Rappresentazione di numeri reali

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

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

Rappresentazione nei calcolatori

Nei calcolatori, non è purtroppo possibile rappresentare numeri con infinite cifre, come possono essere i reali. I numeri che possono essere rappresentati sono solo i numeri frazionari e comunque sempre con precisione finita

Numeri frazionari

Il sistema posizionale funziona anche per i numeri frazionari, usando potenze negative della base. Nel caso decimale, il numero 0,341 significa: "tre decimi più quattro centesimi più un millesimo". Le cifre dopo la virgola indicano quindi il numero di decimi, centesimi, millesimi, ecc., ossia vanno moltiplicate per 1/10, 1/100, 1/1000, ecc. Questi numeri sono le potenze negative di dieci:

$1/10=10^{-1}$

$1/100=10^{-2}$

$1/1000=10^{-3}$

In generale, un numero frazionario

$$0,c_1c_2c_3\ldots$$

ossia uno che ha dopo la virgola le cifre c_1, c_2, c_3, ecc., significa:

$$c_1\times10^{-1} + c_2\times10^{-2} + c_3\times10^{-3} + \ldots$$

Numeri frazionari

Questo nel caso decimale. Si estende a base b usando b al posto di dieci:

$0,c_1c_2c_3\ldots = c_1\times b^{-1} + c_2\times b^{-2} + c_3\times b^{-3} + \ldots$

Nel caso si voglia convertire questo numero in decimale, basta effettuare i calcoli. Per esempio, il numero ottale 0,645 vale:

$0,645 = 6\times 8^{-1} + 4\times 8^{-2} + 5\times 8^{-3} = 6/8 + 4/(8\times 8) + 5/(8\times 8\times 8) = 0,822265625$

Il numero binario 0,110101 si converte nello stesso modo:

$0,110101 = 1\times 2^{-1} + 1\times 2^{-2} + 0\times 2^{-3} + 1\times 2^{-4} + 0\times 2^{-5} + 1\times 2^{-6} = 1/2 + 1/4 + 0/8 + 1/16 + 0/32 + 1/64 = 0,828125$

Nel caso di conversione fra due basi che sono l'una potenza dell'altra, come nel caso di 2 e 16, allora si possono raggruppare le cifre come per la conversione fra numeri interi. Per esempio, il numero binario 0,11011000 è uguale all'esadecimale che ha cifre 1101=D e 1000=8, ossia 0,D8.

Esercizi

Convertire il numero binario 110,101 in decimale.

Basta sviluppare le potenze relative:

$1 \quad 1 \quad 0 \; , \quad 1 \quad 0 \quad 1 =$

$1\times2^{2} + 1\times2^{1} + 0\times2^{0} + 1\times2^{-1} + 0\times2^{-2} + 1\times2^{-3} =$

$4 + 2 + 0 + 0{,}5 + 0 + 0{,}125 =$

$6{,}625$

Esercizi

Convertire il numero 32,3 da base sei a base dieci.

Il sistema è lo stesso, ma ora si usano le potenze di sei al posto di quelle di due:

$3 \quad 2 \quad , \quad 3 \quad =$

$3\times6^{1} \quad + \quad 2\times6^{0} \quad + \quad 3\times6^{-1} \quad =$

$18 \quad + \quad 2 \quad + \quad 0{,}5 =$

$20{,}5$

Esercizi

Convertire il numero 110,101 da binario a base sedici.

In questo caso basta raggruppare i bit a blocchi di quattro:

0110 , 1010 =

----- ------

6 , A

Si ottiene quindi 6,A.

Conversione da decimale ad altra base

Per effettuare la conversione da decimale ad altra base si usa il metodo delle moltiplicazioni successive, simile a quello delle divisioni successive visto per la conversione degli interi.

Per convertire un numero da decimale a base b:

- moltiplicare il numero per b;
- togliere la parte intera, che è la prima cifra;
- moltiplicare quello che rimane per b;
- …

Conversione da decimale ad altra base

Per esempio, i numeri decimale 0,6875 si converte in binario come segue:

- 0,6875×2 = 1,375
  la parte intera è uno; sottratta, lascia 0,375
- 0,375×2 = 0,75
  la parte intera è zero;
- 0,75×2 = 1,5
  la parte intera è uno; si toglie, lasciando 0,5
- 0,5×2 = 1
  la parte intera è uno; togliendola, rimane 0

Le cifre binarie sono le parti intere in ordine (non in ordine opposto co me nella conversione di interi). In questo caso: 0,1011. La procedura termina quando il valore che rimane è zero.

