Rappresentazione di numeri naturali

Fondamenti di Informatica I
Corso di laurea in Ingegneria Informatica e Automatica
Sapienza Università di Roma

A.A. 2018-19

Domenico Lembo, Paolo Liberatore,
Alberto Marchetti Spaccamela, Marco Schaerf

Rappresentazione nei calcolatori

Nei calcolatori, tutti i dati sono rappresentati con numeri:

- ogni lettera è un numero
- ogni colore è una terna di numeri
- ogni intensità di suono è un numero
- …

A loro volta i numeri sono rappresentati *in forma binaria*, ovvero come sequenze di zeri e uni.

Si tratta di un meccanismo di rappresentazione simile a quello decimale, in cui si usano però solo due cifre (0 e 1) invece di dieci (0, 1, 2, 3, 4, 5, 6, 7, 8 e 9).

Sequenza di numeri

Costruiamo la sequenza dei numeri naturali (interi non negativi) usando 10 cifre

0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20
21 … … 98 99 100 101 … …

Metodo:

- si inizia con le cifre a disposizione (0 … 9)
- quando sono finite, si mette 1 all'inizio (10 11 12 …)
- quando finiscono di nuovo, si passa a 2 (20 21 22 …)
- quando finiscono di nuovo, si passa a 3 (30 31 32 …)
- …

Lo stesso sistema si può usare anche con meno cifre

Sequenze rappresentate con otto cifre

Un sistema usato abbastanza spesso prevede l'uso di otto cifre

0, 1, 2, 3, 4, 5, 6, 7

La sequenza dei numeri viene generata con lo stesso sistema:

0 1 2 3 4 5 6 7 10 11 12 13 14 15 16 17 20
21 … … 76 77 100 101

cambia solo il numero di cifre a disposizione.

Confronto fra le due rappresentazioni

I numeri che rappresentano le due sequenze sono gli stessi: gli interi non negativi.

Dieci cifre	Otto cifre	numero
0	0	zero
1	1	uno
2	2	due
3	3	tre
4	4	quattro
5	5	cinque
6	6	sei
7	7	sette
8	10	otto
9	11	nove
10	12	dieci
11	13	undici
12	14	dodici

Vari sistemi di uso comune

- **Decimale**

  dieci cifre (0, 1, 2, 3, 4, 5, 6, 7, 8 e 9)
- **Ottale**

  otto cifre (0, 1, 2, 3, 4, 5, 6, e 7)
- **Binario**

  due cifre (0 e 1)
- **Esadecimale**

  sedici cifre (0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F)

Sistema binario

La sequenza indica sempre i numeri in ordine:

0	zero
1	uno
10	due
11	tre
100	quattro
101	cinque
110	sei
111	sette
…	…
1000010000111	quattromiladuecentotrentuno
…	…

Ovviamente la sequenza non termina, prima o poi incontriamo il numero che vogliamo rappresentare.

Necessità di conversioni fra rappresentazioni

- Nei calcolatori il sistema binario viene usato perchè è facile dal punto di vista tecnologico realizzare sistemi per memorizzare e manipolare dati rappresentati solo come zeri e uni.
- Questo però crea un problema: passare dalla rappresentazione comune in decimale a quella binaria e viceversa.
- Per esempio, nel programma della calcolatrice l'utente digita i numeri in decimale, è poi il programma che li converte in binario, effettua i calcoli e riconverte il risultato in decimale.
- Per fare questo, è necessario chiarire il funzionamento matematico di queste rappresentazioni, che vengono dette **posizionali**.
- Questo permetterà di trovare metodi per effettuare conversioni da un numero di cifre a un altro, per esempio da decimale a binario, da ottale a esadecimale, ecc.

Rappresentazioni posizionali

Consideriamo il caso del sistema ottale. Come anche in decimale e binario, le singole cifre rappresentano se stesse, quindi 0 rappresenta zero e 7 rappresenta sette. Da questo punto in poi le cose cambiano.

In ottale, dopo 7 c'è 10, 11, 12

questi numeri rappresentano otto, nove, dieci, ecc.

7	sette		
10	otto	=	otto più zero
11	nove	=	otto più uno
12	dieci	=	otto più due
13	undici	=	otto più tre
14	dodici	=	otto più quattro

...

quindi: 1 in prima posizione indica otto

Rappresentazioni posizionali

…

14	dodici	=	otto più quattro		
15	tredici	=	otto più cinque		
16	quattordici	=	otto più sei		
17	quindici	=	otto più sette		
20	sedici	=	otto più otto più zero	=	due per otto
21	diciassette	=	otto più otto più uno	=	due per otto più uno

…

uno in prima posizione è otto

due è sedici

Lo stesso vale anche nel sistema di *rappresentazione decimale*:

16 è dieci più sei

24 è due per dieci più quattro

Rappresentazioni posizionali

Nel caso di più cifre:

Decimale

zyx è z per cento (dieci per dieci) più y per dieci più x

Ottale

zyx è z per sessantaquattro (otto per otto) più y per otto più x

Binario

zyx è z per quattro (due per due) più y per due più x

È sempre lo stesso sistema, ma al posto di dieci ci può essere otto oppure due a seconda del **numero di cifre usate nel sistema di rappresentazione**.

