Rappresentazione dell'informazione

Informazione analogica e digitale

L'esigenza di comunicare, memorizzare ed elaborare informazioni ha determinato lo sviluppo di tecniche di rappresentazione dell'informazione sia **analogiche** che **digitali**. La differenza tra le prime e le seconde è ben esemplificata dagli orologi. Per indicare il numero che è la misura del tempo corrente, gli orologi analogici usano delle grandezze variabili con continuità (gli angoli formati dalle lancette con un riferimento fisso) mentre gli orologi digitali usano una sequenza finita di simboli appartenenti ad un insieme finito (spesso le dieci cifre decimali ed i due punti).

In sostanza, mentre la rappresentazione analogica di un'informazione si presenta sotto forma di un segnale che varia con continuità (*segnale analogico*), la rappresentazione digitale ha la forma di un segnale che assume un numero molto limitato di valori significativi diversi (*segnale digitale* o *discreto*).

Le tecniche analogiche di rappresentazione dell'informazione stanno cedendo sempre più il passo a quelle digitali, in quanto queste ultime offrono significativi vantaggi in termini di immunità dagli errori. Per comprenderlo è sufficiente pensare alla sicurezza con cui in un segnale digitale anche notevolmente distorto possono essere riconosciuti e quindi ripristinati i pochi valori significativi, ognuno considerevolmente diverso da quelli contigui.

La **rappresentazione binaria**, che fa uso di due soli valori distinti, è di gran lunga la forma di rappresentazione digitale più utilizzata. Ciò grazie sia alla semplicità realizzativa dei circuiti che operano con due soli valori che al massimo livello di affidabilità che i segnali binari offrono.

Prescindendo completamente dalla natura fisica dei segnali (due diversi livelli di tensione elettrica tra due morsetti o la presenza o l'assenza di corrente in un conduttore elettrico), nella rappresentazione binaria si suole, per convenzione, indicare con i simboli ***0*** e ***1*** i due valori distinti utilizzati. A questi simboli si dà il nome di **bit** (da ***bi***nary digi***t***); si parla quindi di *bit 0* e di *bit 1*.

Sistemi d'elaborazione

Il termine **sistema d'elaborazione** indica qualsiasi apparato di elaborazione automatica dell'informazione che faccia uso della rappresentazione binaria dell'informazione.

Un sistema d'elaborazione che debba scambiare informazione in forma analogica con l'esterno può essere interfacciato ad opportuni *convertitori* che trasformano i segnali analogici in digitali e viceversa. La tecnologia di registrazione digitale dei segnali audio utilizzata nei *compact disk* si fonda proprio su queste conversioni. La **conversione analogico-digitale** viene effettuata campionando la forma d'onda a intervalli regolari. L'informazione si trasforma così in una sequenza di numeri reali che sono resi disponibili all'uscita del convertitore opportunamente rappresentati in binario. Se il campionamento avviene con una frequenza sufficientemente alta, è possibile con un procedimento di **conversione digitale-analogico**, ricostruire la forma d'onda originaria (*teorema di Nyquist*). L'**errore di quantizzazione**, intrinseco nel passaggio dal continuo al discreto e dovuto alla necessità di rappresentare ciascun numero reale con una stringa di bit di lunghezza finita, può essere reso piccolo quanto si vuole aumentando il numero di bit della rappresentazione.

Rappresentazione binaria dell'informazione

Qualunque sia l'informazione che si vuole assoggettare ad elaborazione automatica, è necessario rappresentarla in forma binaria, ossia con una sequenza di bit di lunghezza finita. La finitezza di ciascuna rappresentazione deriva da ovvi motivi di fisica realizzabilità degli apparati.

Per consentire l'elaborazione automatica dell'informazione numerica il problema da risolvere è quello della rappresentazione binaria dei numeri. Altre volte l'informazione ha forma testuale, ossia è una stringa di simboli appartenenti a qualche insieme finito (un testo scritto, numerali, ecc.), ed il problema è quindi quello della codifica di ciascun simbolo dell'alfabeto sorgente con una opportuna sequenza di bit. Infine, spesso l'informazione da rappresentare è strutturata, ossia consta di singoli elementi d'informazione di tipo numerico o testuale organizzati in strutture (esempi sono una matrice di numeri, un elenco di nomi, un albero genealogico, ecc.); questa volta, oltre a codificare i singoli elementi d'informazione occorre trovare la maniera di rappresentare la struttura.

Rappresentazione dei numeri naturali nei sistemi di numerazione posizionale

Il concetto di numero naturale è un'astrazione che sorge dalla comune esperienza che gli oggetti possono essere raggruppati in insiemi. L'uomo ha da sempre avuto l'esigenza di comunicare, memorizzare e calcolare numeri e, per questo motivo, ha inventato i **sistemi di numerazione**, forme di rappresentazione dei numeri che impiegano sequenze di *cifre* (simboli appartenenti ad un alfabeto finito).

La rappresentazione di un numero naturale (*numerale*) assume forme diverse a seconda del sistema di numerazione usato. I primi sistemi di numerazione erano chiaramente pittorici. Ad esempio, nel *sistema di numerazione romano* i primi numeri sono rappresentati da I, II e III. Il sistema romano usa le cifre I, V, X, L, C, D, M che rappresentano rispettivamente i valori crescenti uno, cinque, dieci, cinquanta, cento, cinquecento, mille e si basa su di un *principio di additività*: ciascuna cifra dà un contributo pari al suo valore, preso positivamente se rispetta l'ordinamento decrescente, negativamente altrimenti. Così, MCDXXXIV rappresenta il numero millequattrocentotrentaquattro. Il sistema di numerazione romano non è adatto per rappresentare numeri molto grandi né consente l'agevole effettuazione di calcoli.