Conversione da decimale ad altra base

Non è però sempre detto che la procedura termini. Per esempio, la conversione di 0,2 produce:

0,2×2=0,4

0,4×2=0,8

0,8×2=1,6 parte intera 1

0,6×2=1,2 parte intera 1

0,2×2=0,4

…

Si è tornati a 0,2, per cui si ripete tutto uguale, tornando ancora a 0,2. Il numero binario che si ottiene è quindi 0,001100110011…, dove le cifre 0011 si ripetono all'infinito.

Conversione da decimale ad altra base

Questo avviene perché 0,2=1/5. Dato che 10=2×5, ossia il dieci contiene il cinque come numero primo, la rappresentazione in base dieci è finita: 0,2=1/5=2/10. D'altra parte, due non contiene cinque come numero primo, per cui la rappresentazione di 0,2=1/5 in base due non è finita: non c'è modo di ottenere 1/5 come somma delle potenze del due 1/2, 1/4. 1/8, ecc.

Questa proprietà vale in generale: se la base di arrivo contiene tutti i numeri primi di quella di partenza allora un numero frazionario con cifre finite si converte sempre in uno con cifre finite. Nel caso contrario, questo può o meno avvenire a seconda dei casi: 1,375 si converte in un numero binario con cifre finite, 0,2 no.

Esercizi

Convertire il numero 0,78125 in binario.

Si parte dal numero e lo si moltiplica per due. Di questo si elimina la parte intera (se c'è) e si ripete l'operazione:

0,78125 per 2 fa 1,56250, parte intera 1
0,5625 per 2 fa 1,125, parte intera 1
0,125 per 2 fa 0,25, parte intera 0
0,25 per 2 fa 0,5, parte intera 0
0,5 per 2 fa 1, parte intera 1

Il risultato si ottiene prendendo le parti intere nell'ordine, in questo caso si ottiene 0,11001.

Notare che, al contrario della conversione dei numeri interi, queste cifre sono nello stesso ordine in cui sono state ottenute.

Esercizi

Convertire 0,1875 in binario.

0,1875 per 2 fa 0,3750, parte intera 0

0,3750 per 2 fa 0,7500, parte intera 0

0,7500 per 2 fa 1,5000, parte intera 1

0,5000 per 2 fa 1,0000, parte intera 1

Il numero in binario è quindi 0,0011.

Esercizi

Esempio: convertire il numero 0,640625 in ottale.

Anche in questo caso si usa il metodo delle moltiplicazioni successive, ma si moltiplica per otto invece che per due:

0,640625 per 8 fa 5,125, parte intera 5

0,125 per 8 fa 1, parte intera 1

Il numero in ottale è quindi 0,51.

Correttezza del sistema di conversione

La correttezza del sistema di conversione deriva dalla definizione di numero nel sistema posizionale: se n va convertito in base b, quello che si vuole ottenere è la sequenza di cifre $0,c_1c_2c_3\ldots$ tale che:

$n = c_1 \times b^{-1} + c_2 \times b^{-2} + c_3 \times b^{-3} + \ldots$

Moltiplicando entrambi i membri di questa equazione per b si ottiene:

$n \times b = c_1 \times b^0 + c_2 \times b^{-1} + c_3 \times b^{-2} + \ldots =$

$= c_1 + 0,c_2c_3\ldots$

L'ultimo passaggio si può fare per la regola che dice il valore di un numero in base b: il valore di $0,c_2c_3\ldots$ è proprio $c_2 \times b^{-1} + c_3 \times b^{-2} + \ldots$.

Dato che c_1 è intero mentre $0,c_2c_3\ldots$ è minore di uno, $n \times b$ ha come parte intera c_1. Togliendola, rimane $0,c_2c_3\ldots$, a cui si può ancora moltiplicare b per ottenere c_2.

Riassumendo:

la parte intera di $n \times b$ è la prima cifra frazionaria di n in base b. Tolta questa da $n \times b$, si può moltiplicare ancora per b per ottenere la seconda cifra, ecc.