Questo numero viene detto **base**.

Il sistema con dieci cifre (decimale) usa base dieci, quello con otto cifre (ottale) base otto, quello con due cifre (binario) base due. In tutti i casi:

zyx indica z per base al quadrato più y per base più x

Rappresentazioni posizionali

Questo vale anche per numeri di più cifre. Per esempio:

2315 in base dieci è $2\times10^3 + 3\times10^2 + 1\times10 + 5$

73512 in base otto è: $7\times8^4 + 3\times8^3 + 5\times8^2 + 1\times8 + 2$

101011 in base due è: $1\times2^5 + 0\times2^4 + 1\times2^3 + 0\times2^2 + 1\times2 + 1$

La regola generale è che l'ultima cifra rappresenta il suo valore "normale", quella che la precede è il suo valore per la base, quella ancora prima il suo valore per la base al quadrato, ecc.

Considerando che $b^1=b$ e che $b^0=1$, la regola generale è che il numero

$c_k c_{k-1} \ldots c_1 c_0$ in una base b qualsiasi vale $c_k\times b^k + c_{k-1}\times b^{k-1} + \ldots + c_1\times b^1 + c_0\times b^0$

Nota: Esistono anche sistemi di rappresentazione che non sono posizionali: i numeri romani (I, II, III, IV, ecc.) sono un esempio di rappresentazione non posizionale, in quanto la posizione di una cifra non indica il suo valore: la prima I in II indica uno, in IV indica meno uno (stessa posizione, valore diverso).

Conversioni: passaggio da una rappresentazione ad un'altra

principio della conversione: si usa sempre la definizione di numero:

$c_k c_{k-1} \ldots c_1 c_0$

indica

$c_k \times b^k + c_{k-1} \times b^{k-1} + \ldots + c_1 \times b^1 + c_0 \times b^0$

il valore della somma deve essere lo stesso anche se si cambia base!

Esempio:

$2 \times 10 + 1 = 2 \times 8 + 5 = 1 \times 16 + 0 \times 8 + 1 \times 4 + 0 \times 2 + 1 \times 1$

quindi 21 in decimale è 25 in ottale e 10101 in binario.

Per distinguere le varie rappresentazioni di un numero, nel seguito, in caso di ambiguità, useremo un pedice per indicare la base in cui il numero è rappresentato. Con questa notazione ad esempio possiamo scrivere:

$(21)_{10} = (25)_8 = (10101)_2$

Conversione in decimale

Per convertire un numero da una base qualsiasi a decimale basta applicare la regola sostituendo a b il 10 ed a c_k, c_{k-1}, … le cifre (espresse in decimale).

esempio (ovvio): conversione da decimale a decimale

il valore decimale di $(3284)_{10}$ è:

$$
\begin{array}{lcllll}
(3284)_{10} & = & 3 & 2 & 8 & 4 \\
 & & 1000 & 100 & 10 & 1 \\
 & = & 3\times1000 + 2\times100 + 8\times10 + 4\times1 & & &
\end{array}
$$

Il risultato è ovviamente il numero di partenza (conversione da decimale a decimale).

Conversione da binario a decimale

Nel caso dei numeri in binario, il sistema è lo stesso ma invece di 1, 10, 100, 1000, ecc., si usano le potenze di due: 1, 2, 4, 8, 16, ecc.

L'esempio che segue mostra la conversione del numero binario 10110 in decimale:

$(10110)_2$ = 1 0 1 1 0

16 8 4 2 1

= $1\times16 + 0\times8 + 1\times4 + 1\times2 + 0\times1$

= $16+4+2$

= $(22)_{10}$

Conversione da ottale a decimale

Per i numeri in ottale si usano le potenze di otto: 1, 8, 64, 512, 4096, ecc.

