

HLIN102 – Du binaire au web

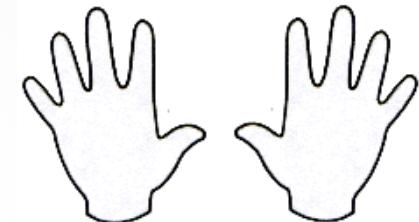
Représentation de nombres et de texte

Sources :

http://lslwww.epfl.ch/pages/teaching/cours_lsl/intro/2.Representation.pdf
<http://perso.univ-lemans.fr/~ywanko/Info1/page25/files/chap1-representation.pdf>
<http://www.linguistes.com/phrase/representations.html>

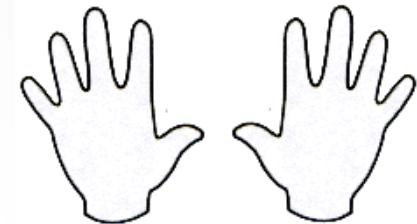
Base de numération

- Humains : base 10



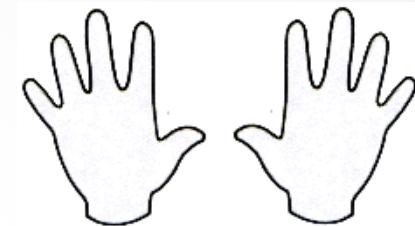
Base de numération

- Humains : base 10
- Machines : base 2



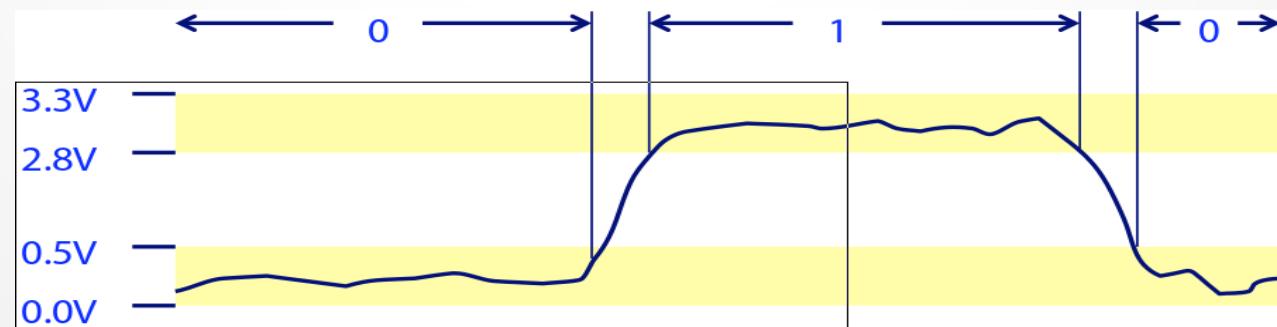
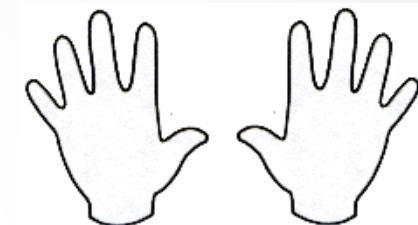
Base de numération

- Humains : base 10
- Machines : base 2
 - stockage facile, à l'aide d'éléments électroniques bistables



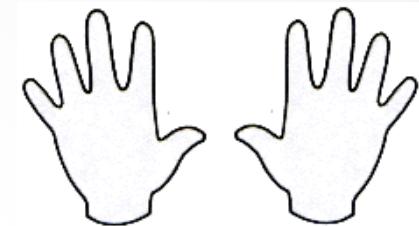
Base de numération

- Humains : base 10
- Machines : base 2
 - stockage facile
 - transmission fiable, même sur des environnements bruyants et imprécis



Base de numération

- Humains : base 10
- Machines : base 2
 - stockage facile
 - transmission fiable
 - simplicité des opérations arithmétiques