Numeri con parte sia intera che frazionaria

Per convertire un numero composto sia da una parte intera che da una frazionaria, si convertono separatamente le due parti. In base b vale:

$$c_k...c_0,c_{-1}c_{-2}... = c_k \times b^k + ... \; c_0 \times b^0 + c_{-1} \times b^{-1} + c_{-2} \times b^{-2} + ... = \sum_{i=k,k-1,...} c_k \times b^k$$

Per esempio, per convertire 12,6875 da decimale a binario, si converte prima il 12 e poi 0,6875. Per la parte intera si usa il metodo delle divisioni successive:

12 diviso 2 fa 6 con resto di 0

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

La parte intera è 1100 perché questi sono i resti in ordine inverso. Per la parte frazionaria si effettuano le moltiplicazioni successive su 0,6875:

0,6875 per 2 fa 1,3750, parte intera 1

0,3750 per 2 fa 0,7500, parte intera 0

0,7500 per 2 fa 1,5000, parte intera 1

0,5000 per 2 fa 1,0000, parte intera 1 → $(12,6875)_{10} = (1100,1011)_2$

Esercizi

Convertire 30,625 da decimale a binario.

Si converte prima la parte intera 30 usando il metodo delle divisioni successive:

30 | 0

15 | 1

7 | 1

3 | 1

1 | 1

Quindi la parte intera vale 11110. La parte frazionaria si ottiene moltiplicando ripetutamente 0,625 per due e togliendo ogni volta la parte intera del risultato.

0,625 | 1

0,25 | 0

0,5 | 1

Il risultato della conversione è quindi 11110,101.

Esercizi

Convertire il numero 30,12 da decimale a base cinque

Si usando sempre il metodo delle divisioni successive per la parte intera e delle moltiplicazioni successive per la parte frazionaria, ma questa volta il numero per cui dividere e moltiplicare è cinque invece di due. Per la parte intera:

30 | 0

6 | 1

1 | 1

La parte intera in base cinque è quindi 110. Per la parte frazionaria si moltiplica 0,12 per cinque. E si prosegue moltiplicando sempre per cinque la parte frazionaria:

0,12 | 0

0,6 | 3

La parte frazionaria si ottiene prendendo le parti intere nell'ordine, in questo caso 0,03.

L'intero numero vale quindi 110,03.

Virgola fissa

Come già visto per gli interi, di solito i numeri vengono rappresentati in binario con un numero prefissato di bit, per esempio sessantaquattro. Questo vuol dire che non è possibile rappresentare numeri molto grandi, ma nemmeno numeri che abbiano molte o infinite cifre dopo la virgola.

Nel caso dei numeri frazionari, un sistema che si può usare è quello di decidere in anticipo quali dei sessantaquattro bit verranno usati per la parte intera e quali per la parte frazionaria. Per esempio, si possono usare trentadue per la parte intera e trentadue per la parte frazionaria, oppure quarantotto e sedici.

Il problema di questo sistema è che non si riescono a rappresentare numeri molto grandi o numeri molto piccoli. Nel caso della suddivisione 48/16, il massimo numero rappresentabile è 281474976710655,999984741210937 5 (duecentoottantunomilaquattrocentosettantaquattro miliardi, ecc.) il minimo è 0,000015258789062 5. Questo può apparire sufficiente, ma esistono applicazioni in cui non lo è. Si tenga presente che questi limiti valgono non solo per i numeri che hanno un significato in sé, ma anche tutti i risultati intermedi delle elaborazioni.

Virgola fissa

In questa rappresentazione in virgola fissa, il numero massimo è quarantotto uni virgola sedici uni. Il numero immediatamente inferiore rappresentabile ha l'ultimo uno frazionario rimpiazzato da zero. I due numeri in decimale sono:

281474976710655,9999847412109375

281474976710655,999969482421875

Su numeri dell'ordine dei centomila miliardi, questa rappresentazione arriva a distinguere due numeri che differiscono per la quinta cifra dopo la virgola, ossia meno di un millesimo. Nella maggior parte delle applicazioni, i due numeri si possono considerare di fatto indistinguibili. In altre parole, non ha senso usare due rappresentazioni diverse per essi.

La tecnica della virgola mobile consente di rappresentare un intervallo più ampio a scapito della concentrazione di numeri molto grandi rappresentabili. In altre parole, si possono rappresentare anche numeri più grandi di quelli visti sopra; al tempo stesso, due numeri molto grandi ma molto simili come quelli visti sopra vengono rappresentati nello stesso modo.