Esempio:

$(42703)_8$ = 4 2 7 0 3

4096 512 64 8 1

= $4\times4096 + 2\times512 + 7\times64 + 0\times8 + 3\times1$

= $16384 + 1024 + 448 + 0 + 3$

= $(17859)_{10}$

Altri esempi

- convertire 101101 da binario a decimale

$1\times2^5 + 0\times2^4 + 1\times2^3 + 1\times2^2 + 0\times2^1 + 1\times2^0 =$

$32 + 0 + 8 + 4 + 0 + 1 = 45$

- convertire 5720 da ottale a decimale

$5\times8^3 + 7\times8^2 + 2\times8^1 + 0\times8^0 =$

$2560 + 448 + 16 + 0 = 3024$

- convertire 43012 da base cinque a decimale

$4\times5^4 + 3\times5^3 + 0\times5^2 + 1\times5^1 + 2\times5^0 =$

$2500 + 375 + 0 + 5 + 2 = 2882$

- convertire 6513 da base sette a decimale

$6\times7^3 + 5\times7^2 + 1\times7^1 + 3\times7^0 =$

$2058 + 245 + 7 + 3 = 2313$

Conversione da decimale ad altra base

In linea di principio potremmo usare lo stesso metodo precedente, ma le operazioni andrebbero fatte nella base di "arrivo", e quindi non in base 10. Vediamo invece un metodo di conversione basato su operazioni nella base di partenza, e quindi 10.

per convertire n da base 10 a base b:

- si calcolano quoziente e resto di n diviso b
- il resto è l'ultima cifra in base b (quella più a destra)
- il quoziente si divide per b, ottenendo un nuovo quoziente e resto
- il resto è la penultima cifra in base b
- il quoziente si divide per b...
- si intera fino a quando il quoziente è zero, che indica che abbiamo terminato

Conversione da decimale a binario

Esempio: $(319)_{10} = (??)_2$

N : b	*Quoziente*	*Resto*	*cifra*
319 : 2	159	1	c_0
159 : 2	79	1	c_1
79 : 2	39	1	c_2
39 : 2	19	1	c_3
19 : 2	9	1	c_4
9 : 2	4	1	c_5
4 : 2	2	0	c_6
2 : 2	1	0	c_7
1 : 2	0	1	c_8

$(319)_{10} = (100111111)_2$

Conversione da decimale a ottale

Esempio: $(319)_{10} = (??)_8$

N : b	*Quoziente*	*Resto*	*cifra*
319 : 8	39	7	c_0
39 : 8	4	7	c_1
4 : 8	0	4	c_2

$(319)_{10} = (477)_8$

Esercizi

1. Dato il numero in base dieci 6842, convertirlo in base due
2. Dato il numero in base dieci 1209, convertirlo in base otto
3. Dato il numero in base sei 4215, convertirlo in base dieci
4. Dato il numero in base dieci 92341, convertirlo in base cinque.

Soluzione Esercizio 1

- 6812 diviso 2 fa 3406 con resto di 0
- 3406 diviso 2 fa 1703 con resto di 0
- 1703 diviso 2 fa 851 con resto di 1
- 851 diviso 2 fa 425 con resto di 1
- 425 diviso 2 fa 212 con resto di 1
- 212 diviso 2 fa 106 con resto di 0
- 106 diviso 2 fa 53 con resto di 0
- 53 diviso 2 fa 26 con resto di 1
- 26 diviso 2 fa 13 con resto di 0
- 13 diviso 2 fa 6 con resto di 1
- 6 diviso 2 fa 3 con resto di 0
- 3 diviso 2 fa 1 con resto di 1
- 1 diviso 2 fa 0 con resto di 1

$(6812)_{10} = (1101010011100)_2$

Soluzione Esercizio 2

- 1209 diviso 8 fa 151 con resto di 1
- 151 diviso 8 fa 18 con resto di 7
- 18 diviso 8 fa 2 con resto di 2
- 2 diviso 8 fa 0 con resto di 2

$(1209)_{10} = (2271)_{8}$

Soluzione Esercizio 3

In questo caso si applica il metodo “da altra base a base 10”

$4\times6^3 + 2\times6^2 + 1\times6^1 + 5\times6^0 = 864 + 72 + 6 + 5 = 947$

$(4215)_6 = (947)_{10}$

Soluzione Esercizio 4

- 92341 diviso 5 fa 18468 con resto di 1
- 18468 diviso 5 fa 3693 con resto di 3
- 3693 diviso 5 fa 738 con resto di 3
- 738 diviso 5 fa 147 con resto di 3
- 147 diviso 5 fa 29 con resto di 2
- 29 diviso 5 fa 5 con resto di 4
- 5 diviso 5 fa 1 con resto di 0
- 1 diviso 5 fa 0 con resto di 1

$(92341)_{10} = (10423331)_5$

Dimostrazione del metodo delle divisioni successive

Il valore di un numero *n* in una base è dato dal polinomio visto in precedenza:

$$n = c_0 + c_1 \times b + \ldots + c_{k-1} \times b^{k-1} + c_k \times b^k$$

$$= c_0 + b \times (c_1 + \ldots + c_{k-1} \times b^{k-2} + c_k \times b^{k-1}) \quad \text{con } c_i \text{ intero e tale che } 0 <= c_i < b$$

Consideriamo la divisione di *n* per *b (R indica il resto e Q il quoziente)*

$$n = R + b \times Q \qquad (0 <= R < b)$$

$$= c_0 + b \times (c_1 + c_2 \times b + \ldots + c_{k-1} \times b^{k-2} + c_k \times b^{k-1})$$

$\Rightarrow$ $R = c_0$ ovvero, il resto della divisione di *n* per *b* dà c_0 (cifra meno significativa)
$Q = c_1 + b \times (\cdots)$

A partire dal quoziente Q si può iterare il procedimento per ottenere le cifre successive (fino a che Q diventa 0).