Il *sistema di numerazione arabo*, diffusosi nel Medio Evo e usato ancora oggi, è basato su di un *principio posizionale*: la stessa cifra assume valori diversi a seconda della posizione assunta nella rappresentazione. Precisamente, i **sistemi di numerazione posizionale in base *b*** sono fondati sul seguente:

> **Teorema**: Fissata una base b (naturale maggiore di uno), per ogni naturale n, esiste una ed una sola k-pla (con $k = 1$ per n nullo e $k = \lfloor \log_b n + 1 \rfloor$ per n positivo) di numeri naturali minori di b
>
> $$c_{k-1},\ c_{k-2},\ \ldots,\ c_1,\ c_0 \qquad (c_{k-1} > 0)$$
>
> tali che
>
> $$(1)\quad n = c_{k-1} \times b^{k-1} + c_{k-2} \times b^{k-2} + \ldots + c_1 \times b + c_0$$

Pertanto, si può assumere come rappresentazione in base b del numero n la stringa:

$$(2)\quad n_b = c_{k-1}\ c_{k-2}\ \ldots\ c_1\ c_0$$

La rappresentazione (2) si dice **posizionale** perché ciascuna cifra rappresenta una quantità diversa a seconda della posizione che occupa nel corpo della stringa; questa quantità è il prodotto del valore numerico della cifra per una potenza intera della base. Quanto più la cifra è a sinistra nella stringa tanto più essa è significativa, perché maggiore è la potenza per cui va moltiplicata.

La (2) si dice rappresentazione posizionale *binaria* se $b = 2$, *ternaria* se $b = 3$, *quaternaria* se $b = 4$, *ottale* se $b = 8$, *decimale* se $b = 10$, *esadecimale* se $b = 16$, ecc.

Scelte arbitrariamente b cifre per i primi b numeri naturali (da *zero* a $b-1$), la (2) risolve il problema di rappresentare con queste cifre in modo univoco un qualsiasi naturale. Per convenzione, i sistemi di numerazione con base non maggiore di 16 usano come cifre i primi b simboli della sequenza

0 1 2 3 4 5 6 7 8 9 A B C D E F

Per evitare confusione, nel caso si adoperino sistemi di numerazione con basi differenti, è d'uso specificare con un pedice la base b (espressa in decimale); ad esempio:

$$321_4 = 57_{10}$$

$$A3B_{12} = 1487_{10}$$

Nei sistemi di numerazione posizionali valgono le seguenti proprietà:

a) Con p posizioni in base b sono rappresentabili b^p numeri naturali distinti, tutti i valori dell'intervallo chiuso $[0, b^p-1]$. La rappresentazione del massimo numero b^p-1 è la sequenza in cui la cifra $b-1$ è ripetuta p volte.

b) La potenza *n-ma* della base è rappresentata dalla cifra *1* seguita da n cifre *0*. In particolare, la base di un sistema di numerazione posizionale viene rappresentata nel sistema stesso sempre dalla sequenza di cifre *10*.

c) La rappresentazione di $n \times b$ si trova aggiungendo alla rappresentazione di n uno *0* nella posizione meno significativa. Pertanto, la rappresentazione di $n \times b$ ha bisogno di una cifra in più di quelle necessarie per n.

d) La rappresentazione di n **div** b, con **div** operatore di divisione intera si ottiene scartando la cifra meno significativa della rappresentazione di n. Pertanto, la rappresentazione di n **div** b ha bisogno di una cifra in meno di quelle necessarie per n.

L'aritmetica nei sistemi posizionali

I sistemi di numerazione posizionali si sono affermati perché consentono di effettuare agevolmente le quattro operazioni aritmetiche fondamentali. I metodi per eseguire operazioni aritmetiche su numeri rappresentati in una qualsiasi base b sono del tutto uguali a quelli dell'aritmetica decimale che usiamo quotidianamente. Ovviamente, occorre fare riferimento alle tabelle di somma e prodotto relative alla base d'interesse che forniscono la somma ed il prodotto a due a due dei primi b naturali. Per il sistema binario le tabelle sono:

+	0	1
0	0	1
1	1	10

x	0	1
0	0	0
1	0	1

Esempi

$01101_2 + 01011_2 = 11000_2$

```
riporto 011110
         01101 +
         01011 =
        --------
somma    11000
```

$01101_2 + 11111_2 = (1)01100_2$

```
riporto 111110
         01101 +
         11111 =
        --------
somma   101100
```

Il riporto nullo dalla posizione più significativa indica che non vi è stato *supero di capacità* (*trabocco*, *overflow*) della rappresentazione. La verifica di questa condizione è sempre necessaria nei sistemi d'elaborazione, in quanto il numero delle posizioni a disposizione di ciascuna rappresentazione (e quindi anche del risultato) è fissato a priori. Viceversa, il riporto unitario dalla posizione più significativa indica che vi è stato supero di capacità. In questo caso, la quantità che rimane nelle p posizioni disponibili del risultato è in difetto di b^p rispetto al risultato vero. Più in generale, quando il risultato di un'operazione non è rappresentabile nelle p posizioni a disposizione, la quantità che rimane nelle p posizioni è:

$$r \textbf{ mod } b^p$$

ove r è il risultato vero e **mod** è l'operatore resto della divisione intera.

Esempio

$1101_2 \times 1011_2 = 11000_2$

```
      1101 ×
      1011 =
------------
      1101
     1101=
    1101==
------------
  10001111
```

Il procedimento di moltiplicazione esemplificato si giustifica osservando che, per la (1):

$$a \times d = c_{k-1} \times a \times b^{k-1} + c_{k-2} \times a \times b^{k-2} + \ldots + c_1 \times a \times b + a \times c_0$$

dove $c_{k-1} c_{k-2} \ldots c_1 c_0$ è la rappresentazione in base b di d.

Osserviamo che per esprimere il prodotto di due numeri di p cifre occorrono, in generale $2p$ cifre.

Conversioni di base dei numeri naturali

Fissata una base b, si pone il problema di determinare per ogni numero naturale n la sua rappresentazione in base b. La formula (1) evidenzia che le cifre c_0, c_1, ..., c_{k-1} sono ottenibili con ripetute divisioni intere:

$$c_0 = n \bmod b$$
$$c_1 = (n \operatorname{div} b) \bmod b$$
$$c_2 = ((n \operatorname{div} b) \operatorname{div} b) \bmod b$$
$$\ldots$$

Il resto della divisione intera tra n e la base b fornisce la cifra meno significativa c_0. Le altre cifre c_1, c_2, ... , c_{k-1} via via più significative si ottengono ripetendo ad ogni passo la divisione tra il quoziente ottenuto al passo precedente e la base b. Il procedimento termina quando si ottiene un quoziente nullo.

Per eseguire concretamente il metodo delle divisioni successive occorre che n e b siano rappresentati nella base B la cui aritmetica si intende usare. Pertanto, il metodo delle divisioni successive effettua in realtà la conversione dalla base B alla base b, operando secondo l'aritmetica della base di partenza B. Esso, se eseguito manualmente, è adatto al caso $B = dieci$.