Esercizi

Convertire il numero 12,4 da base 10 a binario, virgola fissa con 4 cifre binarie intere e 6 decimali. Verificare il risultato convertendolo poi in decimale.

Si converte prima la parte intera, con il metodo delle divisioni successive:

12 | 0

5 | 1

2 | 0

1 | 1

Per la parte intera 1010 bastano quindi i quattro bit a disposizione. La parte frazionaria si converte con le moltiplicazioni successive:

0,4 | 0

0,8 | 1

0,6 | 1

0,2 | 0

0,4 | 0

0,8 | 1

Il sistema delle moltiplicazioni successive non è terminato, ma sono finiti i bit a disposizione per la parte frazionaria, ossia sei. Quindi, la rappresentazione in virgola fissa 4+6 bit del numero è 1010,011001.

Esercizi (cont. sol.)

Convertendo questo numero in decimale si ottiene:

1010,011001 =

$1\times2^3 + 1\times2^1 + 1\times2^{-2} + 1\times2^{-3}+1\times2^{-6}$ =

8 + 2 + 1/4 + 1/8 +1/64 =

8 + 2 + 0,25 + 0,125 + 0,015625 =

10,390625

Il numero di partenza era 10,4. La conversione ha diminuito questo numero, introducendo un errore pari a -0,009375.

Esercizi

Dire quali sono il massimo e il minimo numero diverso da zero rappresentabili senza segno a virgola fissa, tre bit di parte intera e tre bit di parte frazionaria.

Sono 111,111 e 000,001. Convertiti in decimale:

$111,111 = 1\times2^2 + 1\times2^1 + 1\times2^0 + 1\times2^{-1} + 1\times2^{-2} + 1\times2^{-3} = 4 + 2 + 1 + 1/2 + 1/4 + 1/8 = 7,875$

$000,001 = 1\times2^{-3} = 0,125$

Il secondo risultato implica che i numeri minori di 0,125 vengono tutti rappresentati come zero.

Esercizi

Convertire il numero 0,01 da base dieci a base due, virgola fissa con quattro bit di parte intera e quattro bit di parte decimale.

In questo caso la parte intera è zero, per cui non c'è bisogno di nessuna conversione. Per la parte frazionaria:

0,01 | 0
0,02 | 0
0,04 | 0
0,08 | 0

Il sistema delle moltiplicazioni successive non è terminato, ma i bit disponibili per la parte frazionaria (quattro) sono terminati. Questo significa che il numero $(0,01)_{10}$ viene rappresentato come zero in binario con questo numero di bit.

Esercizi

Dire quanti bit servono (almeno) per far sì che i due numeri 12,3456 e 12,3560 siano rappresentati in modo diverso in virgola fissa.

Per la rappresentabilità della parte intera, la conversione produce:

12 | 0

6 | 0

3 | 1

1 | 1

Quindi bastano quattro bit. Per poter rappresentare in modo diverso le parti frazionarie, basta effettuare le moltiplicazioni successive in parallelo fino al punto in cui si ottiene un bit diverso nei due casi.

0,3456 | 0	0,3560 | 0
0,6912 | 1	0,7120 | 1
0,3824 | 0	0,4240 | 0
0,7648 | 1	0,8480 | 1
0,5296 | 1	0,6960 | 1
0,0592 | 0	0,3920 | 0
0,1184 | 0	0,7840 | 1

Affinché i due numeri abbiano rappresentazioni diverse servono almeno sette bit di parte frazionaria.

Virgola mobile

Nella rappresentazione in virgola fissa si decide in anticipo il numero di bit da usare prima e dopo la virgola. In quella mobile, si usa sempre un numero prefissato totale di bit, ma la posizione della virgola all'interno di questi può variare. Dal momento che il numero cambia a seconda della posizione della virgola:

la posizione della virgola fa parte della rappresentazione del numero

Il numero viene quindi rappresentato come una parte chiamata **mantissa** che indica il suo valore (senza la virgola) e una parte chiamata **esponente** che indica la posizione della virgola al suo interno. Per esempio:

351#0

valore 351 con virgola in posizione zero, ossia 0,351

1023#1

valore 1023 con virgola in posizione uno, ossia 1,023

1023#2

valore 1023 con virgola in posizione due, ossia 10,23

21#5

valore 21 con virgola in posizione cinque, ossia 21000,0

7214#-3

valore 7214 con virgola in posizione meno tre, ossia 0,0007214

Virgola mobile

La posizione della virgola indica di quanto va moltiplicato il numero, considerato come un valore frazionario *0,cifre*. In particolare, per la posizione zero il numero rimane inalterato, per la posizione uno il numero è moltiplicato per dieci, per la posizione due si moltiplica per cento, ecc. In altre parole, la posizione indica una potenza di dieci da moltiplicare al valore iniziale.

