Tecniche della Programmazione, lez. 11

Ebbene sì …

Sistemi di numerazione e aritmetica binaria

Rappresentazione dei numeri

- interi: complemento a due
- reali: Floating Point

...

There are only 10
types of people
in the world:
Those who understand binary
and those who don't.

Sistemi di numerazione: NUMERI e NUMERALI

Un numero è un concetto … il NUMERALE rappresenta quel concetto.
E ci sono vari modi per rappresentare i numeri.

Un numerale è scritto usando CIFRE

sistema di numerazione decimale

CIFRE decimali

0	1	2	3	4
5	6	7	8	9

millenovecentosessantanove → 1969

millenovecentosessantanove → MDCCCCLXIX

M	D	C	L
X	V	I	

CIFRE romane

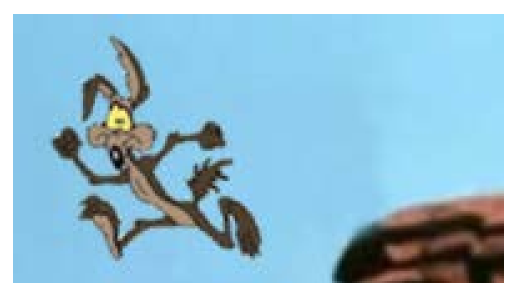
sistema di numerazione Romano (*)

CIFRE maya

sistema di numerazione Maya

Numero	Forma verticale	Forma orizzontale	Numero	Forma verticale	Forma orizzontale
0			10		
1			11		
2			12		
3			13		
4			14		
5			15		
6			16		
7			17		
8			18		
9			19		

millenovecentosessantanove

→ 

(*) MCMLXIX

Sistemi di numerazione posizionali

Quello che usiamo normalmente (sistema decimale) è un sistema posizionale in base 10.

$$1969 = 1 \times 10^3 + 9 \times 10^2 + 6 \times 10^1 + 9 \times 10^0$$

"posizionale" = ogni cifra ha un peso, che è una potenza della base; la potenza dipende dalla posizione della cifra nel numerale

numero espresso in base ***b***, con un numerale di n cifre (in cui ogni cifra è una tra le ***b*** possibili): $(c_{n-1}c_{n-2}\dots c_2c_1c_0)_b$

$(1\ 9\ 6\ 9)_{10}$

corrisponde al valore (nel sistema decimale)

$$\sum_{k=n-1}^{0} c_k \cdot b^k$$

quali sono I numerali qui? ☺

$$(c_{n-1}c_{n-2}\dots c_2c_1c_0)_b = c_{n-1} \cdot b^{n-1} + c_{n-2} \cdot b^{n-2} + \dots + c_1 \cdot b^1 + c_0 \cdot b^0$$

Sistemi di numerazione posizionali

Quello che usiamo normalmente (sistema decimale) è un sistema posizionale in base 10.

$$1969 = 1 \times 10^3 + 9 \times 10^2 + 6 \times 10^1 + 9 \times 10^0$$

"posizionale" = ogni cifra ha un peso, che è una potenza della base; la potenza dipende dalla posizione della cifra nel numerale

numero espresso in base ***b***, con un numerale di n cifre (in cui ogni cifra è una tra le ***b*** possibili): $(c_{n-1}c_{n-2}\ldots c_2c_1c_0)_b$

corrisponde al valore (nel sistema decimale)

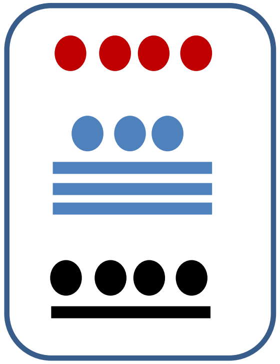
$$\sum_{k=n-1}^{0} c_k \cdot b^k$$

numerale

$$(c_{n-1}c_{n-2}\ldots c_2c_1c_0)_b = c_{n-1} \cdot b^{n-1} + c_{n-2} \cdot b^{n-2} + \ldots + c_1 \cdot b^1 + c_0 \cdot b^0$$

ACME manuals: Maya number system

millenovecentosessantanove →



= 4 * (20*20) = 1600

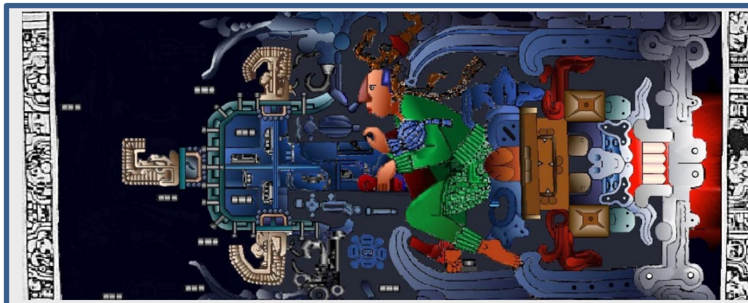
= 18 * (20) = 360

= 9 * (1) = 9

Uso base …
peso della cifra più in basso: 1;
peso della seconda cifra, salendo, 20;
peso della terza cifra 20^2 cioè 400
… 20^3…20^4…20^5 …

Numero	Forma verticale	Forma orizzontale	Numero	Forma verticale	Forma orizzontale
0			10	=	||
1	∘	∘	11	∘ =	∘||
2	∘∘	∘∘	12	∘∘ =	∘∘||
3	∘∘∘	∘∘∘	13	∘∘∘ =	∘∘∘||
4	∘∘∘∘	∘∘∘∘	14	∘∘∘∘ =	∘∘∘∘||
5	—	|	15	≡	|||
6	∘ —	∘|	16	∘ ≡	∘|||
7	∘∘ —	∘∘|	17	∘∘ ≡	∘∘|||
8	∘∘∘ —	∘∘∘|	18	∘∘∘ ≡	∘∘∘|||
9	∘∘∘∘ —	∘∘∘∘|	19	∘∘∘∘ ≡	∘∘∘∘|||

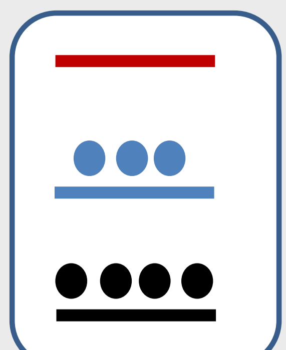
sistema di numerazione Maya; Numeri in **base 20**, scritti verticalmente - cifra più pesante più in alto



Uso mistico

18*20 invece di 20*20
(fa 360, che è special)…

millenovecentosessantanove →



= 5 * (18*20) = 1800

= 8 * 20 = 160

= 9 * 1 = 9

Calcolo delle cifre in base b per un numero decimal

per ritrovare la prima cifra (c_0) del numero rappresentato da

$$(c_{n-1}c_{n-2}\dots c_2c_1c_0)_b$$

(facciamo con b=10)

$$numero = \frac{numero}{10} \cdot 10 + RESTO$$

$$RESTO = c_0$$

Ex. $1969 = (1969/10) \cdot 10 + 9$

$196 = (196/10) \cdot 10 + 6$

$19 = (19/10) \cdot 10 + 9$

...

$$(c_{n-1}c_{n-2}\dots c_2c_1c_0)_b = \sum_{k=n-1}^{0} c_k \cdot b^k$$

$$(c_{n-1}c_{n-2}\dots c_2c_1c_0)_b = c_{n-1} \cdot b^{n-1} + c_{n-2} \cdot b^{n-2} + \dots + c_1 \cdot b^1 + c_0 \cdot b^0$$

Calcolo delle cifre in base b per un numero decimal

per ritrovare la prima cifra (c_0) del numero rappresentato da

$$(c_{n-1}c_{n-2}\ldots c_2c_1c_0)_b$$

$$numero = \frac{numero}{b} \cdot b + RESTO \qquad RESTO = c_0$$

le operazioni sono fatte nel sistema "di partenza" (cioè il decimale)

$$\sum_{k=n-1}^{0} c_k \cdot b^k \;/\; b \;\cdot\; b \;+\; c_0 = \sum_{k=n-1}^{0} c_k \cdot b^k$$

$$(c_{n-1} \cdot b^{n-1} + c_{n-2} \cdot b^{n-2} + \ldots + c_1 \cdot b^1 + c_0 \cdot b^0) \;/\; b \;\cdot\; b \;+\; c_0 =$$

$$= c_{n-1} \cdot b^{n-1} + c_{n-2} \cdot b^{n-2} + \ldots + c_1 \cdot b^1 + c_0 \cdot b^0$$

Calcolo delle cifre in base b per un numero decimal

$$(c_{n-1} \cdot b^{n-1} + c_{n-2} \cdot b^{n-2} + \ldots + c_1 \cdot b^1 + c_0 \cdot b^0) \ / \ b \cdot b + c_0 =$$

$$= c_{n-1} \cdot b^{n-1} + c_{n-2} \cdot b^{n-2} + \ldots + c_1 \cdot b^1 + c_0 \cdot b^0$$

NB se usassi QUESTO NUMERO

per ripetere l'operazione ...

dividendolo per b e poi moltiplicando per b ... otterrei c_1

e continuando si ottengono anche le altre cifre:

Il metodo si chiama ...