Esempi

$10268_{10} = 24034_8$

	8
10268	4
1283	3
160	0
20	4
2	2
0	

$524_{10} = 20C_{16}$

	16
524	12
32	0
2	2
0	

Per usare l'aritmetica della base di arrivo, si può adoperare direttamente la (1). Per evitare le elevazioni a potenza è possibile operare con ripetute moltiplicazioni:

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

Il metodo delle moltiplicazioni ripetute effettua quindi la conversione dalla base B alla base b, operando secondo l'aritmetica della base di arrivo b. Esso, se eseguito manualmente, è adatto al caso $b = dieci$.

Esempi

$24034_8 = 10268_{10}$

$(((2 \times 8 + 4) \times 8 + 0) \times 8 + 3) \times 8 + 4 =$
$(160 \times 8 + 3) \times 8 + 4 =$
$1283 \times 8 + 4 =$
10268

$20C_{16} = 524_{10}$

$(2 \times 16 + 0) \times 16 + 12 =$
$32 \times 16 + 12 =$
524

In generale, per effettuare conversioni tra basi entrambe non decimali, i due metodi possono essere applicati in cascata, usando la rappresentazione decimale come intermedia.

Il processo di conversione di base è particolarmente semplice se le due basi sono l'una una potenza intera dell'altra. Infatti, se $B = b^k$, è sufficiente sostituire a ciascuna cifra della rappresentazione in base B le corrispondenti k cifre della sua rappresentazione in base b. Viceversa, se $b = B^k$, è sufficiente sostituire ad ogni gruppo di k cifre in base B la corrispondente cifra in base b. Le cifre vanno aggregate a partire da quelle meno significative e l'ultimo gruppo di cifre può essere completato, se occorre, con degli *0*. Quando le due basi sono potenze distinte di una terza base, i due metodi di conversione possono essere usati in cascata.

Esempi

$24034_8 = 010\ 100\ 000\ 011\ 100_2$

$121_3 = 01\ 21_3 = 17_9$

$AB_{16} = 10\ 101\ 011_2 = 253_8$

Le rappresentazioni ottale ed esadecimale sono usate molto spesso per rappresentare in forma compatta sequenze di bit. Il metodo illustrato consente infatti di compattare e scompattare agevolmente le sequenze.

Rappresentazione binaria dei numeri naturali

I sistemi di numerazione posizionali risolvono il problema della rappresentazione binaria dei numeri naturali. Infatti, la *rappresentazione binaria diretta* di un naturale si ottiene esprimendolo nel sistema posizionale in base 2 ed aggiungendo degli zeri non significativi in testa se il numero di bit a disposizione p supera quello delle cifre binarie. Poiché le *disposizioni con ripetizione di 2 simboli su p posti* sono 2^p, p bit consentono di rappresentare 2^p numeri naturali diversi, tutti quelli appartenenti all'intervallo chiuso $[0, 2^p-1]$.

La *rappresentazione binaria indiretta* usa invece una base diversa dalla base 2 e codifica ciascuna cifra della base prescelta con una stringa di bit (scegliendo una base potenza intera di 2 e codificandone le cifre con le rappresentazioni binarie dirette dei relativi valori, si ottengono delle rappresentazioni indirette indistinguibili da quella diretta). Se, come di solito accade, si sceglie la base *dieci*, si ottiene una rappresentazione non ottima in quanto occorrono almeno 4 bit per codificare una cifra decimale (con 6 configurazioni non utilizzate). Pertanto, disponendo di $4p$ bit, la rappresentazione indiretta in base dieci consente di rappresentare solo i naturali da *0* a $10^p - 1$, mentre quella diretta arriva fino a $2^{4p}-1$.

La scelta del codice per le cifre decimali è arbitraria. La tabella seguente riporta alcuni dei codici impiegati:

	8-4-2-1	**eccesso 3**	**4-3-1-1**	**Gray (ciclico)**	**8-4-2-1 con parità**
0	0 0 0 0	0 0 1 1	0 0 0 0	0 0 0 0	0 0 0 0 0
1	0 0 0 1	0 1 0 0	0 0 0 1	0 0 0 1	0 0 0 1 1
2	0 0 1 0	0 1 0 1	0 0 1 1	0 0 1 1	0 0 1 0 1
3	0 0 1 1	0 1 1 0	0 1 0 0	0 0 1 0	0 0 1 1 0
4	0 1 0 0	0 1 1 1	1 0 0 0	0 1 1 0	0 1 0 0 1
5	0 1 0 1	1 0 0 0	0 1 1 1	1 1 1 0	0 1 0 1 0
6	0 1 1 0	1 0 0 1	1 0 1 1	1 0 1 0	0 1 1 0 0
7	0 1 1 1	1 0 1 0	1 1 0 0	1 0 0 0	0 1 1 1 1
8	1 0 0 0	1 0 1 1	1 1 1 0	1 1 0 0	1 0 0 0 1
9	1 0 0 1	1 1 0 0	1 1 1 1	0 1 0 0	1 0 0 1 0

- Il codice **8-4-2-1** (dai pesi che i bit hanno nel codice di ciascuna cifra) codifica ciascuna cifra con la rappresentazione binaria diretta a quattro bit del relativo valore. È questo il codice più utilizzato, che dà luogo alla rappresentazione indiretta BCD (*binary coded decimal*).
- Il codice **eccesso 3** si ottiene dal codice 8-4-2-1 sommando tre ad ogni cifra ed è un codice *autocomplementante* o *emisimmetrico*, in cui la codifica del *complemento diminuito* di una cifra *c* (ossia di *9-c*) (vedi §3.2) si ottiene complementando (scambiando) i singoli bit.
- Il codice **4-3-1-1** è un codice pesato autocomplementante.
- Il codice **ciclico** o di **Gray** codifica cifre adiacenti con parole codice che differiscono in un solo bit. Le cifre *0* e *9* sono considerate adiacenti.
- Il codice **8-4-2-1 con parità** aggiunge ai quattro bit del codice 8-4-2-1 un quinto bit ridondante, detto *bit di parità* (o *di disparità*), scelto in modo da rendere sempre pari (o dispari) il numero di bit *1* presenti in ciascuna parola codice. Un errore di molteplicità dispari, che alteri cioè un numero dispari di bit, può così essere rilevato. È utile sapere che esistono codici che impiegando un numero maggiore di bit ridondanti, consentono non solo di rilevare ma anche di correggere gli errori singoli.