351#0 è $0{,}351 \times 10^0 = 0{,}351$
1023#1 è $0{,}1023 \times 10^1 = 1{,}023$
1023#2 è $0{,}1023 \times 10^2 = 10{,}23$
21#5 è $0{,}21 \times 10^5 = 21000$
7214#-3 è $0{,}7214 \times 10^{-3} = 0{,}0007214$

In generale, nelle rappresentazioni in virgola mobile in base b, ogni numero viene rappresentato nel seguente modo:

$$n = m \times b^e$$

Invece di memorizzare n, vengono memorizzati la mantissa m e l'esponente e. L'esponente si può vedere come la posizione della virgola all'interno di m, assumendo che m sia un numero nella forma 0,cifre. Questa non è una limitazione, dato che ogni numero si può esprimere in questa forma, visto che è possibile moltiplicarlo per l'esponente per ottenere valori più piccoli e più grandi.

Virgola mobile

Il motivo per cui la rappresentazione in virgola mobile è quella usata di norma per i valori frazionari è che permette di memorizzare una gamma di valori più ampi, molto grandi e molto piccoli, a scapito di una perdita di precisione che è però piccola rispetto alla grandezza dei valori in esame. Per esempio:

fra 100 e 200 c'è una certa differenza;
fra 100000000100 e 100000000200 no.

In virgola fissa tutti e quattro i numeri sono rappresentati in modo esatto, sempre che la quantità di bit disponibile sia sufficiente.

Questo vuole dire che 100000000100 e 100000000200 potrebbero non essere rappresentabili, ma se lo sono allora sono sequenze di bit diverse.

Virgola mobile

In virgola mobile, 100000000100 e 100000000200 sono

100000000100#12
100000000200#12

Se i bit non bastano, si possono omettere le ultime cifre, limitandosi per esempio alle prime sei. Si otterrebbero così

$100000\#12 = 0{,}1\times10^{12}$
$100000\#12 = 0{,}1\times10^{12}$

Questi due numeri sono uguali, ma del resto erano già molto simili in origine. la differenza si può considerare in molti casi trascurabile. D'altra parte, i numeri 100 e 200 verrebbero rappresentati come diversi: 1#3 e 2#3.

Usando le sei cifre della mantissa più le due dell'esponente per rappresentare i numeri in virgola fissa, il totale di otto cifre non basterebbe per rappresentare i due numeri 100000000100 e 100000000200, che hanno dodici cifre. In altre parole, questi due numeri che sono rappresentati in forma approssimata con la virgola mobile non si possono rappresentare affatto con la virgola fissa.

Riassumendo, in virgola mobile è possibile rappresentare la "parte principale" di un numero (le sue prime cifre), trascurando eventualmente quello che resta se lo spazio a disposizione non consente di memorizzare tutto.

Esercizi

Dire quanto vale il numero decimale che ha mantissa 0,5235 ed esponente -2.

La definizione è:

numero = mantissa × $b^{esponente}$

In questo caso: $0{,}5235 \times 10^{-2}$ è uguale a 0,5235/100=0,005235.

Esercizi

Dire quanto vale il numero decimale 34576,12 in virgola mobile, con quattro cifre decimali di mantissa e una di esponente.

La mantissa e l'esponente si ottengono portando il numero nella forma 0,cifre. In questo caso, si arriva a un numero del genere spostando la virgola di cinque volte:

$34576{,}12 = 0{,}3457612 \times 10^5$

Quindi la mantissa è 0,3457 e l'esponente 5. Si noti che la mantissa era richiesta di quattro cifre, per questo le cifre 612 alla fine del numero sono state eliminate. La conversione opposta produrrebbe:

$0{,}3457 \times 10^5 = 0{,}3457 \times 10000 = 34570$

In altre parole, la rappresentazione in questo caso non solo non contiene le cifre dopo la virgola, ma nemmeno l'ultima cifra intera.

Esercizi

Convertire il numero decimale 0,000123119 in virgola mobile con quattro cifre di mantissa e due di esponente.