Conversione con basi entrambe diverse da dieci

Per convertire un numero da base b1 a base b2 entrambe diverse da dieci possiamo prima convertire da base b1 a base dieci e poi da base dieci a base b2

- base b1 → base dieci
- base dieci → base b2

Esempio: conversione da base cinque a base otto
$(3421)_5 = (??)_8$

base cinque → base dieci:

$(3421)_5 = 3 \times 5^3 + 4 \times 5^2 + 2 \times 5 + 1 = (486)_{10}$

base dieci → base 8 (divisioni successive)

486/8 = 60 con resto di 6, poi 60/8 = 7 con resto di 4, poi 7/8 = 0 con resto di 7 → $(746)_8$

Alcune conversioni semplici

Da base due a base otto

tre cifre in binario (3 bit) = una cifra ottale

questo perché 8 = 2×2×2

Quindi, per convertire da binario a ottale si raggruppano le cifre binarie in gruppi da tre e si converte ogni terzina in ottale.

Esempio, con sei cifre binarie:

$(c_5c_4c_3c_2c_1c_0)_2 = c_5\times2\times2\times2\times2\times2 + c_4\times2\times2\times2\times2 + c_3\times2\times2\times2 + c_2\times2\times2 + c_1\times2 + c0$

$= (c_5\times2\times2 + c_4\times2 + c_3) \times 8 + (c_2\times2\times2 + c_1\times2 + c0)$

$= d_1 \times 8 + d_0$

la prima cifra ottale d_1 è $c_5\times2\times2 + c_4\times2 + c3$

la seconda cifra ottale d_0 è $c_2\times2\times2 + c_1\times2 + c0$

Consideriamo ora $(11010111001)_2$ = 011 010 111 001 = $(3271)_8$

--- --- --- ---

3 2 7 1

Nota: è servito uno zero davanti per ottenere una terzina

Alcune conversioni semplici

Da base otto a base due

ogni cifra ottale sono tre bit. Si converte quindi una cifra alla volta e si concatena il risultato finale:

Esempio:

$(52172)_8$ = 5 2 1 7 2

= 101 010 001 111 010

= 101010001111010

conversioni facili, in generale

se la base b1 è potenza n-esima della base b2

allora n cifre nella base b2 sono una cifra nella base b2

Da binario a esadecimale e viceversa

- esadecimale = base sedici
- $16 = 2^4$
- 4 cifre in binario sono una cifra in esadecimale

Esempio:

Da binario a esadecimale

$$(10101111001)_2 = \underset{5}{\underline{0101}}\ \underset{7}{\underline{0111}}\ \underset{9}{\underline{1001}} = (579)_{16}$$

Da esadecimale a binario

$(791)_{16}$ = 7 9 1

= 0111 1001 0001

= 011110010001

Rappresentazioni esadecimali

la successione dei primi numeri espressi in esadecimale:

0	zero	F	quindici
1	uno	10	sedici
2	due	11	diciassette
3	tre	12	diciotto
4	quattro	13	diciannove
5	cinque	14	venti
6	sei	15	ventuno
7	sette	16	ventidue
8	otto	17	ventitre
9	nove	18	ventiquattro
A	dieci	19	venticinque
B	undici	1A	ventisei
C	dodici	1B	ventisette
D	tredici	….	
E	quattordici		

Esadecimale: conversioni

Da esadecimale a decimale:

$(A2F)_{16} = 10\times16\times16 + 2\times16 + 15 = (2607)_{10}$

Da decimale a esadecimale:

- 2607/16 = 162 con il resto di 15 → l'ultima cifra è F
- 162/16 = 10 con il resto di 2 → la penultima cifra è 2
- 10/16 = 0 con il resto di 10 → la prima cifra è A

Da cui $(2607)_{10}=(A2F)_{16}$

Come visto in precedenza, la conversione da binario a esadecimale è più facile visto che $16 = 2^4$.

Basta quindi prendere le cifre binarie in gruppi da quattro e convertire ogni singolo gruppo.

Nella direzione opposta, una cifra esadecimale corrisponde a un gruppo di quattro cifre binarie.

Generalizzazioni

Come si è visto nel caso esadecimale, è possibile usare cifre che non esistono nell'uso comune, come A, B, ecc. Più in generale, le cifre in un sistema posizionale sono simboli, tanti quanto è la base.