La scelta tra la rappresentazione binaria diretta e quella indiretta in base 10 va guidata dalla considerazione che la prima consente di economizzare i bit e dà luogo ad un'aritmetica più semplice e veloce mentre la seconda evita le costose operazioni di conversione di base.

Rappresentazione per segno e modulo dei numeri relativi

L'uomo usa di solito per i numeri relativi la **rappresentazione per segno e modulo**, in cui ogni numero è rappresentato dal valore assoluto (*modulo*) e dal segno, positivo (+) o negativo (-).

Nel caso binario, occorre riservare un bit (quello alla sinistra o bit più significativo) al segno, con la convenzione che + sia codificato con *0* e - con *1*. Pertanto, avendo a disposizione *p* bit e scegliendo la rappresentazione binaria diretta del modulo, sono rappresentabili tutti i numeri relativi appartenenti all'intervallo chiuso e simmetrico rispetto allo zero $[-(2^{p-1}-1), +(2^{p-1}-1)]$.

La rappresentazione per segno e modulo ha due distinte rappresentazioni dello zero (*+0* e *-0*), che in binario sono rispettivamente *00...0* e *10...0*. Questo è fonte di complicazioni, ad esempio, nel confronto di un numero con lo zero.

Inoltre, la rappresentazione per segno e modulo rende alquanto difficoltosa l'effettuazione di addizioni e sottrazioni. Infatti, ad esempio, per sommare due numeri relativi occorre analizzarne i segni, per decidere se sommarne o sottrarne aritmeticamente i moduli; nel secondo caso, i moduli vanno anche confrontati per stabilire quale dei due debba essere il sottraendo e quale il minuendo.

Rappresentazione per complemento alla base dei numeri relativi

La **rappresentazione per complemento alla base** è una rappresentazione *senza segno*, in cui un numero relativo viene rappresentato per mezzo (della rappresentazione) di un numero naturale. Se il numero è non negativo questo naturale è il modulo *m* del numero, altrimenti è il **complemento alla base** del modulo, definito come

$$\underline{m} = b^p - m$$

ove *b* è la base prescelta e *p* sono le cifre a disposizione.

La definizione di complemento alla base si estende a zero ponendo $\underline{0} = 0$.

Pertanto, supposta la base *b* pari, e suddivisi i b^p naturali in parti uguali tra numeri negativi e non negativi, sono rappresentabili per complemento alla base tutti i numeri relativi dell'intervallo chiuso $[-b^p \textbf{ div } 2, +b^p \textbf{ div } 2 - 1]$.

Esempi

Con *b* = `10` e *p* = `2` sono rappresentabili tutti i numeri relativi dell'intervallo `[-50, +49]`:

Numero relativo	`-50`	`-49`	`...`	`-1`	`0`	`+1`	`...`	`+49`
Rappresentazione per complemento alla base	`50`	`51`	`...`	`99`	`00`	`01`	`...`	`49`

Con *b* = `2` e *p* = `4` sono rappresentabili tutti i numeri relativi dell'intervallo `[-8, +7]`:

Numero relativo	`-8`	`-7`	`...`	`-1`	`0`	`+1`	`...`	`+7`
Rappresentazione per complemento alla base	`1000`	`1001`	`...`	`1111`	`0000`	`0001`	`...`	`0111`

Nella rappresentazione per complemento alla base i numeri negativi hanno la cifra più significativa maggiore o uguale a b **div** *2*. In particolare, se $b = 2$, il bit più significativo della rappresentazione per complemento è *1* se il numero è negativo, *0* altrimenti. Per questo, esso è detto *bit segno*.

La rappresentazione per complemento alla base di un numero relativo può interpretarsi come una rappresentazione pesata in cui però si sostituisca la cifra più significativa con il suo complemento alla base cambiato di segno:

$$(3)\quad r = -\underline{c}_{k-1} \times b^{k-1} + c_{k-2} \times b^{k-2} + \ldots + c_1 \times b + c_0$$

Ne caso binario, il complemento alla base di una cifra coincide con la cifra stessa, quindi:

$$(4)\quad r = -c_{k-1} \times b^{k-1} + c_{k-2} \times b^{k-2} + \ldots + c_1 \times b + c_0$$

Esempi

Il numero relativo la cui rappresentazione per complemento alla base dieci è **99** è, per la (3):

$$-\underline{9} \times 10 + 9 = -1 \times 10 + 9 = -10 + 9 = -1$$

Il numero relativo la cui rappresentazione per complemento alla base due è **1111** è, per la (4):

$$-1 \times 8 + 1 \times 4 + 1 \times 2 + 1 = -8 + 4 + 2 + 1 = -1$$

La rappresentazione per complemento a $p + q$ cifre di un numero relativo si può ottenere da quella a p cifre completando quest'ultima sulla sinistra con q cifre tutte uguali a $b - 1$ se il numero è negativo, a *0* altrimenti.

Esempi

La rappresentazione per complemento a due cifre decimali di **-49** è **51**, mentre quelle a tre e quattro cifre sono rispettivamente **951** e **9951**.

La rappresentazione per complemento a quattro cifre binarie di **-8** è **1000**, mentre quelle a cinque e sei cifre sono rispettivamente **11000** e **111000**.

Il calcolo del complemento alla base di m, dato da $\underline{m} = b^p - m$, dove b è la base impiegata e p sono le cifre a disposizione, non richiede in realtà l'effettuazione di una sottrazione. Infatti, introdotto il **complemento diminuito di** m:

$$\underline{m} = (b^p - 1) - m$$

si ha:

$$\underline{m} = \underline{\underline{m}} + 1$$

Il calcolo del complemento diminuito si effettua agevolmente osservando che $b^p - 1$ è rappresentato in base b dalla stringa di p cifre tutte uguali a $b-1$ e che, quindi, la rappresentazione in base b del complemento diminuito $\underline{m}$ si ottiene semplicemente sostituendo a ciascuna cifra c di m il relativo complemento diminuito $(b-1) - c$. In particolare, nel caso binario, basta scambiare ciascun *1* con uno *0* e viceversa.

Esempi

La rappresentazione in base dieci del complemento diminuito di **49** è **(9-4)(9-9)** ossia **50**. Quella del complemento alla base è quindi **50 + 1 = 51**.