Ancora una volta, si tratta di moltiplicazioni o divisioni per potenze di dieci che portino il numero nella forma 0,cifra.

Nel caso specifico, servono tre spostamenti di virgola, questa volta verso destra invece che sinistra:

$0{,}000123119 = 0{,}123119 \times 10^{-3}$

La mantissa in quatto cifre è 0,1231, l'esponente -3.

Esercizi

Convertire 10034402341 con quattro cifre di mantissa e una di esponente.

Sempre con il solito sistema:

$10034402341 = 0{,}10034402341 \times 10^{11}$

Il numero non è rappresentabile perché l'esponente richiesto sarebbe di due cifre, mentre era specificato che doveva essere di una. Per la mantissa non ci sarebbero stati problemi, dato che bastava troncare il numero a quattro cifre: 0,1003.

In generale, se un numero non è rappresentabile è perché l'esponente richiesto è troppo grande o troppo piccolo, mai per una limitazione del numero di cifre della mantissa. Si può dire che l'esponente indica l'ordine di grandezza del numero, la mantissa il valore specifico; il valore specifico si può approssimare (eliminando le ultime cifre), ma l'ordine di grandezza va memorizzato in forma esatta.

Calcoli in virgola mobile

La rappresentazione in forma di mantissa ed esponente causa delle difficoltà aggiuntive quando occorre fare calcoli. Per esempio, i numeri 101#3 e 2#2 non si possono sommare facendo la somma delle mantisse:

101+2

errato, perché 101#3 significa 101 ma 2#2 significa 20;

0,101+0,2

errato, perché $101\#3=0{,}101\times10^3$ mentre $2\#2=0{,}2\times10^2$, e quindi 0,101 e 0,2 sono moltiplicati per due diverse potenze di dieci.

Dal momento che i numeri espressi possono essere molto grandi o molto piccoli, si preferisce evitare di ridurli alla forma esplicita (come si è fatto in questo esempio con 101 e 20), ma si utilizza un diverso sistema:

- si modifica il numero con esponente più piccolo in modo che abbia lo stesso esponente dell'altro;
- si sommano le mantisse;
- se il risultato è maggiore o uguale a uno, si divide per dieci e si aumenta l'esponente di uno.

Calcoli in virgola mobile

Nell'esempio, 2#2 ha l'esponente minore, quindi è quello che va modificato. Dal momento che 2#2 = 0,2×100 = 20, questo numero si può anche rappresentare come 0,02×1000. Ora che entrambi i numeri sono moltiplicati per mille, si può effettuare la somma delle mantisse:

0,101×1000 + 0,02×1000 = (0,101+0,02) × 1000

La somma 0,101+0,02 = 0,121 produce il risultato corretto: 0,121×1000 = 121#3.

In questo caso non è stato necessario il terzo passo della procedura, dal momento che il risultato 0,121 della somma delle mantisse era minore di uno. Questo può però succedere, come quando si sommano 95#2 e 7#1:

si uguagliano gli esponenti: 7#1 = 0,7×10 = 0,07×100 = 07#2;

si sommano le mantisse: 95+07 = 102;

il risultato non è 102#2 = 0,102×100 ma 1,02×100, ossia 102#3.

In altre parole, se il risultato della somma ha una cifra in più, occorre aumentare l'esponente di uno. Modificare l'esponente in modo che la mantissa abbia la forma 0,cifre si chiama normalizzazione.

In decimale, un numero nella forma 0,cifre, in cui la prima cifra è diversa da zero, risulta compreso fra 0,1 e 0,9999…, ossia fra 0,1 (incluso) e 1 (escluso).

Esercizi

Sommare i due numeri 32#4 e 123#2, assumendo quattro cifre di mantissa e una di esponente.

Occorre portare il secondo all'esponente del primo:

$0{,}123 \times 10^2 = 0{,}00123 \times 10^4$

Ora si possono sommare le mantisse: 0,32 + 0,00123 = 0,32123.

Considerato che le cifre di mantissa possono essere solo quattro, il risultato è 3212#4.

Esercizi

Sommare i numeri 994#5 e 671#3, assumendo sempre quattro cifre di mantissa e una di esponente.