Metodo delle divisioni successive

per trovare le cifre (in base b) che rappresentano il numero dato (usando la rappresentazione e le operazioni decimali del numero)

$$\textbf{numero} = (c_{n-1}c_{n-2}\ldots c_2c_1c_0)_b$$

$$numero = \frac{numero}{b} \cdot b + RESTO \qquad RESTO = c_0$$

Metodo delle divisioni successive

per trovare le cifre (in base b) che rappresentano il numero dato (usando la rappresentazione e le operazioni decimali del numero)

$$\textbf{numero} = (c_{n-1}c_{n-2}\ldots c_2c_1c_0)_b$$

$$numero = \boxed{\frac{numero}{b}} \cdot b + RESTO \qquad RESTO = c_0$$

$$\boxed{} = \boxed{} \cdot b + RESTO \qquad RESTO = c_1$$

Metodo delle divisioni successive

per trovare le cifre (in base b) che rappresentano il numero dato (usando la rappresentazione e le operazioni decimali del numero)

$$\textbf{numero} = (c_{n-1}c_{n-2}\ldots c_2c_1c_0)_b$$

$$numero = \boxed{\frac{numero}{b}} \cdot b + RESTO \qquad RESTO = c_0$$

$$\boxed{*} = \boxed{\frac{*}{b}} \cdot b + RESTO \qquad RESTO = c_1$$

$$\boxed{**} = \boxed{\frac{**}{b}} \cdot b + RESTO \qquad RESTO = c_2$$

$$\boxed{***} = \boxed{\frac{***}{b}} \cdot b + RESTO \qquad RESTO = c_3$$

e così via, fino a che le cifre sono tutte

Metodo delle divisioni successive

per trovare le cifre (in base b) che rappresentano il numero dato (usando la rappresentazione e le operazioni decimali del numero)

$$\textbf{numero} = (\ldots c_5 c_4 c_3 c_2 c_1 c_0)_b$$

$$1969 = \boxed{196} \cdot 10 + RESTO \qquad RESTO = c_0$$

$$\boxed{} = \boxed{} \cdot b + RESTO \qquad RESTO = c_1$$

☺

$$\boxed{} = \boxed{} \cdot b + RESTO \qquad RESTO = c_2$$

$$\boxed{} = \boxed{} \cdot b + RESTO \qquad RESTO = c_3$$

Metodo delle divisioni successive

per trovare le cifre (in base b) che rappresentano il numero dato (usando la rappresentazione e le operazioni decimali del numero)

quando la divisione (intera) viene 0 … il resto è l'ultima cifra

$$\textbf{numero} = (c_{n-1}c_{n-2}\ldots c_2c_1c_0)_b$$

$$1969 = \boxed{196} \cdot 10 + RESTO \qquad RESTO = 9$$

$$\boxed{196} = \boxed{19} \cdot 10 + RESTO \qquad RESTO = 6$$

$$\boxed{19} = \boxed{1} \cdot 10 + RESTO \qquad RESTO = 9$$

$$\boxed{1} = \boxed{0} \cdot 10 + RESTO \qquad RESTO = 1$$

anche numeri frazionari …

Quello che usiamo di solito (sistema decimale) è un sistema posizionale in base 10.

$$\mathbf{756.25} = \mathbf{7}\times 10^{2} + \mathbf{5}\times 10 + \mathbf{6} + \mathbf{2}\times 10^{-1} + \mathbf{5}\times 10^{-2}$$

"posizionale" = ogni cifra ha un peso, che è una potenza della base; la potenza dipende dalla posizione della cifra nel numerale

anche numeri frazionari ...

Quello che usiamo di solito (sistema decimale) è un sistema posizionale in base 10.

$$\mathbf{756.25} = \mathbf{7}\times 10^{2} + \mathbf{5}\times 10 + \mathbf{6} + \mathbf{2}\times 10^{-1} + \mathbf{5}\times 10^{-2}$$

"posizionale" = ogni cifra ha un peso, che è una potenza della base; la potenza dipende dalla posizione della cifra nel numerale

numero espresso in base *b*, con n cifre per la parte intera e m cifre per la parte frazionaria (cifre della base b):

$$(c_{n-1}c_{n-2}\ldots c_2c_1c_0.c_{-1}c_{-2}\ldots c_{-(m-1)}c_{-m})_b$$

corrisponde al valore (nel sistema decimale)

$$\sum_{k=n-1}^{-m} c_k \cdot bk$$

$$(c_{n-1}c_{n-2}\ldots c_2c_1c_0.c_{-1}\ldots\ldots c_{-m})_b =$$

$$c_{n-1}\cdot b^{n-1} + c_{n-2}\cdot b^{n-2} + \ldots + c_1\cdot b^1 + c_0\cdot b^0 +$$

$$+ c_{-1}\cdot b^{-1} + c_{-2}\cdot b^{-2} + \ldots + c_{-m}\cdot b^{-m}$$

basi

base 10

cifre	0 1 2 3 4 5 6 7 8 9									
contiamo …	0	1	2	3	4	5	6	7	8	9
dieci	**10**	11	12	13	14	15	16	17	18	19
	20	…	**99**	**100**						

con due cifre, al più 99

100 = 10x10

Con 5 cifre [0, 99999]

Con 7 cifre [0, 9999999]

NB con n cifre si rappresentano b^n numeri naturali in base b da 0 a $(b^n - 1)$ … $[0, (b^n - 1)]$

INTERVALLO DI RAPPRESENTABILITà

basi

NB con n cifre si rappresentano b^n numeri naturali in base b da 0 a $(b^n - 1)$ … $[0, (b^n - 1)]$

base 10

cifre	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**
contiamo …	0	1	2	3	4	5	6	7	8	9
dieci	**10**	11	12	13	14	15	16	17	18	19
	20 …		**99**	**100**						

con due cifre, al più 99

100 = 10x10

base 8

cifre	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**
contiamo …	0	1	2	3	… ☺			
otto	**?**							
sedici	20 …				**?**			
	…	**77** $(63)_{10}$	100 $(64 = 8x8)_{10}$	101	102 …			

quindici ?

venti

$(21)_8 = 2*8^1 + 1*8^0 = (17)_{10}$ diciassette

basi

NB con n cifre si rappresentano b^n numeri naturali in base b da 0 a $(b^n - 1)$ ... $[0, (b^n - 1)]$

base 10

cifre	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**
contiamo ...	0	1	2	3	4	5	6	7	8	9
dieci	**10**	11	12	13	14	15	16	17	18	19
	20 ...		**99**	**100**						

con due cifre, al più 99

100 = 10x10

base 8

cifre	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**
contiamo ...	0	1	2	3	4	5	6	7
otto	**10**	11	12	13	14	15	16	17
sedici	20	21	22	23	24	...		
	...	**77** $(63)_{10}$	100 $(64 = 8x8)_{10}$	101	102	...		

quindici

venti

$(21)_8 = 2*8^1 + 1*8^0 = (17)_{10}$ diciassette

$(47)_8 =$ ☺ trentanove

$(101)_8 =$ ☺

basi

NB con n cifre si rappresentano b^n numeri naturali in base b da 0 a $(b^n -1)$ ... $[0, (b^n -1)]$

base 10

cifre	0	1	2	3	4	5	6	7	8	9
contiamo ...	0	1	2	3	4	5	6	7	8	9
dieci	**10**	11	12	13	14	15	16	17	18	19
	20 ...		**99**	**100**						

con due cifre, al più 99

100 = 10x10

base 8

cifre	0	1	2	3	4	5	6	7
contiamo ...	0	1	2	3	4	5	6	7
otto	**10**	11	12	13	14	15	16	17
sedici	20	21	22	23	24 ...			
	...	**77**	100	101	102 ...			
		$(63)_{10}$	$(64 = 8x8)_{10}$					

quindici

venti

$(21)_8 = 2*8^1 + 1*8^0 = (17)_{10}$ diciassette

$(47)_8 = 4*8^1 + 7*8^0 = (39)_{10}$ trentanove

$(101)_8 = 1*8^2 + 0*8^1 + 1*8^0 = (65)_{10}$ sessantacinque

basi

NB con n cifre si rappresentano b^n numeri naturali in base b da 0 a $(b^n -1)$ … $[0, (b^n -1)]$

base 8 (Sistema ottale – o anche *octal*, *oct*)

cifre	**0 1 2 3 4 5 6 7**							
contiamo …	0	1	2	3	4	5	6	7
	10	11	12	13	14	15	16	17
	20	21	22	23	24	…		
	…	**77**	100	101				

$(31)_8 = (?)_{10}$

feste confuse (dagli informatici)

base 8 (Sistema ottale – o anche *octal*, *oct*)

cifre	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**
contiamo …	0	1	2	3	4	5	6	7
	10	11	12	13	14	15	16	17
	20	21	22	23	24	…		
	…	**77**	100	101				

$(31)_8 = (3 \times 8+1 = 25)_{10}$

quindi (Dec = sistema decimale ...)

Dec 25 = Oct 31

feste confuse (dagli informatici)

base 8 (Sistema ottale – o anche *octal*, *oct*)

cifre	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**
contiamo …	0	1	2	3	4	5	6	7
	10	11	12	13	14	15	16	17
	20	21	22	23	24	…		
	…	**77**	100	101				

$(\ 31\ )_8 = (\ 3\times8+1 = 25\ )_{10}$

quindi (Dec = sistema decimale ...)

Dec 25 = Oct 31

basi

NB con n cifre si rappresentano b^n numeri naturali in base b da 0 a $(b^n - 1)$ … $[0, (b^n - 1)]$

base 8

cifre	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**
contiamo …	0	1	2	3	4	5	6	7
	10	11	12	13	14	15	16	17
	20	21	22	23	24	…		
	…	**77**	100	101				

Con 5 cifre [0, 77777] cioè da 0 a trentaduemilasettecentosessantaset

Con 7 cifre [0, 7777777] cioè da 0 a $(2.097.151)_{10}$

☺ check a casa …

basi

NB con n cifre si rappresentano b^n numeri naturali in base b da 0 a $(b^n - 1)$ ... $[0, (b^n - 1)]$

base 10

```
cifre        0 1 2 3 4 5 6 7 8 9
contiamo ... 0   1   2   3   4   5   6   7   8   9
             10  11  12  13  14  15  16  17  18  19
             20 ...  99  100
```

dieci

con due cifre, al più 99

100 = 10x10

base 8

```
cifre        0 1 2 3 4 5 6 7
contiamo ... 0   1   2   3   4   5   6   7
             10  11  12  13  14  15  16  17
             20  21  22  23  24 ...
             ... 77  100  101 102 ...
```

otto

sedici

quindici

venti

$(63)_{10}$ $(64 = 8x8)_{10}$

base 2

```
cifre        0 1
contiamo ... 0   1
             10  11
             100 101 110 111
             1000 1001 1010 1011 1100 1101 1110 1111
             ...
```

con due cifre, si rappresentano 4 numeri in totale (num max: 3)

con tre cifre, max numero rappresentato: 7

$(4=2x2)_{10}$

$(12)_{10}$ $(15)_{10}$

base 2

base 2

cifre **0 1**

contiamo … 0 1

10 11 con due cifre, si rappresentano 4 numeri in totale (max 3)

$(4=2\text{x}2)_{10}$ **100** 101 110 111 con tre cifre, max numero rappresentato: 7