Entiers positifs

- Conversion binaire → décimal :

$$\overline{2} \overline{01101011} = ?$$

Entiers positifs

- Conversion binaire → décimal :

$$\overline{01101011} = 0 \times 2^7 + 1 \times 2^6 + 1 \times 2^5 + 0 \times 2^4 + 1 \times 2^3 + 0 \times 2^2 + 1 \times 2^1 + 1 \times 2^0$$

$$\overline{01101011} = 64 + 32 + 8 + 2 + 1$$

$$\overline{01101011} = 107$$

Entiers positifs

- Conversion binaire → décimal :

$$\overline{^201101011} = 0 \times 2^7 + 1 \times 2^6 + 1 \times 2^5 + 0 \times 2^4 + 1 \times 2^3 + 0 \times 2^2 + 1 \times 2^1 + 1 \times 2^0$$

$$\overline{^20110101} = 64 + 32 + 8 + 2 + 1$$

$$\overline{^20110101} = 107$$

- Conversion décimal → binaire :

$$193 = ?$$

Entiers positifs

- Conversion binaire → décimal :

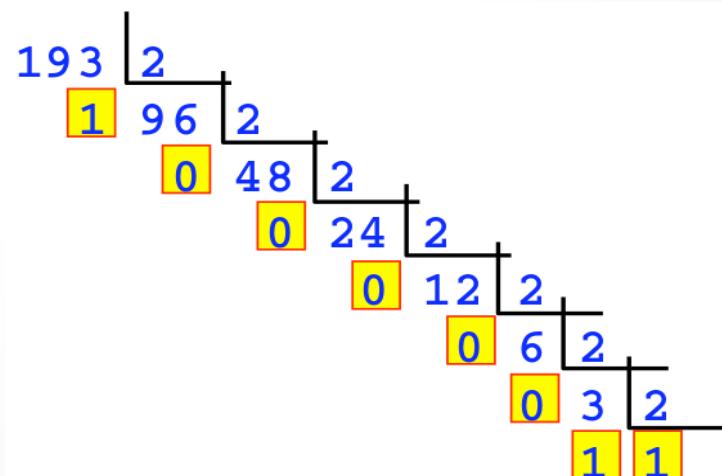
$$\overline{^201101011} = 0 \times 2^7 + 1 \times 2^6 + 1 \times 2^5 + 0 \times 2^4 + 1 \times 2^3 + 0 \times 2^2 + 1 \times 2^1 + 1 \times 2^0$$

$$\overline{^20110101} = 64 + 32 + 8 + 2 + 1$$

$$\overline{^20110101} = 107$$

- Conversion décimal → binaire :

$$193 = \overline{^211000001}$$



Du binaire à l'hexadécimal

- 1 bit = 1 chiffre binaire (binary digit)
 - Abréviation : b
- 1 octet = 8 bits
 - Abréviation : o (français) ou B (anglais)
 - Attention : byte (B) = octet en anglais \neq bit (b)
- Chaque format de données a une taille multiple d'un octet
- La base hexadécimale (1 chiffre = 4 bits) permet une écriture économique des données

Conversion binaire ↔ hexa

- Conversion binaire → hexa :

$$\overline{2} \overline{1011101000101110} = ?$$

Conversion binaire ↔ hexa

- Conversion binaire → hexa :

$$\overline{1011101000101110} = \text{BA2E}$$

1011 1010 0010 1110
B A 2 E

Conversion hexadécimale

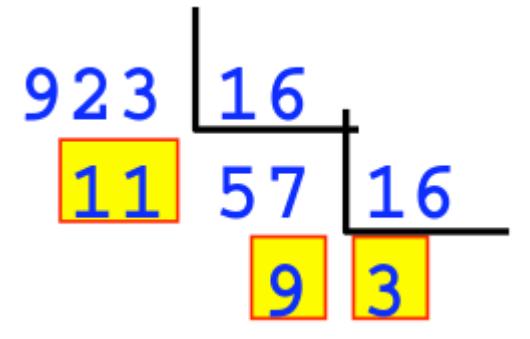
- Conversion binaire → hexa
- Conversion décimal → hexa :