La rappresentazione in base due del complemento diminuito di **1011** è **0100**. Quella del complemento alla base è quindi **0100 + 1 = 0101**.

Nel caso binario la rappresentazione del complemento alla base di un numero si può ottenere ancora più agevolmente da quella del numero copiando da destra verso sinistra i bit del numero, fino al primo 1 incluso, indi complementando bit a bit gli altri.

Esempi

La rappresentazione in base due del complemento alla base di **1111** è **0001**.

La rappresentazione in base due del complemento alla base di **1110** è **0010**.

La rappresentazione in base due del complemento alla base di **1000** è **1000**.

Il sistema di rappresentazione per complemento alla base si è affermato perché consente di effettuare agevolmente le operazioni algebriche fondamentali.

Innanzitutto osserviamo che l'operazione di *cambio di segno* di un numero si riduce all'operazione di complementazione della sua rappresentazione. Questo è ovvio se il numero è non negativo e deriva, nel caso di numero negativo, dal fatto che il complemento del complemento di un naturale è il naturale stesso. Essendo l'intervallo dei valori rappresentabili per complemento alla base non simmetrico rispetto allo zero, l'operazione di cambio di segno del più piccolo numero ($-b^p$ **div** *2*) dà luogo a supero di capacità.

Esempi

La rappresentazione per complemento alla base dieci di **-1** è 99, che complementata dà **01**, rappresentazione di **+1**.

La rappresentazione per complemento alla base dieci di **1** è **01**, che complementata dà **99**, rappresentazione di **-1**.

La rappresentazione per complemento alla base due di **7** è **0111**, che complementata dà **1001**, rappresentazione di **-7**.

La **proprietà fondamentale** della rappresentazione per complemento alla base è la seguente:

Sommando aritmeticamente le rappresentazioni per complemento alla base di due numeri relativi, e ignorando l'eventuale riporto dalla posizione più significativa, si ottiene la rappresentazione per complemento della somma dei due numeri. Questo sempre che non vi sia supero di capacità, segnalato dalla discordanza tra il segno del risultato e quello comune dei due addendi.

Infatti:

a) se $x >= 0$ e $y >= 0$, la cosa è ovvia;

b) se $x >= 0$ e $y < 0$ (o se $x < 0$ e $y >= 0$), sommando le rappresentazioni si ottiene $(x + y) + b^p$ che è uguale a $x + y$, se $x + y >= 0$ (per il supero di capacità), al complemento di $-(x + y)$, se $x + y < 0$;

c) se $x < 0$ e $y < 0$, sommando le rappresentazioni si ottiene $(x + y) + b^p + b^p$ che è uguale a $(x + y) + b^p$ (per il supero di capacità) e quindi al complemento di $-(x + y)$.

Come già detto, il supero di capacità è segnalato dalla discordanza tra il segno del risultato e quello comune dei due addendi. Questa regola è equivalente all'altra che afferma la presenza di supero di capacità nel caso di diversità dei riporti in ingresso e in uscita relativi all'ultimo stadio di addizione (ossia a quello che somma i segni).

Esempi

$0010_2 + 0011_2 = 0101_2$ [2 + 3 = 5]

```
riporto 00100
         0010 +
         0011 =
        -------
somma    0101
```

$0010_2 + 1101_2 = 1111_2$ [2 + (-3) = -1]

```
riporto 00000
         0010 +
         1101 =
        -------
somma    1111
```

$0011_2 + 1110_2 = 0001_2$ [3 + (-2) = 1]

```
riporto 11100
         0011 +
         1110 =
        -------
somma    0001
```

$1110_2 + 1101_2 = 1011_2$ [-2 + (-3) = -5]

```
riporto 11000
         1110 +
         1101 =
        -------
somma    1011
```

$0110_2 + 0111_2 =$ *trabocco* [6 + 3 = *trabocco*]

```
riporto 01100
         0110 +
         0111 =
        -------
somma    1101
```

Rappresentazione per complemento diminuito dei numeri relativi

Anche se non riveste più grande importanza, si segnala che in passato è stata talvolta impiegata, invece che la rappresentazione per complemento alla base, la **rappresentazione per complemento diminuito** in cui, al posto del complemento alla base, si fa uso del complemento diminuito, così come definito nel paragrafo precedente.

In questa rappresentazione:

a) Lo zero ha due distinte rappresentazioni (la sequenza di tutti bit *0* e quella di tutti bit *1*).

b) L'intervallo dei valori rappresentabili è simmetrico rispetto allo zero.

c) L'operazione di cambio di segno si riduce alla complementazione bit a bit.

d) La rappresentazione di una somma algebrica si ottiene sommando aritmeticamente le rappresentazioni per complemento diminuito e usando l'eventuale riporto dalla posizione più significativa come riporto in ingresso alla posizione meno significativa (*end-around carry*).

e) Il supero di capacità è segnalato, come nel caso della rappresentazione per complemento alla base, dalla discordanza tra il segno del risultato e quello comune dei due addendi.

Esempi

$0010_2 + 1100_2 = 1110_2$ [2 + (-3) = -1]

```
riporto 00000
         0010 +
         1100 =
        -------
somma    1110
```

$0011_2 + 1101_2 = 0001_2$ [3 + (-2) = 1]

```
riporto 11110              00000
         0011 +             0000 +
         1101 =             0001 =
        -------            -------
somma    0000               0001
```

$1011_2 + 1100_2 = 1010_2$ [-4 + (-3) = -7]

```
riporto 10000              01110
         1011 +             0111 +
         1100 =             0001 =
        -------            -------
somma    0111               1000
```

$1010_2 + 1100_2 = trabocco$ [-5 + (-3) = *trabocco*]

```
riporto 10000              00000
         1010 +             0110 +
         1100 =             0001 =
        -------            -------
somma    0110               0111
```

Rappresentazione per eccesso dei numeri relativi

La **rappresentazione per eccesso** è ancora una rappresentazione *senza segno*, in cui un numero relativo viene rappresentato per mezzo (della rappresentazione) di un numero naturale che si ottiene sommando al numero relativo una costante pari al modulo del più piccolo numero rappresentabile, ossia facendo scorrere l'intervallo dei valori verso destra, in modo da portare l'estremo inferiore a coincidere con lo zero. Se la costante che viene sommata è *k*, si parla di **rappresentazione eccesso-*k***.