Per effettuare la somma si porta l'esponente minore al livello del primo. In questo caso, il secondo va aumentato di due:

$0{,}671 \times 10^3 = 0{,}00671 \times 10^5$

Si può ora fare la somma delle mantisse: 0,994 + 0,00671 = 1,00071. Dato che questo numero non è minore di uno, l'esponente del risultato non è quello del primo addendo ma è l'esponente del primo addendo aumentato di uno:

$1{,}00071 \times 10^5 = 0{,}100071 \times 10^6$

Il risultato si ottiene tagliando le cifre di troppo dalla mantissa: 1#6.

Esercizi

Sommare i due numeri 9561#3 e 12#4 usando solo la somma fra interi. Si assumano quattro cifre di mantissa e una di esponente.

Come prima, si eleva l'esponente più basso al livello dell'altro, dividendo al tempo stesso la mantissa:

$0{,}9561 \times 10^3 = 0{,}09561 \times 10^4$

Volendo effettuare una somma fra interi, basta considerare per entrambe le mantisse il numero esatto di cifre disponibili, ossia quattro:

0,09561 → 0,0956

0,12 → 0,1200

Ora si può fare la somma di questi due numeri come fossero interi:

```
  1    ← riporti
 0956 +
 1200 =
--------
 2156
```

Il risultato della somma è il numero che ha questa mantissa e l'esponente dell'addendo maggiore, che era quattro. Il risultato è quindi 2156#4.

Esercizi

Sommare i due numeri 412#5 e 1#9, assumendo tre cifre di mantissa e una di esponente.

Si modifica il numero che ha la mantissa più piccola, in questo caso il primo: 412#5=0000412#9. Ora le due mantisse si possono sommare: 0,0000412+0,1=0,1000412. Data la limitazione a tre cifre della mantissa, di questo numero si possono tenere solo le cifre 0,100. Il risultato è quindi 1#9.

In questo caso il risultato viene uguale al secondo numero. Il motivo è che con questo numero di cifre della mantissa, il primo viene considerato in proporzione trascurabile.

Algoritmo di somma

Le operazioni necessarie per la somma sono facili da realizzare in pratica. Il cambio di esponente è una divisione della mantissa per una potenza di dieci e un aumento dell'esponente di questo valore. Per esempio:

$0{,}9561 \times 10^3 = 0{,}09561 \times 10^4$

Il numero 9561#3 diventa quindi 09561#4: è stato aggiunto uno zero all'inizio della mantissa e l'esponente è stato aumentato di uno. Allo stesso modo, per portare 42#5 a esponente 8, bisogna aggiungere tre zeri alla mantissa: 42#5=00042#9.

In generale, per aumentare l'esponente di x, bisogna aggiungere x zeri davanti alla mantissa.

Per la somma, due numeri 0,cifre e 0,altrecifre si possono sommare come se fossero interi, aggiungendo abbastanza zeri in fondo in modo che il numero totale di cifre sia lo stesso:

0,1342 + 0,01 = 0,1342 + 0,0100 → 1342 + 0100

La eventuale normalizzazione che va fatta dopo la somma consiste in un semplice aumento dell'esponente di uno. Infatti, se la somma 1cifre rappresenta 1,cifre, basta aumentare l'esponente di uno per ottenere 0,1cifre. Le cifre della mantissa sono quindi sempre 1cifre ossia quello che si otteneva nella somma fra interi.

Algoritmo di somma

Dovendo sommare a#b e c#d, i passi da fare sono:

1. se b<d, si converte a#b in 0…0a#d, dove gli zeri aggiunti sono d-b (se b>d, si fa lo stesso su c#d);
2. si aggiungono zeri in fondo a 0…0a e c in modo che abbiano la stessa lunghezza;
3. si sommano come interi: e = 0…0a0…0 + c0…0;
4. se la somma produce una cifra in più allora f=d+1, altrimenti f=d;
5. il risultato è e#f.

La sequenza di passi non ha richiede altro che confronto fra interi, aggiunte di zeri a sinistra e a destra, una somma fra interi e una eventuale addizione di uno

Algoritmo di somma - esempio

Per esempio, per sommare 994#3 con 7169#1 con quattro cifre mantissa e due di esponente, si procede nel modo seguente:

1. aumentare l'esponente minore: in questo caso l'esponente minore è quello del secondo numero, per cui 7169#1 si trasforma in 007169#3
2. uguagliare il numero di cifre delle mantisse: dei due numeri 994#3 e 007169#3, il secondo ha più cifre; si aggiungono zeri al primo, trasformando 994#3 in 994000#3
3. somma fra interi: è una normale somma:

```
 994000 +
 007169 =
----------
1001169
```