$$923 = ?$$

Conversion hexadécimale

- Conversion binaire → hexa
- Conversion décimal → hexa :

$$923 = ^{16}39B$$



Conversion hexadécimale

- Conversion binaire → hexa
- Conversion décimal → hexa
- Conversion hexa → décimal :

$$\overline{A8CE}_{16} = ?$$

Conversion hexadécimale

- Conversion binaire → hexa
- Conversion décimal → hexa
- Conversion hexa → décimal :

$$\overline{\text{A8CE}}_{16} = 10 \times 16^3 + 8 \times 16^2 + 12 \times 16^1 + 14 \times 16^0$$

$$\overline{\text{A8CE}}_{16} = 40960 + 2048 + 192 + 14$$

$$\overline{\text{A8CE}}_{16} = 43214$$

Les 0 à gauche

- Les ordinateurs ont un format pour représenter les nombres, c'est-à-dire un nombre de chiffres pré-établi
→ on écrit souvent les zéros à gauche
- Exemple: en base 2, sur 4 chiffres, on peut représenter $2^4=16$ entiers entre 0 et 15 :

0	0000	1	0001	2	0010	3	0011
4	0100	5	0101	6	0110	7	0111
8	1000	9	1001	10	1010	11	1011
12	1100	13	1101	14	1110	15	1111

Opérations arithmétiques

- Exemple d'addition :

addition

$$\begin{array}{r} & (1) & (1) \\ & 1 & 0 & 1 & 1 \\ + & 1 & 0 & 0 & 1 \\ \hline 1 & 0 & 1 & 0 & 0 \end{array}$$

(vérification en décimal : $11 + 9 = 20$)

- Possibilité de dépasser la taille du format → « Overflow »

Opérations arithmétiques

- Exemple de multiplication :

multiplication

$$\begin{array}{r} & 1 & 1 & 0 & 1 \\ \times & & 1 & 0 & 1 \\ \hline & 1 & 1 & 0 & 1 \\ & 0 & 0 & 0 & 0 \\ \hline & 1 & 1 & 0 & 1 \\ \hline 1 & 0 & 0 & 0 & 0 & 0 & 1 \end{array}$$

Opérations arithmétiques

- Exemple de soustraction :

soustraction

$$\begin{array}{r} 1 \ 0 \ 1 \ 0 \\ - \ 1 \ 1 \ 1 \\ \hline 0 \ 0 \ 1 \ 1 \end{array}$$

Opérations arithmétiques

- Exemple de division (euclidienne) :

division

$$\begin{array}{r} 1 & 0 & 1 & 0 \\ - & 1 & 1 \\ \hline 1 & 0 & 0 \\ - & 1 & 1 \\ \hline 0 & 0 & 1 \end{array} \left| \begin{array}{cc} 1 & 1 \\ \hline 1 & 1 \end{array} \right.$$

Vers la représentation en mémoire

- « taille d'un mot » = nombre de bits utilisés par un ordinateur pour stocker un entier
- Ordinateurs actuels : (32 bits ou) 64 bits
- Données stockées dans une mémoire, chaque donnée à une adresse différente
- Chaque octet de la mémoire possède une adresse différente.
- Si une donnée contient plus d'un octet, l'adresse de la donnée est celle du premier octet

Stockage en mémoire

- Chaque adresse est stockée dans un mot de la mémoire d'un ordinateur
- Le nombre de bits d'un mot limite donc la taille maximale de la mémoire
- Ordinateurs 32 bits : la taille maximale de la mémoire est de 2^{32} octets ≈ 4 Go
 - Astuce : pour les ordres de grandeur, on utilise l'approximation $2^{10} \approx 10^3$

Entiers négatifs

- Complément à 2 :
 - codage de la valeur absolue
 - inversion des 0 et des 1
 - addition avec 1
 - si le nombre de bits du négatif dépasse celui du positif, on enlève le 1 de gauche.

