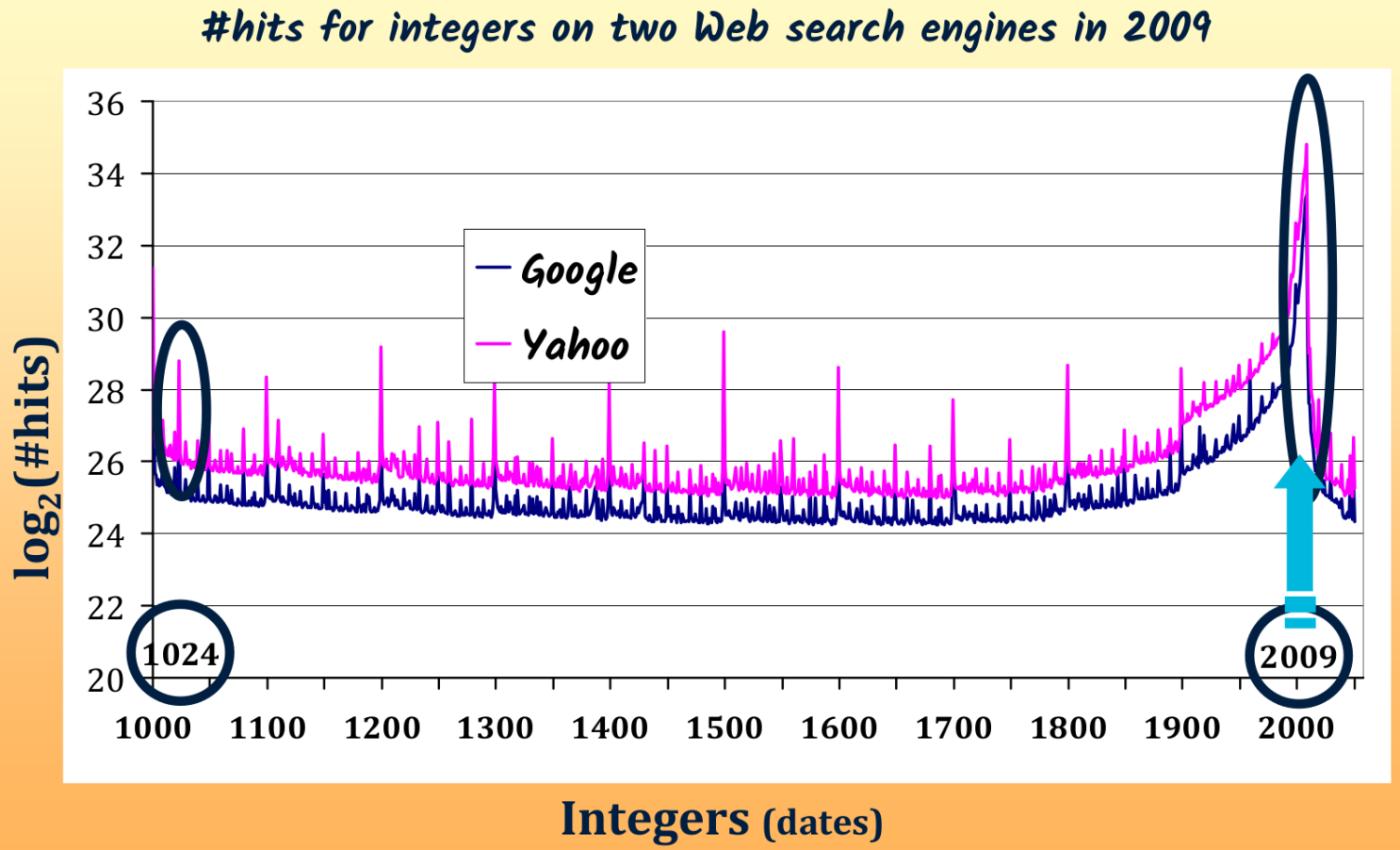
Complexity and Frequency



Complexity and Frequency



Complexity and Frequency

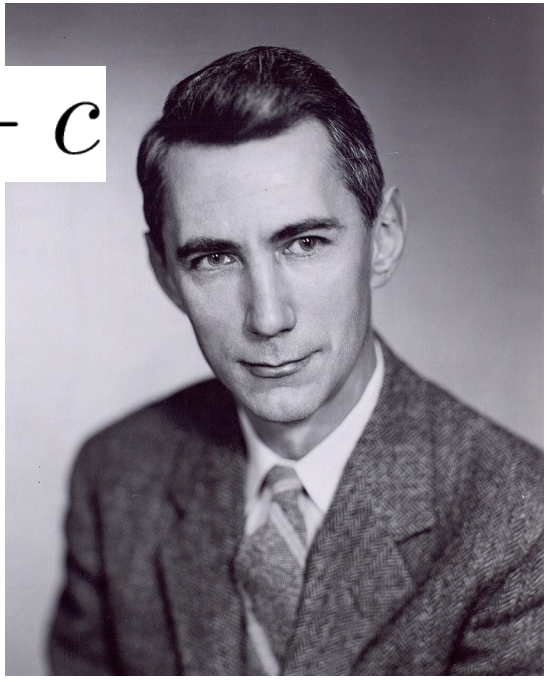


Shannon coding

- Plus un objet *w* est fréquent, plus son code est court

$$l(\mathrm{code}(w)) = -\log_2(\mathrm{freq}(w)) + c$$

$$C(w) = -\log_2(\mathrm{freq}(w))$$



Claude Shannon
1916 - 2001

Shannon coding