1000 1001 1010 1011 1100 $(12)_{10}$ 1101 1110 1111 $(15)_{10}$

…

Con 2 cifre [0, 11] da 0 a tre [0, (2^2-1)]

Con 5 cifre [0, 11111] da 0 a (2^5-1) [0, 31] decimale

Con 7 cifre [0, 1111111] da 0 a (2^7-1) [0, 127] decimale

NB con n cifre si rappresentano b^n numeri naturali in base b da 0 a $(b^n -1)$ … [0, $(b^n -1)$]

INTERVALLO DI RAPPRESENTABILITà

base 2

> NB con n cifre si rappresentano b^n numeri naturali in base b da 0 a $(b^n - 1)$ ... $[0, (b^n - 1)]$

base 2

```
cifre          0 1
contiamo …     0   1
               10  11
               100 101 110 111
               1000 1001 1010 1011 1100 1101 1110 1111
```

con 32 bit si rappresentano i numeri naturali in $[0, (2^{32} - 1)]$

con 5 bit si rappresentano i numeri naturali in $[0, 31]$

Quindi se lo spazio per i numerali è limitato, altrettanto limitato è l'insieme dei numeri rappresentati dai numerali

Quanti bit servono per rappresentare 64 256 ?

basi

NB con n cifre si rappresentano b^n numeri naturali in base b da 0 a $(b^n - 1)$ … $[0, (b^n - 1)]$

base 2

```
cifre          0 1
contiamo …     0   1
               10  11
               100 101 110 111
               1000 1001 1010 1011 1100 1101 1110 1111
```

con 32 bit si rappresentano i numeri naturali in $[0, (2^{32} - 1)]$

con 5 bit si rappresentano i numeri naturali in [0, 31]

Quindi se lo spazio per i numerali è limitato, altrettanto limitato è l'insieme dei numeri rappresentati dai numerali

Quanti bit servono per rappresentare 64 256 ?

6 8

?

basi

NB con n cifre si rappresentano b^n numeri naturali in base b da 0 a $(b^n - 1)$ ... $[0, (b^n - 1)]$

base 2

```
cifre         0 1
contiamo …    0   1
              10  11
              100 101 110 111
              1000 1001 1010 1011 1100 1101 1110 1111
```

con 32 bit si rappresentano i numeri naturali in $[0, (2^{32} - 1)]$

con 5 bit si rappresentano i numeri naturali in $[0, 31]$

Quindi se lo spazio per i numerali è limitato, altrettanto limitato è l'insieme dei numeri rappresentati dai numerali

Quanti bit servono per rappresentare 64 256 ?

6 8

$[0, (2^6 - 1)]$! !

?

basi

NB con n cifre si rappresentano b^n numeri naturali in base b da 0 a $(b^n - 1)$ … $[0, (b^n - 1)]$

base 2

```
cifre          0 1
contiamo …     0   1
               10  11
               100 101 110 111
               1000 1001 1010 1011 1100 1101 1110 1111
```

con 32 bit si rappresentano i numeri naturali in $[0, (2^{32} - 1)]$

con 5 bit si rappresentano i numeri naturali in [0, 31]

Quindi se lo spazio per i numerali è limitato, altrettanto limitato è l'insieme dei numeri rappresentati dai numerali

Quanti bit servono per rappresentare 64 256 ?

6 8

intervallo di rappresentabilità

$[0, (2^6 - 1)]$! !

[0, 63] non contiene 64

6 non bastano per 64;
8 non bastano per 256

!

basi

NB con n cifre si rappresentano b^n numeri naturali in base b da 0 a $(b^n -1)$ … $[0, (b^n -1)]$

base 2

```
      cifre          0 1
      contiamo …     0   1
                     10  11
                     100 101 110 111
                     1000 1001 1010 1011 1100 1101 1110 1111
```

con 32 bit si rappresentano i numeri naturali in $[0, (2^{32} -1)]$

con 5 bit si rappresentano i numeri naturali in [0, 31]

Quindi se lo spazio per i numerali è limitato, altrettanto limitato è l'insieme dei numeri rappresentati dai numerali

Quanti bit servono per rappresentare 64 256 ?

6 8

$[0, (2^8 -1)]$! !

[0, 255] non contiene 256

intervallo di rappresentabilità

!

basi

NB con n cifre si rappresentano b^n numeri naturali in base b da 0 a $(b^n -1)$ ... $[0, (b^n -1)]$

base 2

```
cifre          0 1
contiamo …     0   1
               10  11
               100 101 110 111
               1000 1001 1010 1011 1100 1101 1110 1111
```

con 32 bit si rappresentano i numeri naturali in $[0, (2^{32} -1)]$

con 5 bit si rappresentano i numeri naturali in [0, 31]

Quindi se lo spazio per i numerali è limitato, altrettanto limitato è l'insieme dei numeri rappresentati dai numerali

Quanti bit servono per rappresentare 64 256 796 ? 1025 ?

7 $[0, (2^7 -1)]$

9 $[0, (2^9 -1)]$

ok, con questo numero di cifre ce la facciamo

basi

NB con n cifre si rappresentano b^n numeri naturali in base b da 0 a $(b^n - 1)$ … $[0, (b^n - 1)]$

base 2

```
      cifre          0 1
      contiamo …     0   1
                     10  11
                     100 101 110 111
                     1000 1001 1010 1011 1100 1101 1110 1111
```

con 32 bit si rappresentano i numeri naturali in $[0, (2^{32} - 1)]$

con 5 bit si rappresentano i numeri naturali in $[0, 31]$

Quindi se lo spazio per i numerali è limitato, altrettanto limitato è l'insieme dei numeri rappresentati dai numerali

Quanti bit servono per rappresentare	796	1025	2047?
	10	11	
	$[0, (2^{10} - 1)]$	$[0, (2^{11} - 1)]$	

basi

NB con n cifre si rappresentano b^n numeri naturali in base b da 0 a $(b^n -1)$ … $[0, (b^n -1)]$

base 2

```
      cifre          0 1
      contiamo …     0   1
                     10  11
                     100 101 110 111
                     1000 1001 1010 1011 1100 1101 1110 1111
```

con 32 bit si rappresentano i numeri naturali in $[0, (2^{32} -1)]$

con 5 bit si rappresentano i numeri naturali in [0, 31]

Quindi se lo spazio per i numerali è limitato, altrettanto limitato è l'insieme dei numeri rappresentati dai numerali

Quanti bit servono per rappresentare	1025	2047
	11	11
	$[0, (2^{11} -1)]$	$[0, (2^{11} -1)]$

operazioni in binario (+)

Gli algoritmi sono i medesimi nei vari sistemi posizionali … ma
1 + 1 = 0 con resto 1

5 bit

NB

SOMMA

```
        1
0 0 1 0 1    (5)10
0 1 0 0 1    (9)10
---------
0 1 1 1 0    (14)10
```

```
   1   1
  1 0 1 0 1    (21)10
  0 1 1 1 0
  ---------
1 0 0 0 1 1
  overflow
```

1 + 1 = 0 con riporto di 1
sulla cifra vicina a sinistra

operazioni in binario (-)

Gli algoritmi sono i medesimi nei vari sistemi posizionali … ma
1 + 1 = 0 con resto 1
0 – 1 = 1 con prestito

5 bit

NB

SOMMA

```
      1
0 0 1 0 1      (5)10
0 1 0 0 1      (9)10
---------
0 1 1 1 0      (14)10
```

```
    1   1
  1 0 1 0 1      (21)10
  0 1 1 1 0
  ---------
1 0 0 0 1 1
  overflow
```

5 bit

SOTTRAZIONE

```
      10
0 1 0 0 1
0 0 1 0 1
---------
0 0 1 0 0      (4)10
```

0 - 1 = 1 con prestito di 1 dalla cifra vicina a sinistra

(dopo il "prestito" lo 0 diventa 10 e 10 - 1 = 1

operazioni in binario

Gli algoritmi sono i medesimi nei vari sistemi posizionali …

5 bit
moltiplicazione

```
        0 1 0 0 1
        0 0 1 0 1
        ---------
        0 1 0 0 1
      0 0 0 0 0
    0 1 0 0 1
  0 0 0 0 0
0 0 0 0 0
-----------------
0 0 0 1 0 1 1 0 1
```

operazioni in binario

> Gli algoritmi sono i medesimi nei vari sistemi posizionali …

5 bit
moltiplicazione

```
         0 1 0 0 1
         0 0 1 0 1
         ---------
         0 1 0 0 1
       0 0 0 0 0
     0 1 0 0 1
   0 0 0 0 0
 0 0 0 0 0
 -----------------
 0 0 0 1 0 1 1 0 1
```

= … ☺ **calcolo del numero decimale a partire dalla rappresentazione binaria**

operazioni in binario

5 bit
moltiplicazione

```
          0 1 0 0 1
          0 0 1 0 1
          ---------
          0 1 0 0 1
        0 0 0 0 0
      0 1 0 0 1
    0 0 0 0 0
  0 0 0 0 0
  -----------------
  0 0 0 1 0 1 1 0 1
```

$= 1x2^5 + 0x2^4 + 1x2^3 + 1x2^2 + 0x2^1 + 1 = 45$

calcolo del nunmero decimale a partire dalla rappresentazione binaria

operazioni in binario

5 bit
moltiplicazione

```
          0 1 0 0 1
          0 0 1 0 1
          ---------
          0 1 0 0 1
        0 0 0 0 0
      0 1 0 0 1
    0 0 0 0 0
  0 0 0 0 0
  -----------------
  0 0 0 1 0 1 1 0 1
```

calcolo del nunmero decimale a partire dalla rappresentazione binaria

$= 1x2^5 + 0x2^4 + 1x2^3 + 1x2^2 + 0x2^1 + 1 = 45$

operazioni in binario

5 bit
moltiplicazione

```
                0 1 0 0 1
                0 0 1 0 1
                ---------
                0 1 0 0 1
              0 0 0 0 0
            0 1 0 0 1
          0 0 0 0 0
        0 0 0 0 0
        -----------------
        0 0 0 1 0 1 1 0 1
```