Entiers négatifs

- Complément à 2
- Le premier bit marque le signe (0 : positif, 1 : négatif)
- Addition : dernière retenue ignorée

$$\begin{array}{r} & 0 & 1 & 1 & 1 \\ + & 1 & 0 & 1 & 0 \\ \hline & 1 & 0 & 0 & 0 & 1 \end{array}$$

Entiers négatifs

- Complément à 2
- Le premier bit marque le signe
(0 : positif – 1 : négatif)
- Addition : dernière retenue ignorée
- k bits permettent de représenter les entiers signés entre -2^{k-1} et $2^{k-1}-1$.

Nombres décimaux

- Décomposition d'un nombre en décimal :

$$96,75 = 9 \times 10^1 + 6 \times 10^0 + 7 \times 10^{-1} + 5 \times 10^{-2}$$

- Décomposition en binaire :

$$96,75 = 1 \times 2^6 + 1 \times 2^5 + 1 \times 2^{-1} + 1 \times 2^{-2}$$

$$96,75 = \overline{1100000,11}_2$$

$$96,75 = \overline{1,1000011}_2 \times 2^6$$

96,

Non représenté car toujours =1 Mantisse Exposant

Normes de représentation

$$96,75 = \overline{1,10000011} \times 2^6$$

- Stockage en mémoire
 - Format IEEE754 simple précision (32 bits) :
 - SM = signe mantisse (1 bit)
 $\text{SM} = 0$ car $96,75 >= 0$
 - E_b = exposant+biais (8 bits)
 $E_b = 6 + 127 = 133 = \overline{10000101}$
 - M = mantisse (23 bits)
 $M = 1000\ 0011\ 0000\ 0000\ 000$
 - Soit : 0 10000101 10000011000000000000000

Normes de représentation

$$96,75 = \overline{21,10000011} \times 2^6$$

- Stockage en mémoire
 - Format IEEE754 simple précision (32 bits)
 - Format IEEE754 double précision (64 bits) :
 - S sur 1 bit
 - E_b sur 11 bits (biais : 1023)
 - M sur 52 bits

Normes de représentation

$$96,75 = \frac{2}{1,10000011} \times 2^6$$

- Stockage en mémoire
 - Format IEEE754 simple précision (32 bits)
 - Format IEEE754 double précision (64 bits)
 - Cas particuliers :
 - Eb=0 M=0 : nombre 0
 - Eb=255 (ou 2047) M=0 : $\pm\infty$ (signe selon SM)
 - Eb=255 (ou 2047) M \neq 0 : NaN (code d'erreur)
 - Eb=0 M \neq 0 : $\pm 0, M^{-126}$ (représentation dénormalisée)

Addition en IEEE754

- Pour simplifier, en 16 bits (biais=7)
- $0\ 0111\ 10\dots 0 + 0\ 0110\ 00\dots 0 = ???$
 - Traduction : $1,1 \times 2^0 + 1 \times 2^{-1} = ?$
 - Passage au plus grand des deux exposants :
 $1,1 \times 2^0 + 0,1 \times 2^0 = ?$
 - Ajout des mantisses (ou soustraction si signes opposés) :
 $1,1 + 0,1 = +10,0$
 - Normalisation : $+10,0 \times 2^0 = +1 \times 2^1$
 - Résultat final : $0\ 1000\ 00\dots 0$
 - Vérification : $1,5 + 0,5 = 2$