Pertanto, se b è la base pari prescelta e p sono le cifre a disposizione, l'intervallo rappresentabile è $[-b^p \textbf{ div } 2, +b^p \textbf{ div } 2 - 1]$, con $k = b^p \textbf{ div } 2$.

In questa rappresentazione:

a) Lo zero ha una sola rappresentazione (quella di k, ossia la sequenza *10...0* con *p-1* bit *0*).

b) L'intervallo dei valori rappresentabili non è simmetrico rispetto allo zero.

c) Le rappresentazioni di uno stesso numero per complemento alla base e per eccesso differiscono esclusivamente nel bit segno. Pertanto, nella rappresentazione per eccesso, il bit più significativo può essere ancora interpretato come bit segno, anche se questa volta il segno positivo è codificato con *1* e quello negativo con *0*.

d) *Il confronto di due numeri relativi rappresentati per eccesso può essere effettuato semplicemente confrontandone le rappresentazioni*.

Estensione dei sistemi di numerazione posizionale ai numeri reali

I sistemi di numerazione posizionale in base b possono essere estesi al fine di consentire la codifica anche dei numeri reali. Sussiste infatti il seguente:

> **Teorema**: Fissata una base b (naturale maggiore di uno), per ogni reale x non negativo e minore di 1, esiste una ed una sola successione (x_j) di interi non negativi minori di b tali che:
>
> $$(5)\quad x = \sum_{j \in [1,\ +\infty[} x_j \times b^{-j} \qquad (b^{-j} = 1/b^j)$$
>
> Gli interi della successione sono forniti da:
>
> $$(6)\quad x_j = \lfloor x \times b^j \rfloor - \lfloor x \times b^{j-1} \rfloor \times b \qquad \forall j \in [1,\ +\infty[$$

Se r è un qualsiasi reale non negativo, poiché:

$$r = \lfloor r \rfloor + (r - \lfloor r \rfloor) = \lfloor r \rfloor + x$$

con x parte frazionaria di r:

$$x = r - \lfloor r \rfloor < 1$$

si ha, utilizzando le (1) e (5):

$$(7)\quad r = \sum_{i \in [0,\ k-1]} c_i \times b^i + \sum_{j \in [1,\ +\infty[} x_j \times b^{-j}$$

che può anche ovviamente scriversi, ponendo $c_{-i} = x_i$:

$$(8)\quad r = \sum_{i \in]-\infty,\ k-1]} c_i \times b^i$$

Pertanto, si può assumere come rappresentazione in base b del numero reale non negativo r la stringa di lunghezza generalmente infinita:

$$(9)\quad r_b = c_{k-1}\ c_{k-2}\ \ldots\ c_1\ c_0 . c_{-1}\ c_{-2}\ \ldots\ c_{-i}\ \ldots$$

ove il punto che separa le rappresentazioni della parte intera e di quella frazionaria di r prende il nome di **punto frazionario**.

Rappresentazioni finite ed infinite dei numeri reali

La (9) mostra che la rappresentazione in base b della parte frazionaria di un numero reale non negativo ha generalmente lunghezza infinita. Ci proponiamo di caratterizzare i numeri reali

per i quali è possibile individuare rappresentazioni di lunghezza finita, in cui cioè da un certo punto in avanti, le cifre della rappresentazione della parte frazionaria sono tutte nulle. A tal fine sono fondamentali le definizioni e proprietà che seguono.

La rappresentazione in base *b* di *r* si dice **periodica** se esistono due interi *m* e *h* tali che:

$$(10)\quad c_{-j} = c_{-(j+h)} \quad \forall j \geq m$$

In questa ipotesi, si dice **periodo** della rappresentazione di r in base b la sequenza di cifre della parte frazionaria che si ripete all'infinito:

$$(11)\quad c_{-m}\ c_{-(m+1)}\ \ldots\ c_{-(m+h-1)}$$

e, se il più piccolo degli interi m, diciamolo p, è maggiore di 1, si dice **antiperiodo** della rappresentazione di r in base b la sequenza di cifre della parte frazionaria che precede la prima occorrenza del periodo:

$$(12)\quad c_{-1}\ c_{-2}\ \ldots\ c_{-(p-1)}$$

Per convenzione, una rappresentazione periodica si scrive indicando una sola volta il periodo e racchiudendolo tra parentesi, per ricordare che in realtà esso si ripete un numero infinito di volte:

$$(13)\quad r_b = c_{k-1}\ \ldots\ c_0 . c_{-1}\ \ldots\ c_{-(p-1)}\ (c_{-p}\ \ldots\ c_{-(p+h-1)})$$

In particolare, se la rappresentazione di r in base b è **periodica di periodo 0**, ossia se le cifre che seguono l'eventuale antiperiodo sono tutte nulle, allora e solo allora il numero è rappresentabile con un numero finito di cifre:

$$(14)\quad r_b = c_{k-1}\ \ldots\ c_0 . c_{-1}\ \ldots\ c_{-(p-1)}$$

Al fine di caratterizzare i numeri rappresentabili con un numero finito di cifre è fondamentale ricordare che *la periodicità è una proprietà del numero e non della sua rappresentazione*, ossia non dipende dalla base di numerazione impiegata. Infatti, **un numero reale non negativo *r* è periodico se e solo se è razionale**, ossia esprimibile come rapporto di due naturali *p*, *q*: $r = p/q$, con $q > 0$.

Invece, la periodicità di periodo *0* dipende anche dalla base di numerazione prescelta. Infatti, nella rappresentazione in base *b* di *r*, il periodo è nullo se e solo se *r* è esprimibile come rapporto: $r = p/b^k$, con *p* e *k* naturali.

Esempio

Il numero razionale *1/3* ha rappresentazione decimale $0.(3)_{10}$, di periodo *3* e rappresentazione ternaria 0.1_3, di periodo *0*.