La definizione generale di sistema posizionale è quindi la seguente: un numero b e un insieme di b simboli dette cifre, ognuna che rappresenta un numero diverso da 0 a b-1.

La funzione *val(cifra)* dà il valore di ogni cifra.

Un numero rappresentato in base b è una sequenza di simboli $c_k \ldots c_0$ tale che l'espressione che segue ha valore pari al numero:

$$val(c_k) \times b^k + \ldots + val(c_0) \times b^0$$

Altro esempio di sistema posizionale

- b=12
- cifre: ¢ £ ¤ ¥ § © µ ¶ þ ø ℑ ♠
- val(¢) = 0
  val(£) = 1
  val(¤) = 2
  val(¥) = 3
  val(§) = 4
  val(©)= 5
  val(µ) = 6
  val(¶) = 7
  val(þ) = 8
  val(ø) = 9
  val(ℑ) = 10
  val(♠) =11

Altro esempio di sistema posizionale

- Il numero ♠µ¥ in decimale si ottiene per sviluppo del polinomio:
  val(♠)×12×12 + val(µ)×12 + val(¥) = 11×12×12 + 6×12 + 3 = 1659

- il numero decimale 1001 si converte così:
  - 1001/12 = 83 con resto di 5
  - dato che val(©) = 5, l'ultima cifra è ©
  - 83/12 = 6 con resto di 11, ecc.

  Il risultato della conversione è µ♠©

Numeri e numerali

- In termini formali, si distingue fra numero (un valore) e numerale (la sua rappresentazione).
- Per esempio, 12, XII e dodici sono tre modi diversi per rappresentare lo stesso numero; sono quindi tre numerali.
- Fino a questo momento, si è usato impropriamente il termine numero per indicare sia i numeri che i numerali, dato che la distinzione risultava chiara dal fatto che "numero in base b" non poteva che indicare un numerale, ossia un modo di rappresentare un numero.

Addizioni

Solito modo:

```
   2  7  5  +
1  7  5  4  =
---------
         ↓
         5+4=9
         9
      ↓
      7+5=12 (ossia 2 con riporto 1)
   1  2
   ↓
   2+7+1=10 (ossia 0 con riporto 1)
1  0
↓
1+1=2
2
----------
2  0  2  9
```

Addizioni con base generica

- numeri l'uno sopra l'altro
- si somma cifra per cifra a partire da destra, con la somma fatta nella base
- si mettono i riporti nella somma successiva

Esempio: Somma in ottale

```
   5  2  3  +
1  3  6  1  =
---------
         ↓
         3+1=4
         4
      ↓
      2+6=otto, in ottale 10
      ossia 0 con riporto di 1
   1  0
   ↓
   5+3+1=nove, in ottale 11
   ossia 1 con riporto di 1
1  1
↓
1+1=2
2
---------
2  1  0  4
```

Addizioni in binario

```
  1 0 0 1 1 +
1 0 0 1 1 1 =
-------------
            ↓
            1+1=due, in binario 10
            ossia 0 con riporto 1
          1 0
          ↓
          1+1+1=tre, in binario 11
          ossia 1 con riporto 1
        1 1
        ↓
        0+1+1=due, in binario 10
        ossia 0 con riporto 1
      1 0
      ↓
      0+0+1=1
      1
    ↓
    1+0=1
    1
↓
1
-------------
1 1 1 0 1 0
```

Sommare in esadecimale

$(1F2)_{16} + (BA4)_{16}$:

```
   1 F 2 +
   B A 4 =
----------
       ↓
       2+4=6
       6
     ↓
     F+A=quindici+dieci=
     =venticinque=sedici+nove=
     19 in esadecimale (9 con riporto di 1)
   1 9
   ↓
   1+B+1=uno+undici+uno=tredici= D in esadecimale
   D
----------
   D 9 6
```

Esercizio

$(2201)_3 + (202)_3 = ?$

```
   2 2 0 1 +
     2 0 2 =
-----------
          ↓
          1+2=tre=tre+0
          in base tre: 10
          cioè 0 con riporto di 1
        1 0
        ↓
        0+0+1=1
        1
      ↓
      2+2=quattro=tre+uno
      in base tre 11
      cioè 1 con riporto di 1
    1 1
    ↓
    2+0+1=tre=tre+zero
    in base tre 10
    cioè 0 con riporto di 1
1 0
↓
0+0+1=1
1
-----------
1 0 1 1 0
```