Multiplication en IEEE754

- $0\ 1000\ 010\dots 0 \times 0\ 1000\ 10\dots 0 = ???$
 - Traduction : $1,01 \times 2^1 \times 1,1 \times 2^1 = ?$
 - Bit de signe : 0
(0 si les 2 bits de signes sont égaux, 1 sinon)
 - Exposant : $1+1=10$ (somme des exposants)
 - Multiplication des mantisses :
 $1,01 \times 1,1 = 1,01 \times (1+0,1) = 1,01 + 0,101 = 1,111$
 - Résultat final : $0\ 1001\ 1110\dots 0$
 - Vérification : $2,5 \times 3 = 7,5$

Caractères

- Historiquement : code ASCII (7 bits)

Dec	Hx	Oct	Char		Dec	Hx	Oct	Html	Chr		Dec	Hx	Oct	Html	Chr		Dec	Hx	Oct	Html	Chr
0	0 000	NUL	(null)		32	20 040	 	Space			64	40 100	@	Ø	ø		96	60 140	`	`	
1	1 001	SOH	(start of heading)		33	21 041	!	!	!		65	41 101	A	A	a		97	61 141	a	à	
2	2 002	STX	(start of text)		34	22 042	"	"	"		66	42 102	B	B	b		98	62 142	b	ò	
3	3 003	ETX	(end of text)		35	23 043	#	#	#		67	43 103	C	C	c		99	63 143	c	ç	
4	4 004	EOT	(end of transmission)		36	24 044	$	\$	\$		68	44 104	D	D	d		100	64 144	d	đ	
5	5 005	ENQ	(enquiry)		37	25 045	%	%	%		69	45 105	E	E	e		101	65 145	e	é	
6	6 006	ACK	(acknowledge)		38	26 046	&	&	&		70	46 106	F	F	f		102	66 146	f	ƒ	
7	7 007	BEL	(bell)		39	27 047	'	'	'		71	47 107	G	G	g		103	67 147	g	ğ	
8	8 010	BS	(backspace)		40	28 050	(((72	48 110	H	H	h		104	68 150	h	ḥ	
9	9 011	TAB	(horizontal tab)		41	29 051)))		73	49 111	I	I	i		105	69 151	i	í	
10	A 012	LF	(NL line feed, new line)		42	2A 052	*	*	*		74	4A 112	J	J	j		106	6A 152	j	ј	
11	B 013	VT	(vertical tab)		43	2B 053	+	+	+		75	4B 113	K	K	k		107	6B 153	k	ķ	
12	C 014	FF	(NP form feed, new page)		44	2C 054	,	,	,		76	4C 114	L	L	l		108	6C 154	l	ł	
13	D 015	CR	(carriage return)		45	2D 055	-	-	-		77	4D 115	M	M	m		109	6D 155	m	ṁ	
14	E 016	SO	(shift out)		46	2E 056	.	.	.		78	4E 116	N	N	n		110	6E 156	n	ń	
15	F 017	SI	(shift in)		47	2F 057	/	/	/		79	4F 117	O	O	o		111	6F 157	o	ö	
16	10 020	DLE	(data link escape)		48	30 060	0	0	0		80	50 120	P	P	p		112	70 160	p	پ	
17	11 021	DC1	(device control 1)		49	31 061	1	1	1		81	51 121	Q	Q	q		113	71 161	q	ϙ	
18	12 022	DC2	(device control 2)		50	32 062	2	2	2		82	52 122	R	R	r		114	72 162	r	ڒ	
19	13 023	DC3	(device control 3)		51	33 063	3	3	3		83	53 123	S	S	s		115	73 163	s	ڗ	
20	14 024	DC4	(device control 4)		52	34 064	4	4	4		84	54 124	T	T	t		116	74 164	t	ߕ	
21	15 025	NAK	(negative acknowledge)		53	35 065	5	5	5		85	55 125	U	U	u		117	75 165	u	ۊ	
22	16 026	SYN	(synchronous idle)		54	36 066	6	6	6		86	56 126	V	V	v		118	76 166	v	ۊ	
23	17 027	ETB	(end of trans. block)		55	37 067	7	7	7		87	57 127	W	W	w		119	77 167	w	ۊ	
24	18 030	CAN	(cancel)		56	38 070	8	8	8		88	58 130	X	X	x		120	78 170	x	ۊ	
25	19 031	EM	(end of medium)		57	39 071	9	9	9		89	59 131	Y	Y	y		121	79 171	y	ۊ	
26	1A 032	SUB	(substitute)		58	3A 072	:	:	:		90	5A 132	Z	Z	z		122	7A 172	z	ۊ	
27	1B 033	ESC	(escape)		59	3B 073	;	;	;		91	5B 133	[[[123	7B 173	{	{	
28	1C 034	FS	(file separator)		60	3C 074	<	<	<		92	5C 134	\	\	\		124	7C 174	|		
29	1D 035	GS	(group separator)		61	3D 075	=	=	=		93	5D 135]]]		125	7D 175	}	}	
30	1E 036	RS	(record separator)		62	3E 076	>	>	>		94	5E 136	^	^	^		126	7E 176	~	~	
31	1F 037	US	(unit separator)		63	3F 077	?	?	?		95	5F 137	_	_	_		127	7F 177		DEL	