Conversioni di base dei numeri reali

Fissata una base *b*, si pone il problema di determinare per ogni numero reale *x* non negativo e minore di *1*, la sua rappresentazione in base *b*. La formula (6) evidenzia che le cifre c_{-1}, c_{-2}, ... sono ottenibili con ripetute moltiplicazioni:

$$c_{-1} = \lfloor x \times b \rfloor$$
$$c_{-2} = \lfloor \mathrm{fraz}(x \times b) \times b \rfloor$$
$$c_{-3} = \lfloor \mathrm{fraz}(\mathrm{fraz}(x \times b) \times b) \times b \rfloor$$
$$\ldots$$

ove **fraz(z) = z - ⌊ z ⌋** è la parte frazionaria di ***z***. Pertanto, le prime ***s*** cifre della rappresentazione in base ***b*** di ***x*** vengono calcolate ripetendo ***s*** volte il passo seguente, in cui si assume la parte frazionaria iniziale uguale a ***x***:

> all'*i*-mo passo si moltiplica la parte frazionaria corrente per ***b***, ottenendo la parte intera c_{-i} e la nuova parte frazionaria.

È evidente che si ottiene l'esatta rappresentazione solo se ***x*** è periodico di periodo ***0*** nella base *b* e se l'antiperiodo ha un numero di cifre non maggiore di *s*.

Per eseguire concretamente il metodo delle moltiplicazioni successive occorre che *n* e *b* siano rappresentati nella base *B* la cui aritmetica si intende usare. Pertanto, il metodo delle moltiplicazioni successive effettua in concreto la conversione dalla base *B* alla base *b*, operando secondo l'aritmetica della base di partenza *B*. Esso, se eseguito manualmente, è adatto al caso *B = dieci*.

Esempio

$0.6_{10} = 0.46314_{8}$ **(per s = 5)**

	8
0.6	4
0.8	6
0.4	3
0.2	1
0.6	4

In realtà la tabella mostra che $0.6_{10} = 0.(4631)_8$.

Per usare l'aritmetica della base di arrivo, si può adoperare direttamente la (5). Per evitare le elevazioni a potenza è possibile operare con ripetute divisioni:

$$((\ldots (c_{-s}/b + c_{-(s-1)})/b + \ldots + c_{-2})/b + c_{-1})/b$$

Il metodo delle divisioni ripetute effettua quindi la conversione dalla base *B* alla base *b*, operando secondo l'aritmetica della base di arrivo *b*. Esso, se eseguito manualmente, è adatto al caso *b = dieci*.

Esempio

$0.33_4 = 0.9375_{10}$

```
(((3/4 + 3)/4 =
3.75/4 =
0.9375
```

Rappresentazione in virgola mobile dei numeri reali

Nei calcoli scientifici ci si trova quasi sempre di fronte a due esigenze contrastanti:

- da una parte, l'intervallo dei numeri usati è molto grande (potendo ad esempio andare dal nanosecondo all'età della terra, con un intervallo superiore a 10^{26}, ovvero dalla massa dell'elettrone a quella del sole, con un intervallo superiore a 10^{60}), con la conseguente esigenza di rappresentazioni a moltissime cifre;

- dall'altra, la precisione con cui i numeri sono noti è di norma limitata e spesso non è migliore di qualche per cento (ad esempio l'età della terra è stimata in 4500 milioni di anni con la precisione del 5%; questo significa che il valore vero è compreso tra $4500 \times (1 - 0.05)$ e $4500 \times (1 + 0.05)$, ossia tra 4275 e 4725 milioni di anni, e che delle dieci cifre decimali necessarie per esprimerlo solo quattro sono significative), con la conseguenza che gran parte delle cifre usate sono in realtà non significative e devono essere scartate al momento della visualizzazione dei risultati.

Nasce quindi l'esigenza di un *sistema di rappresentazione dei numeri in cui l'intervallo di valori esprimibile risulti indipendente dal numero di cifre significative*. Questo sistema è quello impiegato nella fisica, nella chimica e nell'ingegneria e comunemente noto come ***notazione scientifica***, in cui un numero reale *r* può essere espresso mediante una qualsiasi coppia di numeri <*s*, *e*>, con *s* **significando (**altrimenti detto ***mantissa*** o, quando di valore minore dell'unità, ***frazione***) reale ed *e* **esponente** intero, tali che:

$$r = s \times 10^e$$

Esempi

Alcuni esempi di numeri reali rappresentati usando la notazione scientifica (con significando a quattro cifre di cui una per la sua parte intera) sono:

$$3.14 = 0.314 \times 10^{+1} = 3.140 \times 10^{0}$$

$$0.000001 = 0.100 \times 10^{-5} = 1.000 \times 10^{-6}$$

$$1941 = 0.194 \times 1^{+4} = 1.941 \times 10^{+3}$$

In generale, esiste più di una forma per rappresentare uno stesso numero. Tra queste, quella in cui la cifra più significativa del significando (negli esempi la cifra intera) è diversa da *0* si dice ***forma normalizzata***. La forma normalizzata ha il pregio di rappresentare i numeri con la massima precisione possibile, evitando che una o più cifre del significando rimangano di fatto inutilizzate (confrontare le due rappresentazioni di 1941 negli esempi precedenti). Inoltre, richiedere che un numero diverso da zero sia rappresentato in forma normalizzata rende la rappresentazione unica.

La versione della notazione scientifica impiegata nei sistemi digitali prende il nome di ***rappresentazione in virgola*** (o, rifacendoci alle convenzioni adottate, ***punto frazionario***) ***mobile***. Al fine di fissare una rappresentazione in virgola mobile, occorre scegliere:

- la *base* di numerazione *b*, sempre pari e di norma scelta tra le piccole potenze di *2*;
- la *precisione p*, ossia il numero di cifre in base *b* del significando;
- quante delle *p* cifre sono riservate alla parte intera del significando, ossia la posizione del punto frazionario implicito;
- il minimo esponente e_{min} e il massimo esponente e_{max}.

Scelta la rappresentazione, vengono detti ***numeri in virgola mobile*** i soli numeri reali che risultino esprimibili esattamente in essa.

Esempio

Al fine di concretizzare le idee, fissiamo la seguente rappresentazione in virgola mobile:

- base di numerazione `10`;

- precisione **3**;
- cifre della parte intera del significando 0, ossia significando frazionario;
- esponente compreso tra **-99** e **+99**.