Esercizio

$(11011001)_2 + (11110000111)_2 = ?$

```
          1 1 0 1 1 0 0 1 +
  1 1 1 1 0 0 0 0 1 1 1 =
-------------------------
                        ↓
                          1+1=0 con riporto 1
                      ↓
                        0+1+1=0 con riporto 1
                    ↓
                      0+1+1=0 con riporto 1
                  ↓
                    1+0+1=0 con riporto 1
                ↓
                  1+0+1=0 con riporto 1
              ↓
                0+0+1=1
            ↓
              1+0=1
          ↓
            1+1=0 con riporto 1
        ↓
          0+1+1=0 con riporto 1
      ↓
        0+1+1=0 con riporto 1
    ↓
    0+1+1=0 con riporto 1
↓
0+0+1=1
-------------------------
1 0 0 0 0 1 1 0 0 0 0 0
```

Dimostrazione del metodo di somma

1. due cifre nella stessa posizione sono moltiplicate per la stessa potenza della base
   - Se i due numeri da sommare sono $a_k \ldots a_0$ e $c_k \ldots c_0$, allora il risultato è:

     $(a_k \times b^k + \ldots + a_0 \times b^0) + (c_k \times b^k + \ldots + c_0 \times b^0) =$

     $(a_k + c_k) \times b^k + \ldots + (a_0 + c_0) \times b^0$

   Posso sommare le singole cifre nella stessa posizione

2. la somma di due cifre più uno dà al massimo riporto uno

Dimostrazione del metodo di somma: primo riporto

a_0 e c_0 possono andare da 0 a b-1

la somma può superare b-1
(massimo valore per una singola cifra)

il massimo è (b-1)+(b-1)=b+(b-2)

in base b è 1x dove x è la cifra che rappresenta b-2

l'1 è il riporto

Abbiamo dimostrato che*: il massimo valore della somma di due cifre in base b è 1x, dove x è la cifra b-2*

Dimostrazione del metodo di somma: riporti successivi

si sommano due cifre più un eventuale riporto 1

il massimo è (b-1)+(b-1)+1=2b-1=b+(b-1)

in base b è 1y dove y è la cifra che rappresenta b-1

in qualsiasi base, il riporto può valere al massimo uno!

Abbiamo dimostrato che*: il massimo valore della somma di due cifre più uno in base b è 1y, dove y è la cifra b-1*

Insieme a quanto dimostrato prima possiamo concludere che il riporto può valere al massimo uno e che non può mai essere fatto di due o più cifre.

Dimostrazione per induzione

Si noti il meccanismo con cui si è dimostrato questo fatto:

- per le ultime due cifre, il riporto al massimo vale uno;
- per una posizione generica: assumendo che il riporto precedente sia al massimo uno, si dimostra che il riporto generato può valere al massimo uno.

Una dimostrazione di questo tipo, in cui si assume vero un fatto per poi dimostrarlo vero in una condizione successiva si dice *dimostrazione per induzione*.

Overflow

Nella maggior parte dei casi pratici, ogni numero si rappresenta con una quantità fissata di bit

esempio: 64 bit

il risultato di una somma può avere un bit in più

addendi a 64 bit, risultato a 65 (non rappresentabile)

Esempio: addendi a otto bit:

```
  11010001 +
  01100001 =
----------
 100110010
```

risultato a nove bit. Non rappresentabile con otto bit

Overflow: condizione in cui:

- esistono dei valori rappresentabili nel numero prefissato di bit
- si effettua un'operazione che produce un risultato non memorizzabile con quei bit

Sottrazioni

- stesso metodo della base dieci applicato alla sottrazione in qualsiasi base
- sottrazione per cifre con eventuale riporto di -1

Il sistema sfrutta ancora due proprietà del sistema posizionale:

1. le due cifre nella stessa posizione sono moltiplicate per la stessa potenza della base, per cui si può fare cifra-cifra;
2. le cifre vanno da 0 a b-1, per cui cifra-cifra-1 vale al minimo 0-(b-1)-1=-b+0, che produce appunto il riporto massimo di meno uno.

La dimostrazione (che il riporto massimo è -1) si può fare per induzione ed è simile a quella vista per la somma.

Sottrazioni in base dieci

```
  3 2 1 -
    6 2 =
---------
        ↓
        1-2=-1, ossia 9-10
        cifra 9 e riporto -1
   -1 9
    ↓
    2-6-1=-5, ossia 5-10
    cifra 5 e riporto -1
 -1 5
  ↓
  3-1=2
  2
---------
  2 5 9
```

9-10 = risultato è 9, riporto -1 (-10 all'ultima cifra = -1 alla penultima, ecc.)

Sottrazioni in base otto

Questo che si è detto per il dieci vale per qualsiasi base.
Ad esempio in ottale.

Il numero negativo -5 è stato riscritto come 3-8, che equivale a (-1)×8+3. È nella stessa forma di 13=1×8+3 in ottale, ma è come se la prima cifra fosse negativa: (-1)3.
Per questo il riporto è -1.