$= 1x2^5 + 0x2^4 + 1x2^3 + 1x2^2 + 0x2^1 + 1 = 45$

calcolo del nunmero decimale a partire dalla rappresentazione binaria

operazioni in binario

5 bit
moltiplicazione

```
          0 1 0 0 1
          0 0 1 0 1
          ---------
          0 1 0 0 1
        0 0 0 0 0
      0 1 0 0 1
    0 0 0 0 0
  0 0 0 0 0
  -----------------
  0 0 0 1 0 1 1 0 1
```

$= 1x2^5 + 0x2^4 + 1x2^3 + 1x2^2 + 0x2^1 + 1 = 45$

calcolo del nunmero decimale a partire dalla rappresentazione binaria

operazioni in binario

5 bit
moltiplicazione

```
              0 1 0 0 1
              0 0 1 0 1
              ---------
              0 1 0 0 1
            0 0 0 0 0
          0 1 0 0 1
        0 0 0 0 0
      0 0 0 0 0
      -----------------
      0 0 0 1 0 1 1 0 1
```

$= 1x2^5 + 0x2^4 + 1x2^3 + 1x2^2 + 0x2^1 + 1 = 45$

calcolo del nunmero decimale a partire dalla rappresentazione binaria

operazioni in binario

5 bit
moltiplicazione

```
          0 1 0 0 1
          0 0 1 0 1
          ---------
          0 1 0 0 1
        0 0 0 0 0
      0 1 0 0 1
    0 0 0 0 0
  0 0 0 0 0
  -----------------
  0 0 0 1 0 1 1 0 1
```

$= 1x2^5 + 0x2^4 + 1x2^3 + 1x2^2 + 0x2^1 + 1 = 45$

calcolo del nunmero decimale a partire dalla rappresentazione binaria

operazioni in binario

5 bit
moltiplicazione

```
          0 1 0 0 1
          0 0 1 0 1
          ---------
          0 1 0 0 1
        0 0 0 0 0
      0 1 0 0 1
    0 0 0 0 0
  0 0 0 0 0
  -----------------
  0 0 0 1 0 1 1 0 1
```

calcolo del nunmero decimale a partire dalla rappresentazione binaria

$= 0x2^8 + 0x2^7 + 0x2^6 + 1x2^5 + 0x2^4 + 1x2^3 + 1x2^2 + 0x2^1 + 1 = 45$

operazioni in binario

Gli algoritmi sono i medesimi nei vari sistemi posizionali …

5 bit
divisione

```
1 0 1 1 | 10
        |-------
        |
```

operazioni in binario

Gli algoritmi sono i medesimi nei vari sistemi posizionali …

5 bit
divisione

```
1 0 1 1 | 10
        |------
        |
```

```
1 0 1 1 | 10
 -      |------
        | 1
```

operazioni in binario

Gli algoritmi sono i medesimi nei vari sistemi posizionali …

5 bit
divisione

```
1 0 1 1 | 10
        |-----
```

```
1 0 1 1 | 10
 -      |-----
        | 1
```

```
1 0 1 1 | 10
 -   1  |-----
        | 1
```

operazioni in binario

Gli algoritmi sono i medesimi nei vari sistemi posizionali …

5 bit
divisione

```
 1 0 1 1  | 10
   -  1   |----
      1   | 10
```

operazioni in binario

Gli algoritmi sono i medesimi nei vari sistemi posizionali …

5 bit
divisione

```
1 0 1 1  | 10
 -  1    |----
    1    | 10
```

```
1 0 1 1  | 10
 -  1    |----
    1 1  | 10
```

```
1 0 1 1  | 10
 -  1    |----
    1 1  | 101
      1  |
```

operazioni in binario

Gli algoritmi sono i medesimi nei vari sistemi posizionali …

5 bit
divisione

```
1 0 1 1 | 10
 -  1   |-----
    1 1 | 101
      1 |
```

```
1 0 1 1   | 10
 -  1     |-----
    1 1   | 101.
      1 0 |
```

```
1 0 1 1   | 10
 -  1     |-----
    1 1   | 101.1
      1 0 |
       -  |
```

operazioni in binario

Gli algoritmi sono i medesimi nei vari sistemi posizionali …

5 bit
divisione

```
1 0 1 1  | 10
 -  1    |------
    1 1  | 101.1
      10 |
```

1011 / 10   è 101.1

è vero?   ☺

operazioni in binario

Gli algoritmi sono i medesimi nei vari sistemi posizionali …

5 bit
divisione

```
1 0 1 1   | 10
 -  1     |------
    1 1   | 101.1
      10  |
```

$1011 = 1x2^3 + 0x2^2 + 1x2 + 1 = 11$

$101.1 = 1x2^2 + 0x2^1 + 1 + 1x2^{-1} = 5.5$

11/2=5.5 effettivamente

e 101.1 moltiplicato 10 quanto fa? ☺

Conversione $(n)_{10} \dashrightarrow (\ldots)_2$

Il procedimento è quello per divisioni successive (divisioni per la base 2). Ogni divisione fornisce un resto, che è la cifra nella rappresentazione di arrivo (binaria)

$(796)_{10} = (\ ?\ )_2$

796/2 =	398	Resto	0
/2	199		0
/2	99		1
/2	49		1
/2	24		1
/2	12		0
/2	6		0
/2	3		0
/2	1		1
/2	0		1

ordine delle cifre

$(796)_{10} = (\ 1100011100\ )_2$

peso 512
peso 256
16
8
4

Conversione $(n)_{10}$ --> $(.)_2$ per parte frazionaria

Il procedimento è per moltiplicazioni successive (per la base 2).
Se il risultato è >=1 la cifra corrispondente nel numero di arrivo è 1;
altrimenti è zero. Si prosegue con la parte frazionaria, fino a ottenere 0,
o a stancarsi

$(0.25)_{10} = (?)_2$

.25x2 =	.5	parte intera	0
.5x2	1	" "	1
0		fine	

ordine delle cifre

$(0.25)_{10} = (0.01)_2$

2^{-2}

$(0.23)_{10} = (?)_2$

Conversione $(n)_{10}$ --> $(.)_2$ per parte frazionaria

Il procedimento è per moltiplicazioni successive (per la base 2).
Se il risultato è >=1 la cifra corrispondente nel numero di arrivo è 1; altrimenti è zero. Si prosegue con la parte frazionaria, fino a ottenere 0, o a verificare l'esistenza di un periodo, **o a stancarsi(*)**

$(0.23)_{10} = (?)_2$

.23x2 =	.46	0
.46x2	.92	0
.92x2	1.84	1
.84x2	1.68	1

☺

ordine delle cifre

0.92X2 è 1.84:
la cifra 1 viene usata per continuare a costruire il numerale, mentre la parte frazionaria (0,84) viene usata per continuare il processo di moltiplicazioni successive

Conversione $(n)_{10}$ --> $(.)_2$ per parte frazionaria

Il procedimento è per moltiplicazioni successive (per la base 2).
Se il risultato è >=1 la cifra corrispondente nel numero di arrivo è 1; altrimenti è zero. Si prosegue con la parte frazionaria, fino a ottenere 0, o a verificare l'esistenza di un period, **o a stancarsi**(*)

$(0.23)_{10} = (?)_2$

.23x2 =	.46	0
.46x2	.92	0
.92x2	1.84	1
.84x2	1.68	1
.68x2	1.36	1
.36x2	.72	0
.72x2	1.44	1
.44x2	.88	0
.88x2	1.76	1
.76x2	1.52	1
.52x2	1.04	1
uff	fine(*) ...	

0.92X2 è 1.84: la cifra 1 viene usata per continuare a costruire il numerale, mentre la parte frazionaria (0,84) viene usata per continuare il processo di moltiplicazioni successive

ordine delle cifre

$(0.23)_{10} = (0.00111010111)_2$ rappresentazione esatta in decimale ma approssimata in binario

(*) nel senso che si è raggiunta la precisione voluta e non si vogliono calcolare altre cifre

Conversione $(n)_{10}$ --> $(.)_2$ per parte frazionaria

Il procedimento è per moltiplicazioni successive (per la base 2).
Se il risultato è >=1 la cifra corrispondente nel numero di arrivo è 1; altrimenti è zero. Si prosegue con la parte frazionaria, fino a ottenere 0, **o a stancarsi**

$(0.23)_{10} = (?)_2$

.23x2 =	.46	0
.46x2	.92	0
.92x2	1.84	1
.52x2	1.04	1
uff	fine	

delle cifre

QUINDI

$(796.25)_{10} = (1100011100.01)_2$

$(796.23)_{10} = (1100011100.00111010111)_2$ circa

$(0.25)_{10} = (0.00111010111)_2$ rappresentazione esatta in decimale ma approssimata in binario

Conversione $(n)_2$ <--> $(.)_8$

Fantastico! Le cifre binarie corrispondono a quelle ottali, just prese a gruppi di 3.
E otto è due alla terza!

Es.

$(796)_{10} = (1\ 1\ 0\ 0\ 0\ 1\ 1\ 1\ 0\ 0)_2$

gruppi di tre da destra verso sinistra; se mancano cifre si aggiungono zeri a sinistra

$= (001\ 100\ 011\ 100)_2 = (1\ 4\ 3\ 4)_8$

base 16

NB con n cifre si rappresentano b^n numeri naturali in base b da 0 a $(b^n - 1)$ … $[0, (b^n - 1)]$

base 16

```
        cifre          0 1 2 3 4 5 6 7 8 9 A B C D E F
contiamo …   0  1  2  3  4  5  6  7  8  9  A  B  C  D  E  F
             10 11 12 13 14 15 16 17 18 19 1A 1B 1C 1D 1E 1F
             20 …     F1 F2 … FF
```

sedici

con due cifre, al più
$15 \times 16 + 15 = 16^2 - 1$

rispetto a 10, 8, 2 … si hanno numerali di solito più corti, dato che ci sono più cifre a disposizione

$(15)_{10} = (1111)_2 = F_{16}$

base 16

NB con n cifre si rappresentano b^n numeri naturali in base b da 0 a $(b^n - 1)$ … $[0, (b^n - 1)]$

base 16

```
      cifre       0 1 2 3 4 5 6 7 8 9 A B C D E F
contiamo …  0  1  2  3  4  5  6  7  8  9  A  B  C  D  E  F
            10 11 12 13 14 15 16 17 18 19 1A 1B 1C 1D 1E 1F
            20 …    F1 F2 … FF
```

sedici

con due cifre, al più $15 \times 16 + 15 = 16^2 - 1$

rispetto a 10, 8, 2 … si hanno numerali di solito più corti, dato che ci sono più cifre a disposizione

$(15)_{10} = (1111)_2 = F_{16}$

Conversione $(n)_2$ <---> $(\ .\ )_{16}$

Fantastico! … gruppi di 4. E sedici è due alla quarta!