- A (42.48%), B (29.50%), C (22.00%), D (6.02%)

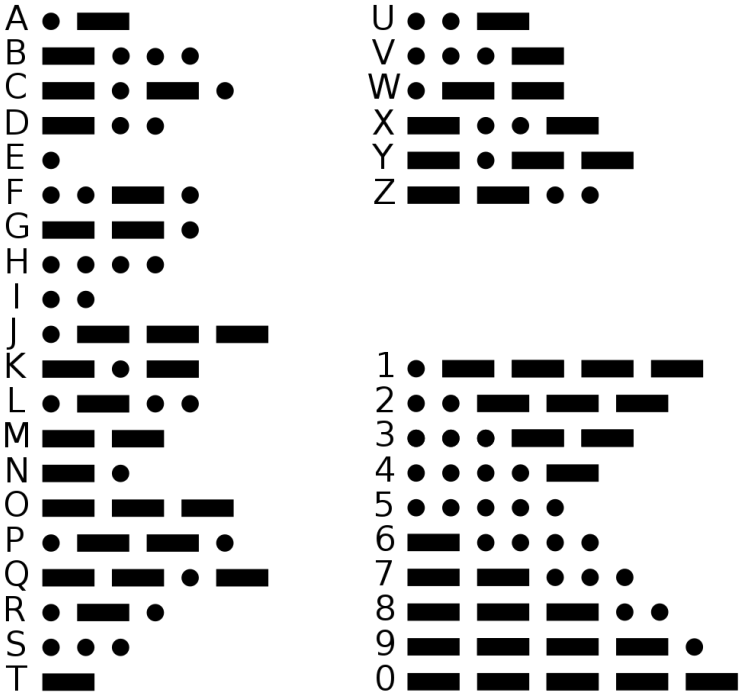
Lettre	Code
A	0
B	1
C	00
D	01

Quel est l’intérêt de ce code ?

Shannon coding

International Morse Code

1. The length of a dot is one unit.
2. A dash is three units.
3. The space between parts of the same letter is one unit.
4. The space between letters is three units.
5. The space between words is seven units.