Detto in un altro modo: nel polinomio che dice il valore di un numero, l'ultima cifra è moltiplicata per uno; la penultima per otto.
Questo vuol dire che un -8 all'ultima cifra è uguale a -1 alla penultima (che è moltiplicata per otto).

```
  2 7 1 -
  1 3 6 =
---------
        ↓
        1-6=-5=3-8
        cifra 3 e riporto -1
   -1 3
    ↓
    7-3-1=3
    3
  ↓
  2-1=1
  1
---------
  1 3 3
```

Sottrazioni in base due

```
 1  1  0  0  1  0  0  -
 1  0  0  1  0  1  1  =
----------------------
                  ↓
                  0-1=-1=1-2, ossia 1 con riporto di -1
               ↓
               0-1-1=-2=0-2, ossia 0 con riporto di -1
            ↓
            1-0-1=0
         ↓
         0-1=-1=1-2, ossia 1 con riporto di -1
      ↓
      0-0-1=-1=1-2, ossia 1 con riporto di -1
   ↓
   1-0-1=0
 ↓
 1-1=0
----------------------
 0  0  1  1  0  0  1
```

Underflow

- Per quello che riguarda l'overflow, sottraendo due numeri positivi il massimo risultato possibile è il primo numero. Dato che il primo numero era rappresentabile, lo sarà anche il risultato: l'overflow è impossibile. D'altra parte, il risultato può valere meno di zero, e non è quindi rappresentabile con i sistemi di rappresentazione dei numeri positivi.

- Una situazione in cui il risultato di un'operazione non è rappresentabile perchè troppo basso si dice underflow.

Moltiplicazione

La moltiplicazione in base qualsiasi si calcola come in decimale:
il moltiplicando per ogni cifra del moltiplicatore.

Per esempio, in **ottale**:

```
   412 ×
    52 =
------
  1024 ← 412 × 2, tenendo conto che 4×2=10
 2462  ← 412 × 5, tenendo conto che 5×2=12 e 5×4=24
------
 25644 ← somma
```

Moltiplicazione

Il metodo funziona grazie al sistema posizionale: indichiamo con a il moltiplicando e con $c_k…c_0$ il moltiplicatore:

$$a \times c_k…c_0 = a \times (c_k \times b^k + … + c_0 \times b^0)$$
$$= a \times c_k \times b^k + … + a \times c_0 \times b^0$$

Si moltiplica a per ogni cifra c_i e si ottiene un numero $d_h…d_0$

Questo numero rappresenta $d_h \times b^h + …. + d_0 \times b^0$

Va moltiplicato per b^i, che lo fa diventare $d_h \times b^{h+i} + …. + d_0 \times b^i$

Questo equivale a spostare le cifre $d_h…d_0$ di i posizioni verso sinistra.

Moltiplicazione in binario

Ogni cifra può essere solo 0 o 1

Primo numero × cifra-del-secondo =

- il primo, se la cifra è 1
- 0, se la cifra è 0

I singoli prodotti o sono il primo operando oppure valgono 0

```
    10111 ×
     1101 =
---------
    10111 ← 10111×1
   00000  ← 10111×0
  10111   ← 10111×1
 10111    ← 10111×1
---------
100101011
```

Ogni prodotto è zero oppure il primo operando spostato a sinistra (shift o spostamento: traslare un numero; shifter: circuiti che realizzano uno shift)

Moltiplicazione per una potenza della base

E’ uguale a uno shift

Esempio: moltiplicazione per due (10) in binario:

```
 110001 ×
     10 =
--------
 000000 ← 110010 × 0
110001  ← 110010 × 1
--------
1100010
```

stesso numero spostato a sinistra di una posizione

Moltiplicazione per una potenza della base

Esempio: moltiplicazione per otto (1000) in binario:

```
    110001 ×
      1000 =
----------
    000000 ← 110010 × 0
   000000  ← 110010 × 0
  000000   ← 110010 × 0
110001     ← 110010 × 1
----------
110001000
```

stesso numero spostato a sinistra di tre posizioni

Vale per tutte le basi: *n-esima potenza della base = 10...0 con n zeri*

l'unico prodotto intermedio è quello generato da 1

è il numero di partenza spostato di n posizioni

Moltiplicazione per potenze di due

In Python esiste un'operatore << :

`a << b` è a (o meglio la sua rappresentazione binaria) spostato a sinistra di b posizioni

è come aggiungere b zeri in fondo

equivale a moltiplicare a per 2^b

```
print(1<<4)
print(5<<2)
```

stampano rispettivamente 16 e 20:

print(1<<4)

1 con quattro zeri ⇒ 10000 = 16

print(5<<2)