Es.

$(796)_{10} = (1\ 1\ 0\ 0\ 0\ 1\ 1\ 1\ 0\ 0)_2$

$= (0011\ 0001\ 1100)_2 = (3\ 1\ C)_{16}$

Tecniche della Programmazione, lez. 11 seconda parte - anticipazione

Rappresentazione dei numeri

- numeri interi: complemento a 2

Remember

con N cifre binarie (N bit) si possono scrivere i numeri naturali da 0 a 2^N-1

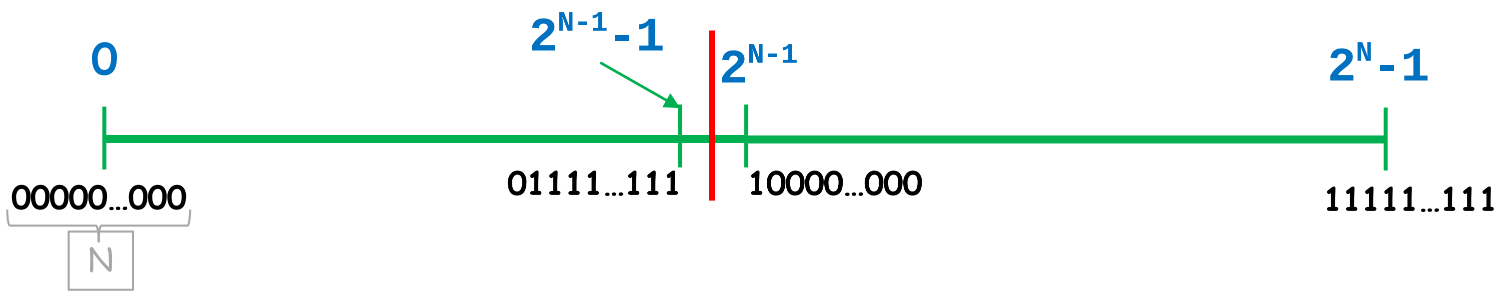
intervallo di rappresentabilita` per i numeri naturali espressi in forma binaria, con N cifre (N bit):

$$[0, 2^N-1]$$

Remember

intervallo di rappresentabilita` per i numeri naturali espressi in forma binaria, con N cifre (N bit):

$$[0, 2^N-1]$$



ESEMPIO : Numerali con 4 bit

0 0 0 0	0
0 0 0 1	1
0 0 1 0	2
0 0 1 1	3
0 1 0 0	4
0 1 0 1	5
0 1 1 0	6
0 1 1 1	7
1 0 0 0	8
1 0 0 1	9
1 0 1 0	10
1 0 1 1	11
1 1 0 0	12
1 1 0 1	13
1 1 1 0	14
1 1 1 1	15

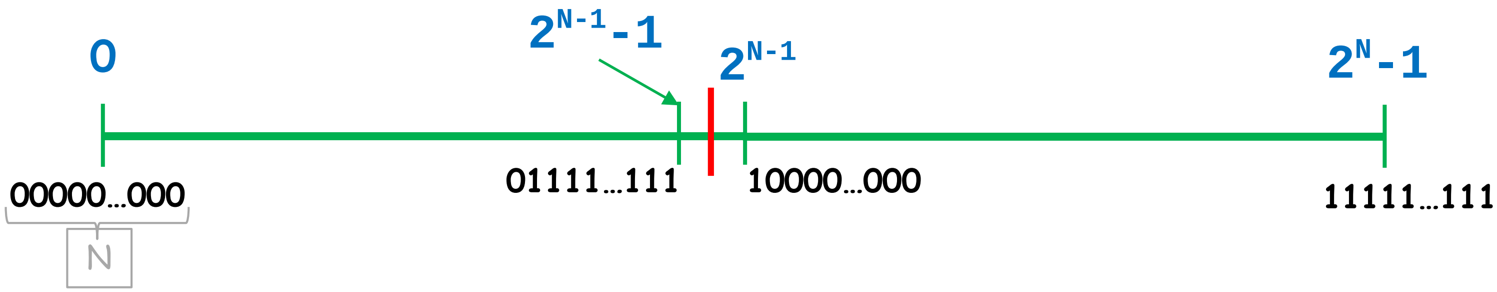
con N = 4, abbiamo 2^4 numerali, cioè 16 da 0000 a 1111 che possono rappresentare i numeri naturali da 0 a 2^4-1 cioè l'intervallo [0, 15]

☺ replicare questa slide, intera, per N=5

$[0, 2^5-1] = [0, 32-1] = [0. 31]$

Remember

con N cifre binarie (N bit) si possono scrivere i numeri naturali da 0 a 2^N-1



Se N = 32, allora abbiamo 2^{32} numerali,

da 00000000000000000000000000000000 a 11111111111111111111111111111111

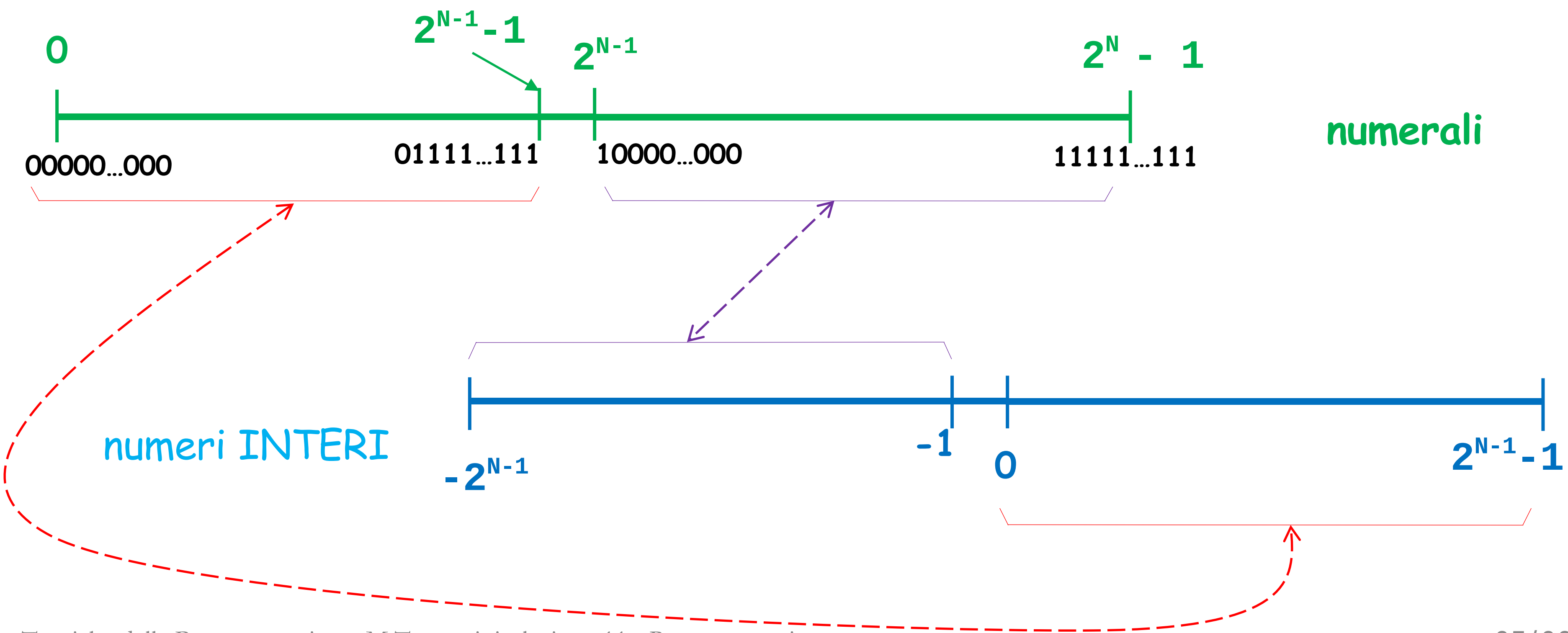
che possono rappresentare i numeri naturali
da 0 a 2^{32}-1
cioè l'intervallo [0, 2^{32}-1]

Rappresentazione dei numeri interi in *Complemento a 2*

Solo che non si può vivere solo con i naturali ...
Vorremmo rappresentare, **con i numerali disponibili**, quanti più numeri INTERI possibile, un pò positivi e un pò negativi (diciamo metà e metà)

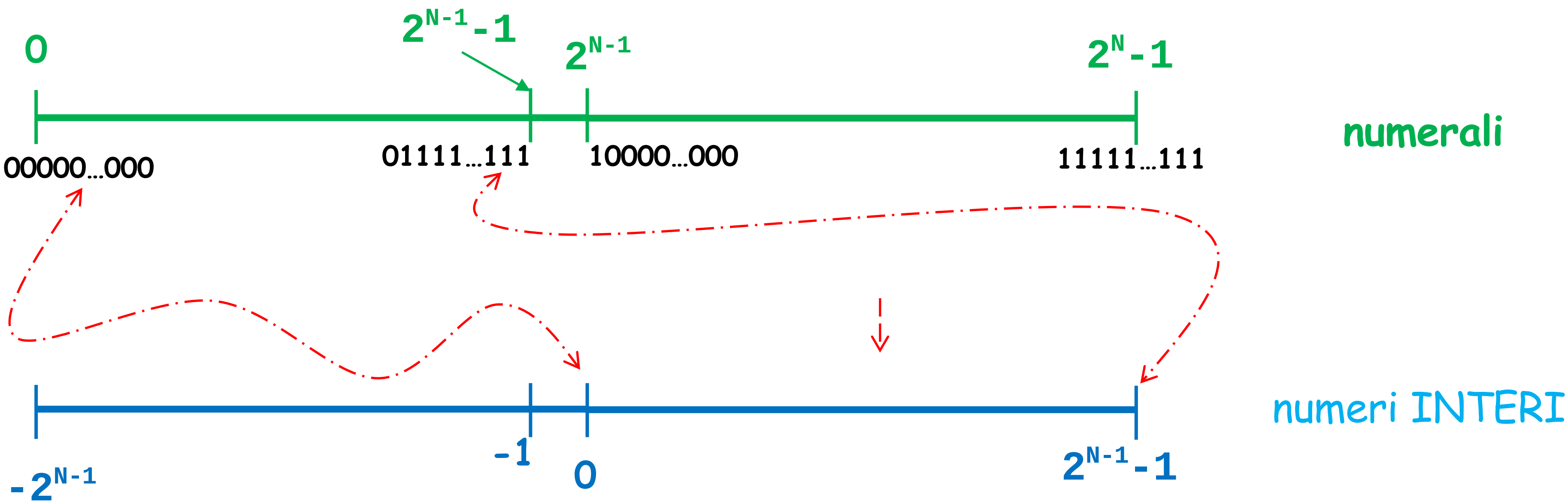
Con N cifre binarie (N bit) i **numerali disponibili** sono 2^N :