4. Normalizzare: è venuta una cifra in più, per cui occorre normalizzare; l'esponente maggiore dei due numeri, cioè tre, va aumentato di uno: l'esponente del risultato è quindi 3+1=4

limitare le cifre: il risultato della somma ha troppe cifre (sette invece di quattro); se ne prendono solo quattro, per cui la mantissa del risultato è 1001; Il risultato della somma è quindi 1001#4

Virgola mobile in base generica

Finora si sono usati numeri in base dieci. Lo stesso sistema di mantissa ed esponente si può realizzare anche in una base diversa da dieci, per esempio in binario. Si può procedere in due modi:

- si continua a usare la regola $0,\text{mantissa} \times 10^{\text{esponente}}$, ma ora sia la mantissa che l'esponente si rappresentano in binario
- si lavora nella base scelta, in questo caso il binario: sia l'esponente che la mantissa sono numeri binari, e il valore rappresentato è $0,\text{mantissa} \times 2^{\text{esponente}}$

In altre parole, il secondo sistema differisce perché è 2 che viene elevato all'esponente piuttosto che 10. Per convertire un numero da decimale a questa forma, occorre prima esprimerlo in binario (con i metodi visti in precedenza), poi verificare quanti spostamenti di virgola occorre effettuare per portarlo nella forma 0,cifre, con la prima cifra diversa da zero.

Formato IEEE 754

Un formato per numeri in virgola mobile usato in pratica è IEEE 754, che nel caso a 64 bit rappresenta i numeri in binario con:

un bit di segno
11 bit di esponente
52 bit di mantissa

Di norma le cifre della mantissa verrebbero interpretate come il numero 0,cifra. Dato però che la prima cifra si assume diversa da zero e che in binario l'unica cifra diversa da zero è uno, questo numero 0,cifra ha sempre la forma 0,1cifre_restanti. Diminuendo l'esponente di uno questo diventa 1,cifre_restanti. Nello standard IEEE 754, si memorizzano solo le cifre restanti, che sono i cinquantadue bit della mantissa.

Formato IEEE 754

In altre parole, i cinquantadue bit di mantissa si interpretano come le cifre binarie che seguono la virgola nel numero 1,cifre. Assumere che la prima cifra sia uno non è una limitazione, dato che sia per numeri maggiori di uno (es. 1100110) che minori (es. 0,00010111), la prima cifra diversa da zero non può che essere uno. Aumentando o diminuendo l'esponente, li si può portare nella forma voluta. Per esempio:

1100110 → 1,100110

0,00010111 → 1,0111

La mantissa è da considerarsi positiva, è il bit di segno a dire se il numero è positivo o negativo.

L'esponente è rappresentato in "eccesso 1023", cioè è aumentato di 1023. I valori 00000000000 e 11111111111 vengono usati per rappresentare condizioni particolari, come l'overflow o la non possibilità di una certa operazione (divisione per zero, radice di un numero negativo, ecc.)

Formato IEEE 754

Per esempio, il numero decimale 349,625 si rappresenta così:

1. si converte in binario con il solito sistema (parte intera con le divisioni successive e parte frazionaria con le moltiplicazioni successive) ottenendo 101011101,101
2. questo numero non è nella forma 1,cifre; occorre quindi spostare la virgola: $101011101{,}101 = 1{,}01011101101 \times 2^8$
3. la mantissa è 01011101101
4. per l'esponente, si memorizza 1023+8=1031 convertito in binario, ossia 10000000111

Il numero è quindi dato da: bit di segno 0, esponente 10000000111, mantissa 01011101101 seguiti da quarantuno zeri. Cioè

01000000011101011101101000000000000000000000000000000000000000

Insieme F dei numeri rappresentabili in virgola mobile

- sottoinsieme finito dei numeri razionali rappresentabili (con n bit)
- simmetrico rispetto allo 0
- gli elementi **non** sono uniformemente distribuiti sull'asse reale
  - densi intorno allo 0
  - radi intorno al massimo rappresentabile
- molti razionali non appartengono ad F (ed es. 1/3, 1/5, . . .)
- non è chiuso rispetto ad addizioni e moltiplicazioni
- per rappresentare un reale X si sceglie l'elemento di F più vicino ad X
- la funzione che associa ad un reale X l'elemento di F più vicino ad X è detta funzione di arrotondamento