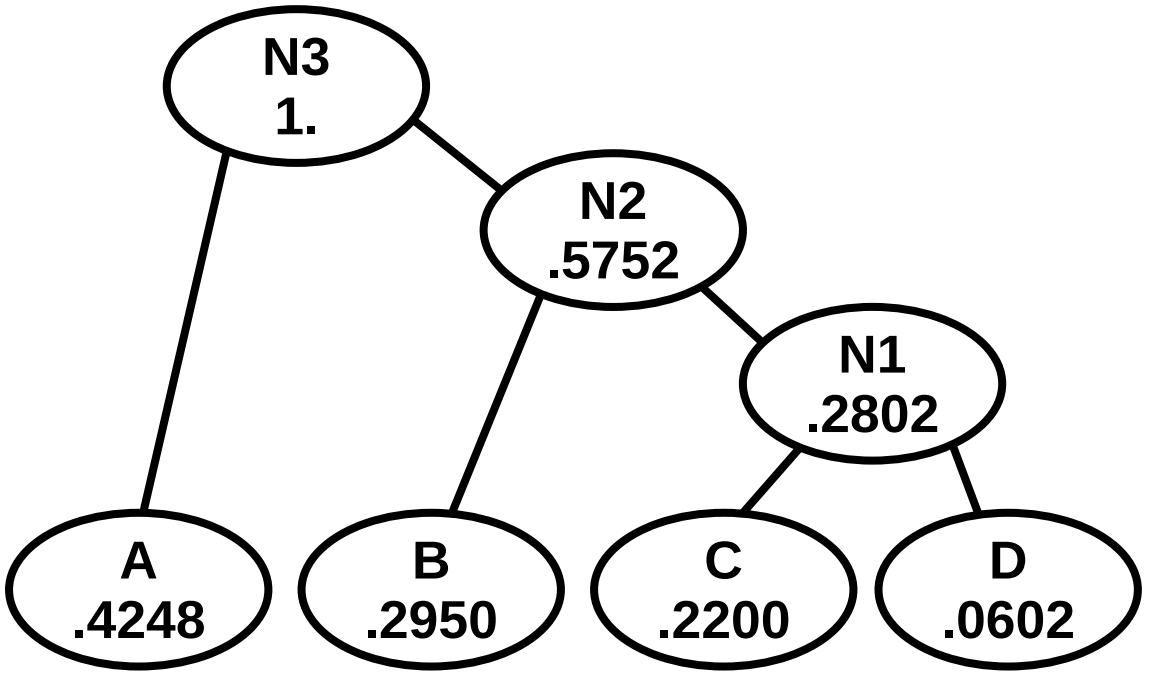


Huffman coding

- Le code Morse suppose qu’il existe un silence pour séparer les lettres (c’est la *couche physique*)
- Sur le ruban d’une machine de Turing il n’y a pas de séparateur

Huffman coding

- A (42.48%), B (29.50%), C (22.00%), D (6.02%)



Lettre	Code
A	0
B	10
C	110
D	111

Quel est l’intérêt de ce code ?