- **ne usiamo metà per i numeri negativi (sono 2^{N-1} numerali)**
- **usiamo gli altri 2^{N-1} numerali per i numeri positivi e per lo zero**



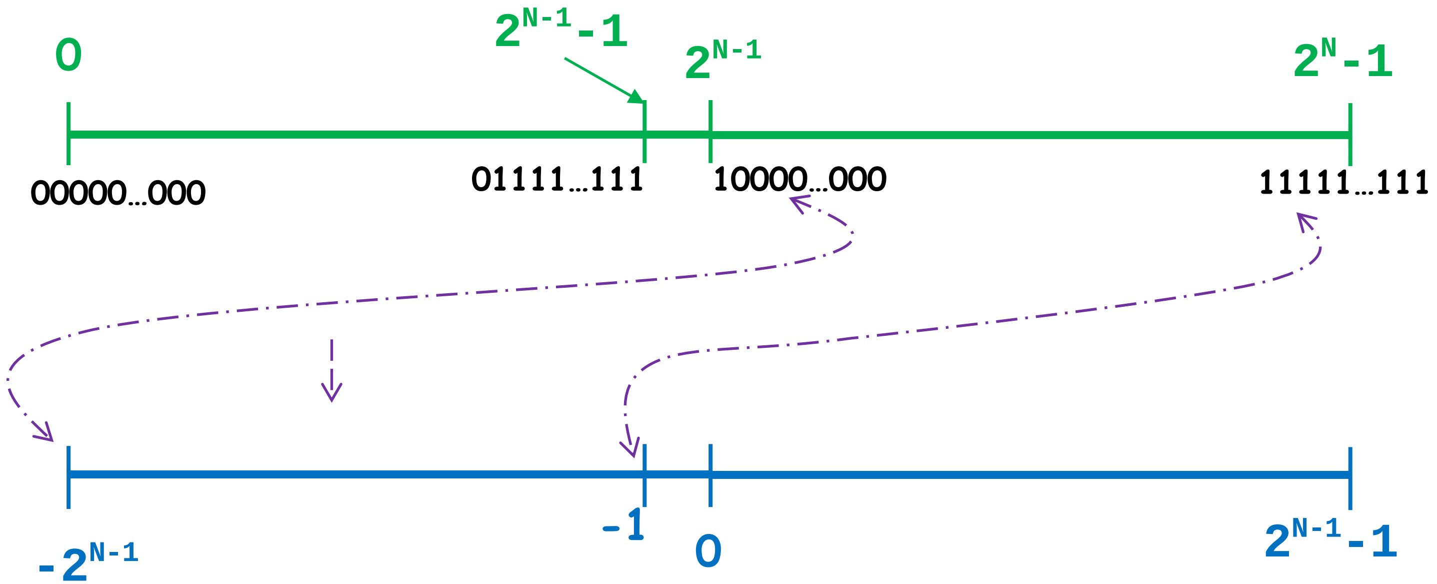
Rappresentazione in *Complemento a 2* (Tipo *int*)

dettaglio su alcuni numeri / numerali estremi ...



Rappresentazione in *Complemento a 2 (Tipo `int`)*

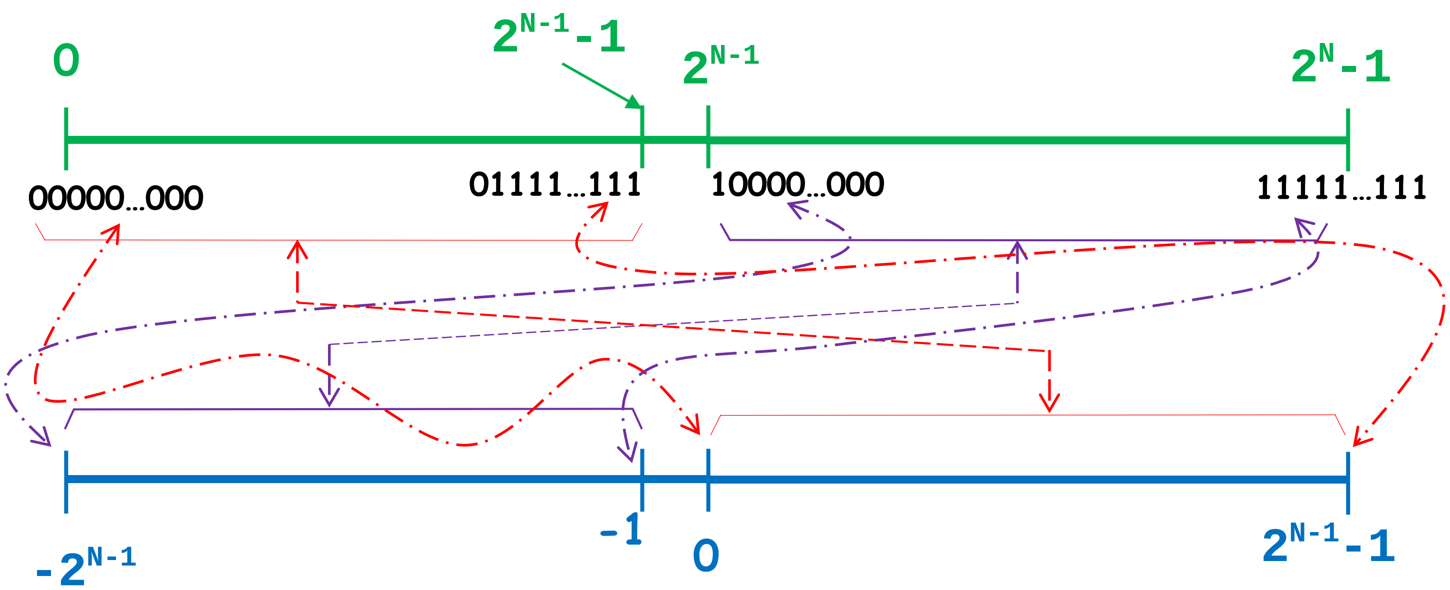
dettaglio su alcuni numeri / numerali estremi ...



Rappresentazione in *Complemento a 2 (Tipo int)*

Solo che non si può vivere solo con i naturali …
Vorremmo rappresentare, con i numerali disponibili, quanti più numeri INTERI possibile, un pò positivi e un pò negativi (diciamo metà e metà)

Con N cifre binarie (N bit) i numerali disponibili sono 2^N :
- **ne usiamo metà per i numeri negativi (sono 2^{N-1} numerali)**
- **usiamo gli altri 2^{N-1} numerali per i numeri positivi e per lo zero**



Con N = 32, i 2^{32} numerali rappresentano i numeri interi nell'intervallo $[-2^{31}, 2^{31}-1]$ (intervallo di rappresentabilità)

Rappresentazione in *Complemento a 2 (Tipo int)*

Solo che non si può vivere solo con i naturali ...
Vorremmo rappresentare, con i numerali disponibili, quanti più numeri INTERI possibile, un pò positivi e un pò negativi (diciamo metà e metà)

Con N cifre binarie (N bit) i **numerali disponibili** sono 2^N :
- **ne usiamo metà per i numeri negativi (sono 2^{N-1} numerali)**
- **usiamo gli altri 2^{N-1} numerali per i numeri positivi e per lo zero**

Con N = 32, i 2^{32} numerali rappresentano i numeri interi nell'intervallo $[-2^{31}, 2^{31}-1]$ (intervallo di rappresentabilità)

NB
Se N e` il numero di bit usati per i numerali, l'intervallo di rappresentabilita` dei numeri interi in complemento a 2 si esprime come

$$[-2^{N-1}, 2^{N-1}-1]$$

Tipo int (rappresentazione in Complemento a 2)

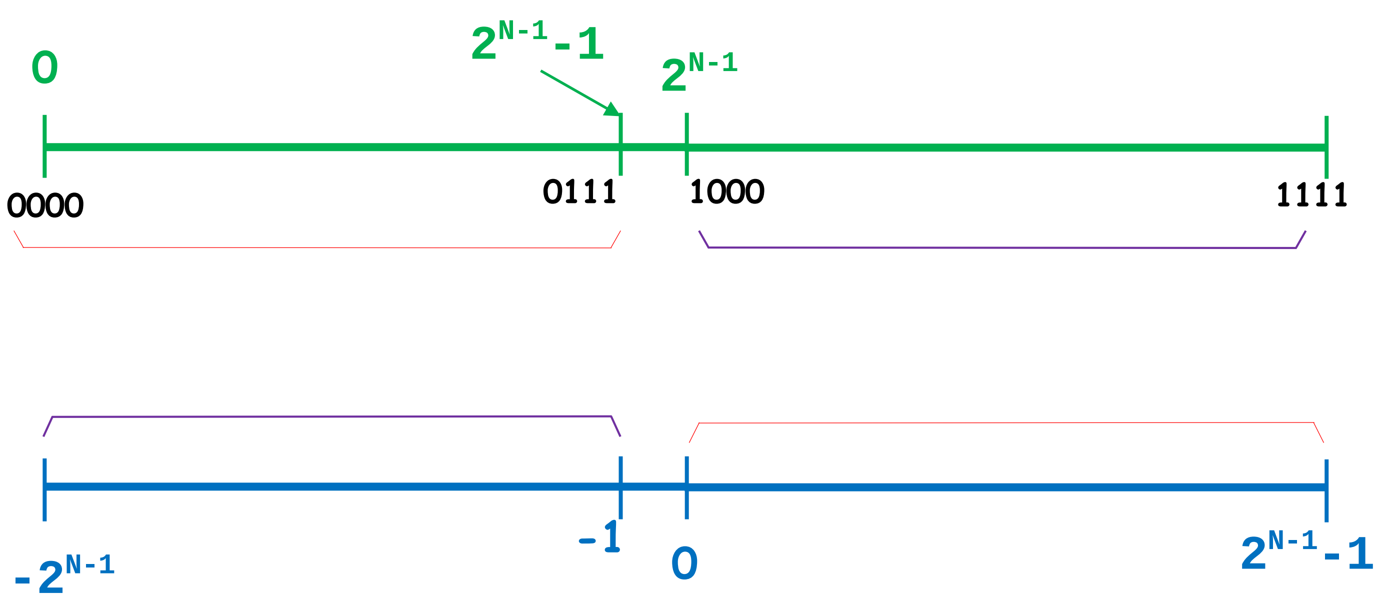
Solo che non si può vivere solo con i naturali …
Vorremmo rappresentare, con i numerali disponibili, quanti più numeri INTERI possibile, un pò positivi e un pò negativi (diciamo metà e metà)