Caractères : code ASCII

- Historiquement : code ASCII (7 bits)
 - Repose sur une table de correspondance entier ↔ caractère
 - Symboles 0 à 31 = commandes de terminal, symbole 127 = suppression d'un caractère
 - Insuffisant pour les variantes régionales
 - nombreuses variantes souvent incompatibles
 - 1 bit inutilisé

Caractères : codes actuels

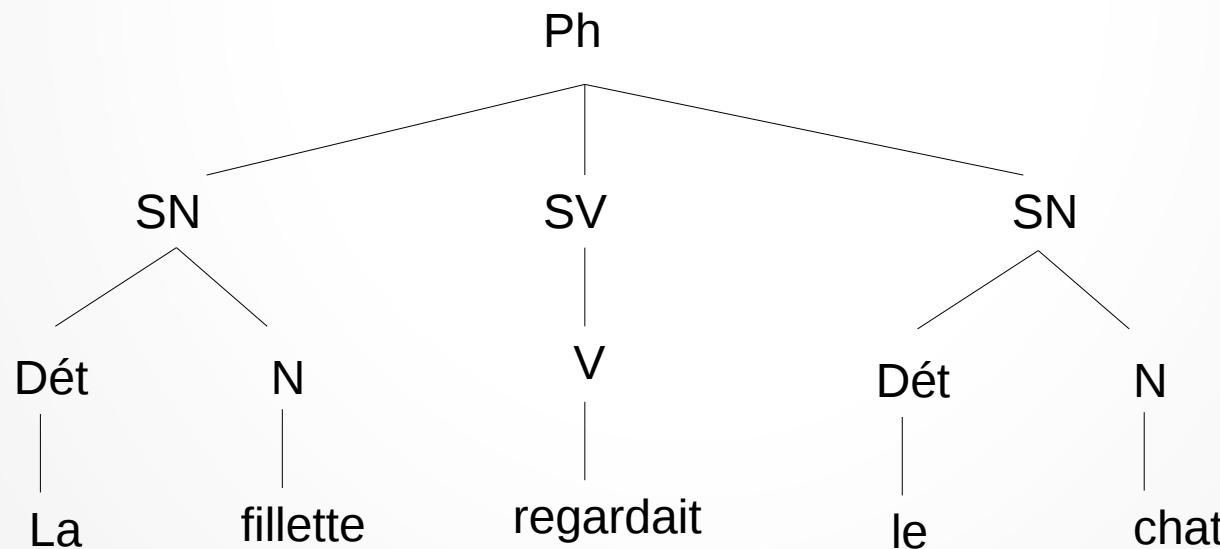
- Historiquement : code ASCII (7 bits)
- Nombreuses extensions
 - sur 8 bits (UTF-8), 32 bits (UNICODE)
 - format d'encodage précisé par 3 octets en début de flux → risques d'incompatibilité avec les anciennes applications