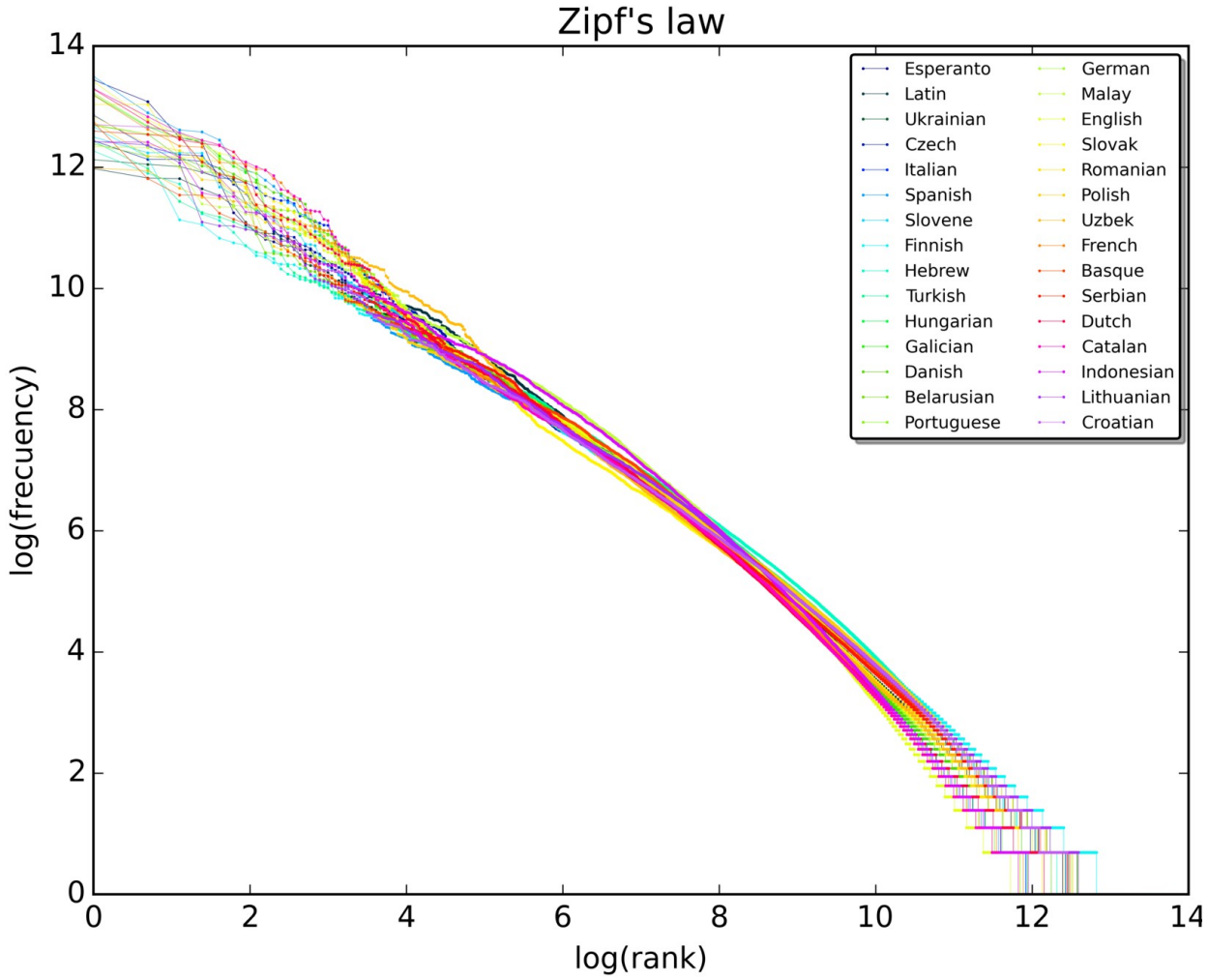
Shannon vs Huffman coding

- Message: 00111
- Shannon:
  - ADBB?
  - CBBB?
  - AABBB?
- Huffman:
  - AAD.

Lettre	Shannon	Huffman
A	0	0
B	1	10
C	00	110
D	01	111

Code préfixe

Loi de Zipf


Zipf's law
log(frecuency)
log(rank)
0
2
4
6
8
10
12
14
Esperanto
Latin
Ukrainian
Czech
Italian
Spanish
Slovene
Finnish
Hebrew
Turkish
Hungarian
Galician
Danish
Belarusian
Portuguese
German
Malay
English
Slovak
Romanian
Polish
Uzbek
French
Basque
Serbian
Dutch
Catalan
Indonesian
Lithuanian
Croatian

Fréquence et complexité

- Shannon et Huffman minimisent le nombre de bits transmis en moyenne
- On peut imaginer d’autres critères à minimiser

Normalized Google distance

$$\mathrm{NID}(x,y) = \frac{C(x,y) - \min(C(x),C(y))}{\max(C(x),C(y))}$$

- Problème : calcul de *C*
- Solution: la fréquence d'un objet est une indication de sa complexité

$$C(x) \approx -\log_2(\mathrm{freq(x)})$$

Nombre de résultats Google

$$C(x) \approx -\log_2(\frac{g(x)}{M})$$

Nombre de pages indexées par Google

Normalized Google distance

Nombre de résultats Google

$$\mathrm{NGD}(x,y) = \frac{\max(\log(g(x)), \log(g(y))) - \log(g(x,y))}{\log M - \min(\log(g(x)), \log(g(y)))}$$

Nombre de pages indexées par Google (130,000,000,000 = 1.3E11 en 2016)

Normalized Google distance

$$\mathrm{NGD}(x,y) = \frac{\max(\log(g(x)), \log(g(y))) - \log(g(x,y))}{\log M - \min(\log(g(x)), \log(g(y)))}$$

- d(“oven”, “fridge”) = 0.405
  - g(“oven”) = 1 260 000 000
  - g(“fridge”) = 937 000 000
  - g(“oven fridge”) = 171 000 000
- d(“oven”, “bird”) = .711
  - g(“bird”) = 2 490 000 000
  - g(“oven bird”) = 92 200 000

Compression, fréquence et complexité

- On peut approximer la complexité ou l’information algorithmique en utilisant
  - La compression
  - La fréquence d’un concept
- On peut utiliser la complexité pour mesurer des distances entre objets