5 = 101 con due zeri ⇒ 10100 = 16+4 = 20

Divisione per potenze di due

divisioni per potenze di due

→

spostamento a destra dei bit

in Python: a>>b è a spostato a destra di b posizioni

a >> b è a (o meglio la sua rappresentazione binaria) spostato a destra di b posizioni

è come eliminare le ultime b cifre

equivale a dividere a per 2^b

i bit che "escono" si perdono

Massimo numero rappresentabile

in hardware, i numeri si rappresentano con un numero prefissato di bit

esempio: 64

massimo numero senza segno rappresentabile: sessantaquattro uni

1111111111111111111111111111111111111111111111111111111111111111 →

$1\times2^{63} + 1\times2^{62} +...+ 1\times2^{1} + 1\times2^{0} =$

9223372036854775808 + 4611686018427387904 + ... + 2 + 1 = 18446744073709551615

massimo numero con n bit

$$1\times2^{n-1} + 1\times2^{n-2} + \ldots + 1\times2^{1} + 1\times2^{0} = 2^{n}-1$$

- il successivo di 11…11 è 100…00
- è il primo numero non rappresentabile: 2^n
- massimo rappresentabile: questo meno uno

64 bit bastano?

massimo rappresentabile: 18446744073709551615

esistono applicazioni che usano valori più grandi

si possono usare due numeri a 64 bit A e B per rappresentarne uno a 128

la coppia $\langle A,B \rangle$ rappresenta $A \times 2^{64} + B$

Esercizi Riepilogativi

Esercizio

Convertire i due numeri decimali 591 e 211 in binario. Effettuare poi la somma in binario. Assumendo sedici bit, controllare se si è verificato l'overflow. In caso contrario, riconvertire il risultato in decimale, verificando se il risultato è effettivamente 802.

Soluzione

Conversione in binario di 591

- 591 diviso 2 fa 295 con resto di 1
- 295 diviso 2 fa 147 con resto di 1
- 147 diviso 2 fa 73 con resto di 1
- 73 diviso 2 fa 36 con resto di 1
- 36 diviso 2 fa 18 con resto di 0
- 18 diviso 2 fa 9 con resto di 0
- 9 diviso 2 fa 4 con resto di 1
- 4 diviso 2 fa 2 con resto di 0
- 2 diviso 2 fa 1 con resto di 0
- 1 diviso 2 fa 0 con resto di 1

resti in ordine inverso: 1001001111

Soluzione (cont.)

Conversione di 211 in binario

- 211 diviso 2 fa 105 con resto di 1
- 105 diviso 2 fa 52 con resto di 1
- 52 diviso 2 fa 26 con resto di 0
- 26 diviso 2 fa 13 con resto di 0
- 13 diviso 2 fa 6 con resto di 1
- 6 diviso 2 fa 3 con resto di 0
- 3 diviso 2 fa 1 con resto di 1
- 1 diviso 2 fa 0 con resto di 1

resti in ordine inverso: 11010011

Soluzione (cont.)

somma dei due numeri

```
 11 11111      ← riporti
1001001111 +
  11010011 =
------------
1100100010
```

rientra in sedici bit → no overflow

Soluzione (cont.)

Conversione di 1100100010 in binario

$1\times2^9 + 1\times2^8 + 0\times2^7 + 0\times2^6 + 1\times2^5 + 0\times2^4 + 0\times2^3 + 0\times2^2 + 1\times2^1 + 0\times2^0 = 512 + 256 + 0 + 0 + 32 + 0 + 0 + 0 + 2 + 0 = 802$

Esercizio

Convertire il numero decimale 52 e quello esadecimale F1 in binario. Moltipicare i due numeri binari così ottenuti e convertire il risultato in ottale.

Soluzione

conversione di 52 in binario

- 52 diviso 2 fa 26 con resto di 0
- 26 diviso 2 fa 13 con resto di 0
- 13 diviso 2 fa 6 con resto di 1
- 6 diviso 2 fa 3 con resto di 0
- 3 diviso 2 fa 1 con resto di 1
- 1 diviso 2 fa 0 con resto di 1

resti dall'ultimo al primo: 110100

Soluzione (cont.)

Conversione di F1 in binario

si potrebbe convertire prima in decimale e poi in binario

ma: $16 = 2^4$

una cifra esadecimale = quattro bit

```
F1 =  F    1    = 11110001
     ---- ----
     1111 0001
```

Soluzione (cont.)

moltiplicazione

il primo numero ha bit meno che valgono uno del secondo

conviene fare secondo per primo (meno addendi)

```
       11110001 ×
         110100 =
-----------------
    11110001
  11110001
 11110001
-----------------
1100011110100
```

Soluzione (cont.)

conversione in ottale

conversione a dieci e poi a otto

oppure: tre bit = una cifra ottale

```
1100001111010O0 = 011 000 011 110 100
                  --- --- --- --- ---
                   3   0   3   6   4

                = 30364
```