Con N cifre binarie (N bit) i numerali disponibili sono 2^N :
- **ne usiamo metà per i numeri negativi (sono 2^{N-1} numerali)**
- **usiamo gli altri 2^{N-1} numerali per i numeri positivi e per lo zero**

ESEMPIO: Numerali con 4 bit

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

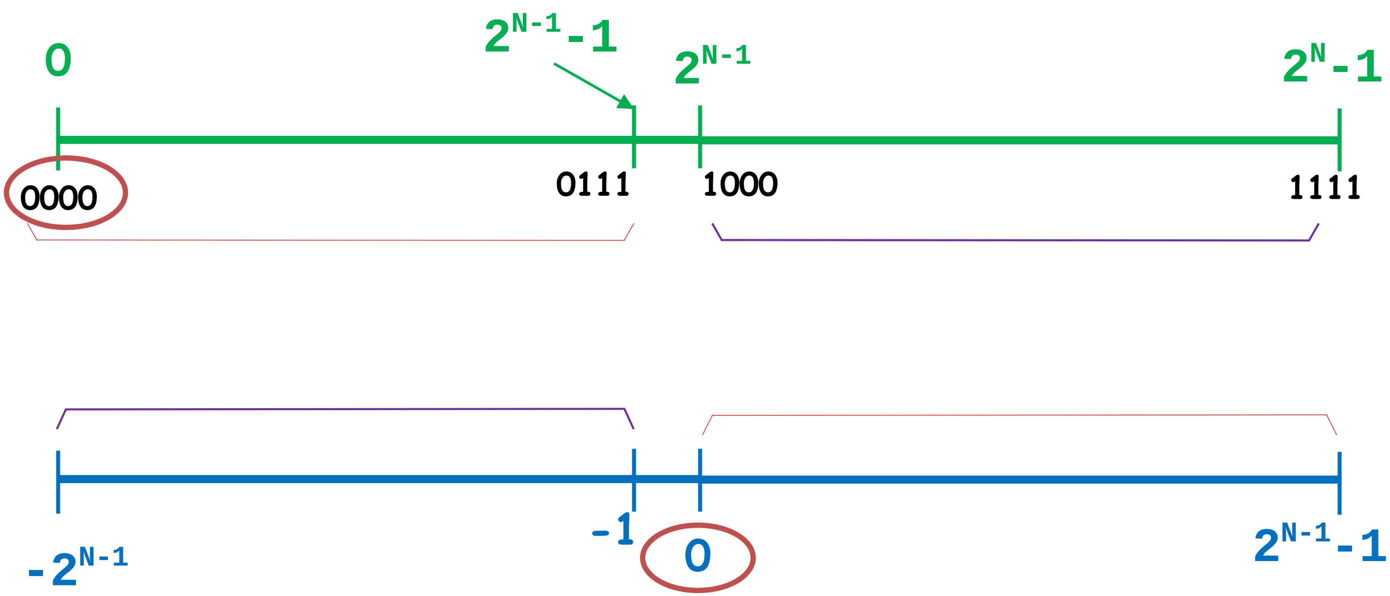
Con N = 4, i 2^4 numerali rappresentano i numeri interi nell'intervallo $[-2^3\ ,\ 2^3-1] = [-8, 7]$



Tipo int (rappresentazione in Complemento a 2)

Solo che non si può vivere solo con i naturali …
Vorremmo rappresentare, con i numerali disponibili, quanti più numeri INTERI possibile, un pò positivi e un pò negativi (diciamo metà e metà)

Con N cifre binarie (N bit) i numerali disponibili sono 2^N :
- **ne usiamo metà per i numeri negativi (sono 2^{N-1} numerali)**
- **usiamo gli altri 2^{N-1} numerali per i numeri positivi e per lo zero**



Con N = 4, i 2^4 numerali rappresentano i numeri interi nell'intervallo $[-2^3 , 2^3-1]$

ESEMPIO: Numerali con 4 bit

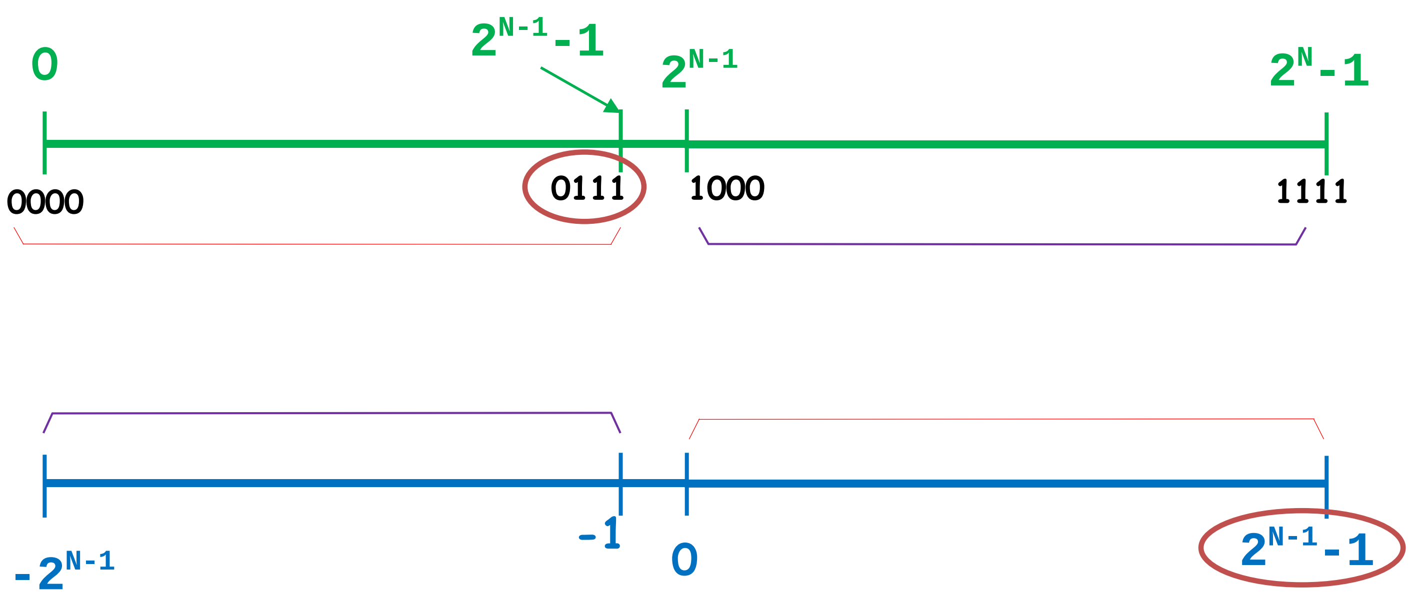
	naturali	interi
0 0 0 0	0	0
0 0 0 1	1	1
0 0 1 0	2	2
0 0 1 1	3	3
0 1 0 0	4	4
0 1 0 1	5	5
0 1 1 0	6	6
0 1 1 1	7	7
1 0 0 0	8	-8
1 0 0 1	9	-7
1 0 1 0	10	-6
1 0 1 1	11	-5
1 1 0 0	12	-4
1 1 0 1	13	-3
1 1 1 0	14	-2
1 1 1 1	15	-1

Tipo int (rappresentazione in Complemento a 2)

Solo che non si può vivere solo con i naturali ...
Vorremmo rappresentare, con i numerali disponibili, quanti più numeri INTERI possibile, un pò positivi e un pò negativi (diciamo metà e metà)

Con N cifre binarie (N bit) i numerali disponibili sono 2^N :

- **ne usiamo metà per i numeri negativi (sono 2^{N-1} numerali)**
- **usiamo gli altri 2^{N-1} numerali per i numeri positivi e per lo zero**