Phrases : exemples

- Notation parenthésée de Chomsky

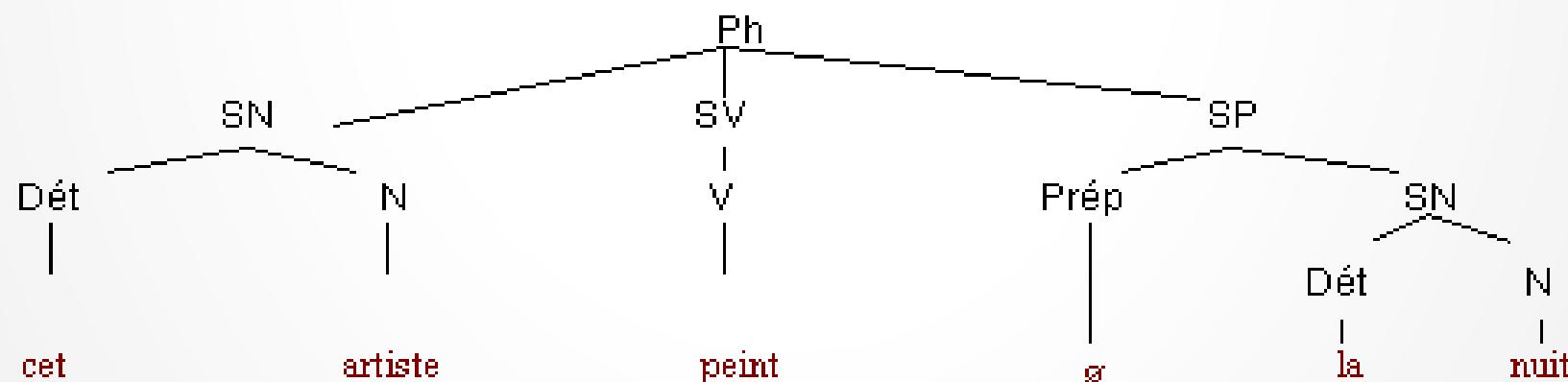
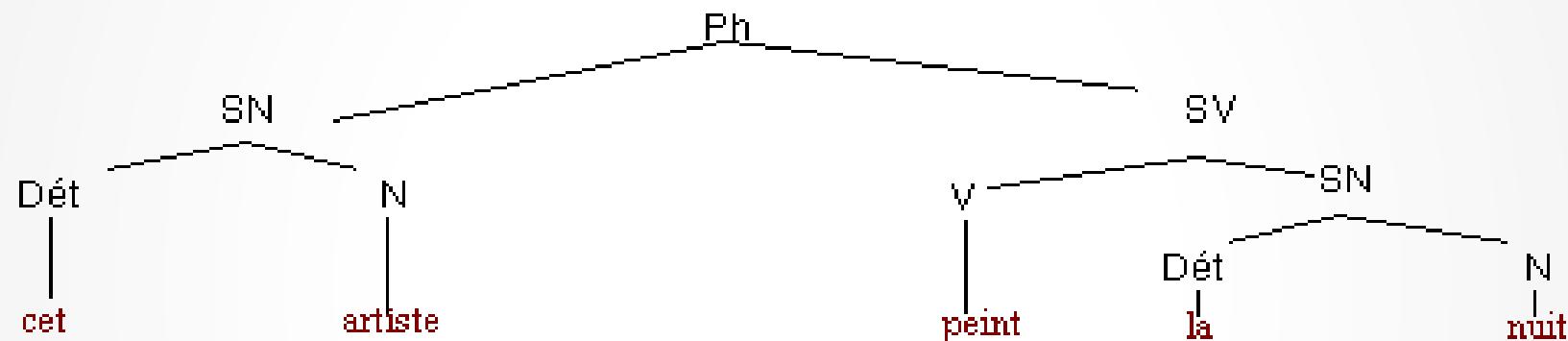
[Ph[SN [Dét la] [N fillette]][SV [V regardait] [SN[Dét le] [N chat]]]]