Osserviamo che, a parte lo zero, con 3 cifre di precisione e 199 esponenti diversi, limitandoci alle sole rappresentazioni normalizzate, la rappresentazione scelta consente di esprimere $9\times10\times10\times199=179100$ numeri in virgola mobile senza segno differenti. Pertanto, considerando il segno e lo zero, i numeri i virgola mobile rappresentabili sono in tutto $2\times179100+1=358201$. Non tutti i numeri reali, che formano un continuo sulla linea retta dei reali, sono quindi rappresentabili, ma solo un loro sottoinsieme finito. Dividendo la linea retta dei reali nelle seguenti sette regioni:

1. Numeri negativi minori di -0.999×10^{99}
2. Numeri negativi compresi tra -0.999×10^{99} e -0.100×10^{-99}
3. Numeri negativi con valore minore di 0.100×10^{-99}
4. Zero
5. Numeri positivi con valore minore di 0.100×10^{-99}
6. Numeri positivi compresi tra 0.100×10^{-99} e 0.999×10^{99}
7. Numeri positivi maggiori di 0.999×10^{99}

risulta agevole capire quali numeri reali sono rappresentabili come numeri in virgola mobile e quali no (limitatamente alla rappresentazione scelta):

- Nessun numero appartenente alle regioni 1 e 7 può essere espresso; qualora il risultato di un'operazione sia uno di questi numeri si ha un errore di ***overflow*** negativo o positivo e il risultato è scorretto.
- Rimanendo nell'ambito delle rappresentazioni normalizzate, nessun numero appartenente alle regioni 3 e 5 può essere espresso; qualora il risultato di un'operazione sia uno di questi numeri si ha un errore di ***underflow*** negativo o positivo e il risultato vero viene sostituito dal valore zero.
- Degli infiniti numeri reali appartenenti alle regioni 2 e 6, sono rappresentabili come numeri in virgola mobile solo 179.100 per ciascuna zona. Quando il risultato di un calcolo, pur essendo interno ad una delle zone non è uno dei numeri esattamente rappresentabili, esso viene in generale sottoposto ad un processo di ***arrotondamento*** (***rounding***), sostituendolo con il numero in virgola mobile ad esso più vicino. Così, 0.100×10^{3} diviso **3** viene arrotondato al valore 0.333×10^{2}, mentre 0.779×10^{3} diviso **2** viene arrotondato al valore 0.390×10^{3}.
- Nelle regioni 2 e 6, le differenze tra numeri in virgola mobile adiacenti crescono man mano che ci si muove verso i numeri più grandi. Così, $0.999\times10^{99}-0.998\times10^{99} >> 0.101\times10^{-99}-0.100\times10^{-99}$. Però, quando le differenze tra numeri consecutivi sono espresse in forma relativa, ossia dividendole per i numeri stessi, le differenze tendono a scomparire. In altri termini, l'***errore relativo*** introdotto dall'arrotondamento è approssimativamente lo stesso sia per i piccoli che per i grandi numeri.

Le considerazioni svolte nell'esempio hanno validità generale, in quanto cambiamenti nei numeri di cifre del significando e dell'esponente modificano semplicemente i confini delle

regioni 2 e 6 e la quantità dei numeri rappresentabili. In particolare, al crescere del numero di cifre del significando aumenta la densità dei numeri in virgola mobile e quindi la precisione della rappresentazione, mentre al crescere del numero di cifre dell'esponente aumenta l'intervallo dei valori rappresentabili.

Lo standard 754 dell'IEEE

Nell'anno 1985 l'IEEE (The Institute of Electrical and Electronics Engineering) pubblicò uno standard per la rappresentazione in virgola mobile, allo scopo di correggere la situazione che vedeva ogni costruttore impiegare differenti rappresentazioni ed algoritmi per l'effettuazione delle operazioni. Lo standard dell'IEEE si è largamente imposto ed è quindi ormai adoperato da tutti i costruttori di processori. Lo standard definisce tre diversi formati, rispettivamente in precisione singola (32 bit), doppia (64 bit) ed estesa (80 bit). In questo paragrafo descriviamo esclusivamente il formato in singola precisione, che risulta così caratterizzato:

1. La base prescelta è *2*, la precisione è *24* e la posizione del punto frazionario implicito è alla destra della prima cifra binaria del significando, in modo che il significando *s* di un numero normalizzato risulti compreso tra *1* e *2*: *1<=s<2*. L'esponente ha valori estremi uguali a *-126* e *+127*.
2. I 32 bit disponibili sono divisi in 3 campi, rispettivamente riservati, da sinistra verso destra:
   - 1 bit al segno del numero in virgola mobile (bit 0 per il +, 1 per il -);
   - 8 bit all'esponente rappresentato per eccesso 127;
   - 23 bit al significando, in quanto il bit più significativo (quello alla sinistra del punto frazionario), essendo per i numeri normalizzati sempre uguale ad 1, non viene esplicitamente memorizzato (*bit nascosto*, *hidden bit*).

   La rappresentazione scelta consente di confrontare due numeri in virgola mobile come se fossero dei semplici interi rappresentati per segno e valore.
3. I valori estremi dell'esponente (*0* e *255*, letti come interi senza segno) sono riservati. Pertanto, solo i valori dell'esponente compresi tra *1* (da interpretare come *1-127 = -126*) e *254* (da interpretare come *254-127 = 127*) individuano numeri normalizzati. Il più piccolo numero positivo normalizzato è $\mathtt{1.0 \times 2^{-126}}$, mentre il più grande è $\mathtt{1.1\ldots1 \times 2^{127}}$ leggermente minore di $\mathtt{2.0 \times 2^{127} = 2^{128}}$.
4. L'esponente *0* individua i numeri denormalizzati, in cui il bit nascosto va assunto uguale a *0* e l'esponente pari a *-126*. Tra i numeri denormalizzati vi è lo *0*, rappresentato da un significando tutto nullo. Quando il risultato di un'operazione è, in valore, minore del più piccolo numero normalizzato, lo standard prevede che esso sia rappresentato in forma denormalizzata. In questo modo risulta possibile rappresentare numeri più piccoli del più piccolo numero normalizzato. Il più piccolo numero positivo denormalizzato è $\mathtt{0.0\ldots1 \times 2^{-126} = 2^{-149}}$, mentre il più grande è $\mathtt{0.1\ldots1 \times 2^{-126}}$ leggermente più piccolo di $\mathtt{1.0 \times 2^{-126} = 2^{-127}}$.
5. L'esponente *255* rappresenta l'infinito (±∞) quando è abbinato ad un significando nullo, mentre rappresenta il valore speciale **`NaN`** (*Not a number*) quando è abbinato ad un qualsiasi significando non nullo. Il valore **`NaN`** viene generato tutte le volte che il risultato di un'operazione è indefinito (ad esempio, ∞/∞).