ESEMPIO: Numerali con 4 bit

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

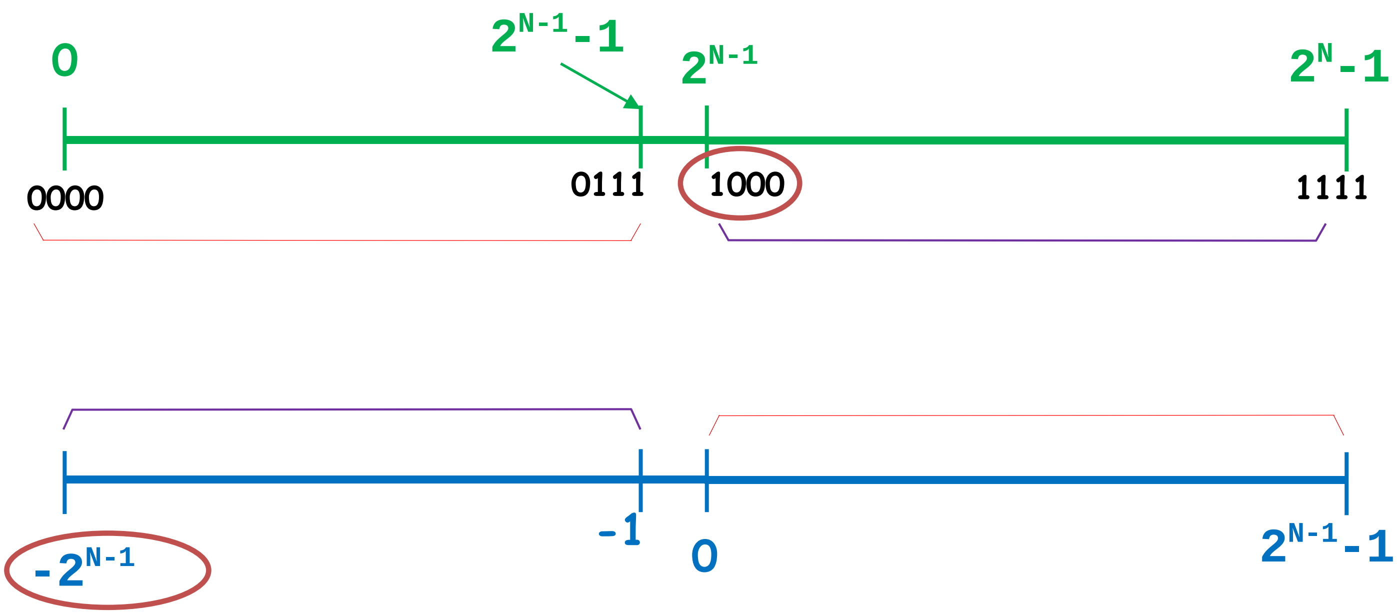
Con N = 4, i 2^4 numerali rappresentano i numeri interi nell'intervallo $[-2^3-, 2^3-1]$

Tipo int (rappresentazione in Complemento a 2)

Solo che non si può vivere solo con i naturali ...
Vorremmo rappresentare, con i numerali disponibili, quanti più numeri INTERI possibile, un pò positivi e un pò negativi (diciamo metà e metà)

Con N cifre binarie (N bit) i **numerali disponibili** sono 2^N :

- **ne usiamo metà per i numeri negativi (sono 2^{N-1} numerali)**
- **usiamo gli altri 2^{N-1} numerali per i numeri positivi e per lo zero**



ESEMPIO: Numerali con 4 bit

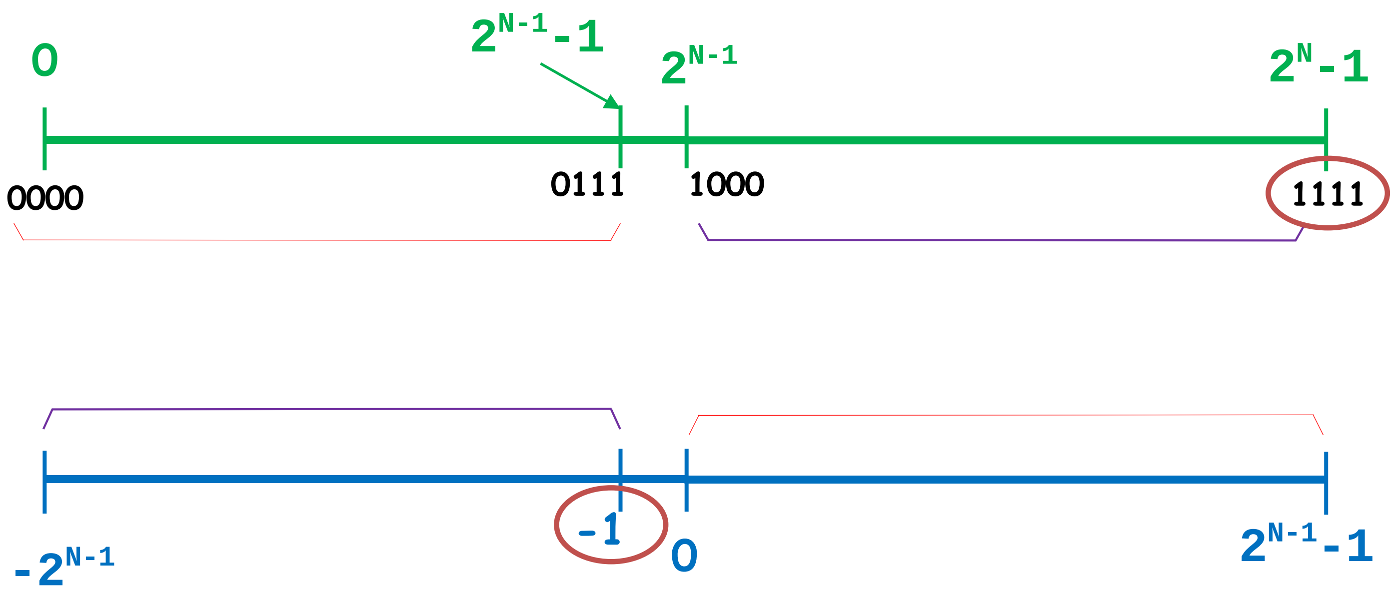
	naturali	interi
0000	0	0
0001	1	1
0010	2	2
0011	3	3
0100	4	4
0101	5	5
0110	6	6
0111	7	7
1000	8	-8
1001	9	-7
1010	10	-6
1011	11	-5
1100	12	-4
1101	13	-3
1110	14	-2
1111	15	-1

Con N = 4, i 2^4 numerali rappresentano i numeri interi nell'intervallo [-2^3-, 2^3-1]

Tipo int (rappresentazione in Complemento a 2)

Solo che non si può vivere solo con i naturali ...
Vorremmo rappresentare, con i numerali disponibili, quanti più numeri INTERI possibile, un pò positivi e un pò negativi (diciamo metà e metà)

Con N cifre binarie (N bit) i **numerali disponibili** sono 2^N :
- **ne usiamo metà per i numeri negativi (sono 2^{N-1} numerali)**
- **usiamo gli altri 2^{N-1} numerali per i numeri positivi e per lo zero**

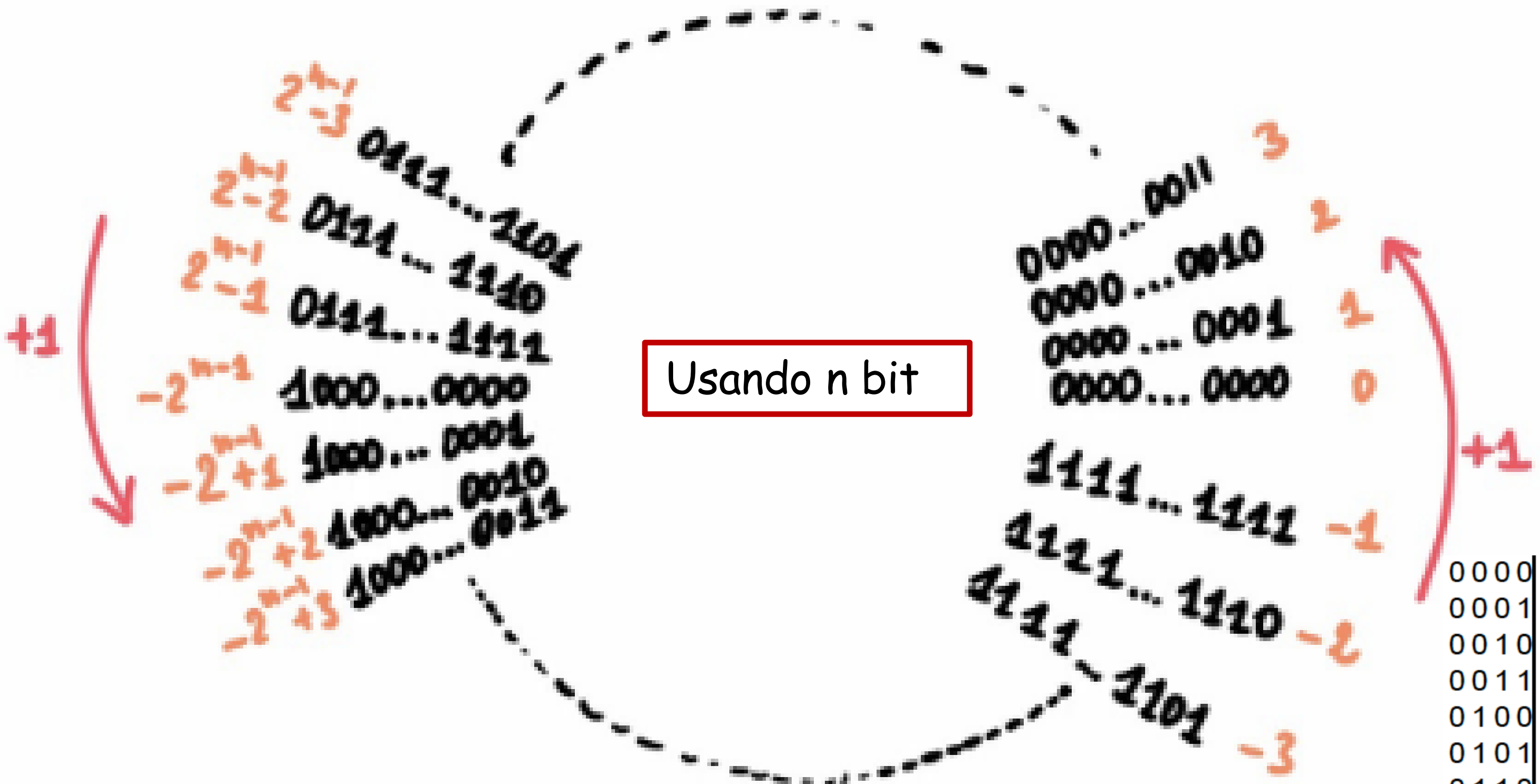


Con N = 4, i 2^4 numerali rappresentano i numeri interi nell'intervallo [-2^3-, 2^3-1]

ESEMPIO: Numerali con 4 bit

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

Tipo int: circolarità della rappresentazione



0 0 0 0	0	0
0 0 0 1	1	1
0 0 1 0	2	2
0 0 1 1	3	3
0 1 0 0	4	4
0 1 0 1	5	5
0 1 1 0	6	6
0 1 1 1	7	7
1 0 0 0	8	-8
1 0 0 1	9	-7
1 0 1 0	10	-6
1 0 1 1	11	-5
1 1 0 0	12	-4
1 1 0 1	13	-3
1 1 1 0	14	-2
1 1 1 1	15	-1