- Arbre syntaxique



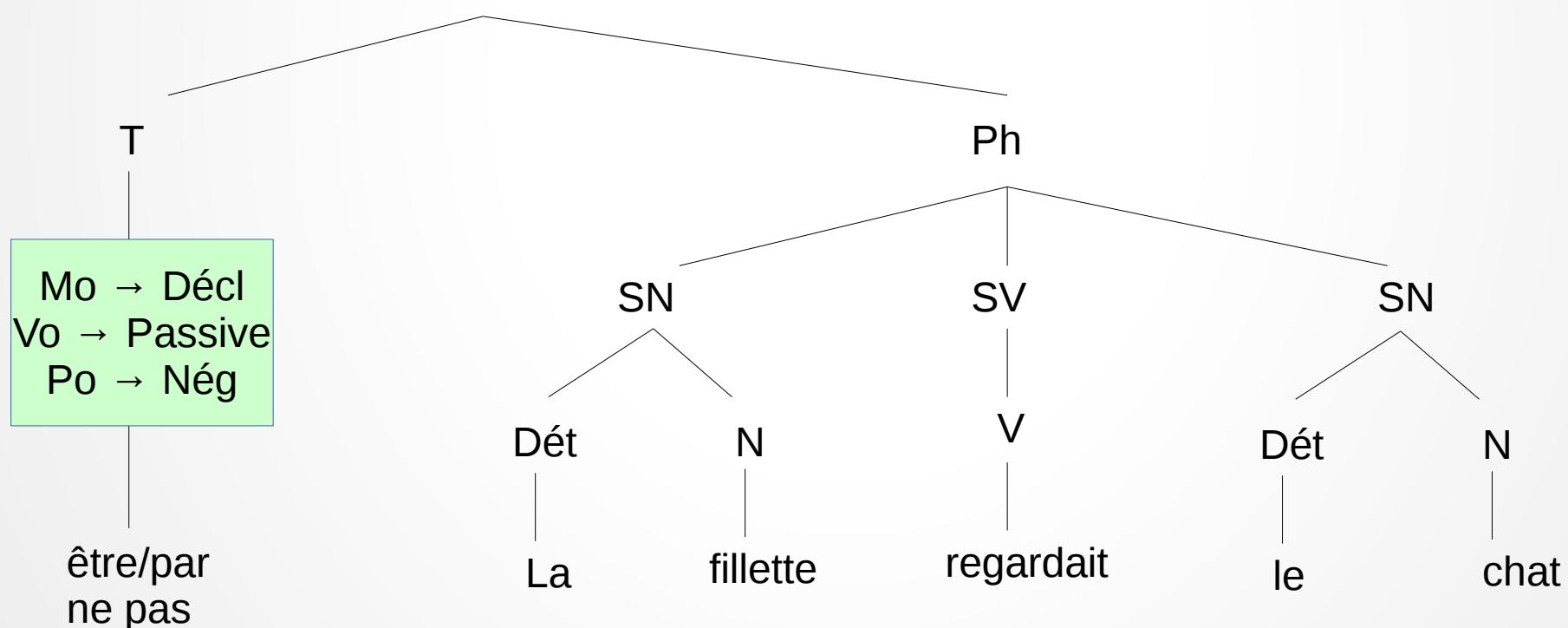
Ambiguité

- Que signifient ces deux arbres ?



Transformation

- On peut appliquer des opérations à une phrase ; comment lire ceci ?



Représenter du sens

- Avec le Web 2.0, intérêt pour la signification contenue dans les informations
- Représentation d'informations par des triplets : élément 1, relation, élément 2. Chaque élément du triplet appartient à un espace de noms
- On obtient un réseau de connexions à partir duquel il est possible de générer de nouvelles informations.

C'est tout pour aujourd'hui.

Merci pour votre attention.

