Laurea in Ingegneria Gestionale

Corso di Fondamenti di Informatica
A.A. 2017/2018

DIPARTIMENTO DI INGEGNERIA INFORMATICA
AUTOMATICA E GESTIONALE ANTONIO RUBERTI

SAPIENZA
UNIVERSITÀ DI ROMA

Rappresentazione dell'Informazione - 2

Le seguenti slide sono rielaborate da slide analoghe del Prof. Diego Calvanese (Università di Bolzano)

L'aritmetica reale

Rappresentazione in virgola fissa

Di un numero reale assoluto si rappresentano separatamente, usando un numero fissato di cifre

- parte intera e
- parte frazionaria

(si usa una virgola per separare le due parti)

$$N_b = c_{n-1}\, c_{n-2}\, \cdots\, c_1\, c_0\, ,\, c_{-1}\, c_{-2}\, \cdots\, c_{-m}$$

rappresenta il numero

$$N = c_{n-1} \cdot b^{n-1} + \cdots + c_0 \cdot b^0 + c_{-1} \cdot b^{-1} + \cdots + c_{-m} \cdot b^{-m}$$

Conversione da base b a base 10

La conversione di un reale assoluto da base b in virgola fissa a base 10 si ottiene usando direttamente la formula precedente, esprimendo le cifre e facendo i conti in base 10.

Conversione di un numero reale assoluto da base 10 a base b in virgola fissa

Con un ragionamento analogo a quello fatto per i numeri naturali, si ottiene il seguente procedimento per la conversione della rappresentazione di un numero reale assoluto da base 10 a base b:

1. si converte la parte intera come fatto per i numeri naturali;
2. si moltiplica la parte frazionaria per b: la parte intera del prodotto ottenuto coincide con la cifra c_{-1}, mentre la parte frazionaria coincide con $c_{-2} \cdot b^{-2} + c_{-3} \cdot b^{-3} + \ldots$ Si itera il procedimento sulla parte frazionaria: Le parti intere dei prodotti via via calcolati costituiscono le cifre, dalla più significativa alla meno significativa, della rappresentazione in base b della parte frazionaria del numero dato. Il procedimento termina al raggiungimento del numero voluto di cifre per la rappresentazione della parte frazionaria (o se la parte frazionaria del prodotto è 0).

Esempio: La rappresentazione binaria in virgola fissa di $3,83$ con 4 cifre per la parte frazionaria è $11,1101$. La parte frazionaria è ottenuta prendendo la parte intera dei seguenti prodotti:

```
0,83 x 2 = 1,66
0,66 x 2 = 1,32
0,32 x 2 = 0,64
0,64 x 2 = 1,28
```

- Con la rappresentazione in virgola fissa un certo numero di cifre è dedicato alla parte intera, ed altre sono dedicate alla parte decimale....
- ....quindi, la rappresentazione in virgola fissa obbliga a "sprecare" un certo numero di bit per la parte decimale (intera) anche se il numero da rappresentare non ne avrebbe bisogno (ad ese. 40000.00).
- Analogamente, ci può essere uno spreco di bit anche nella parte intera (ad es. 00000.07)
- Intervallo di numeri rappresentabili piccolo per molte applicazioni: **in base** 2, considerando solo numeri positivi, con N bit per parte intera e K per parte frazionaria, il numero massimo rappresentabile è $2^N - 1 + (1/2 + 1/2^2 + ... + 1/2^K) = 2^N - 1/2^K$, il minimo numero positivo è $1/2^K$
- Bassa precisione: In base b, con K cifre utilizzate per la parte frazionaria, la precisione massima raggiungibile è $1/b^K$
- La notazione esponenziale o floating point (virgola mobile) riduce questi problemi. In particolare i bit vengono usati più efficacemente, per un intervallo di numeri più ampi

Rappresentazione in virgola mobile

Rappresentazione in **forma normalizzata in base** b (normalizzazione della notazione scientifica)

$$X = m \cdot b^e$$

- e è la **caratteristica** in base b di X: intero relativo
- m è la **mantissa** in base b di X: numero frazionario tale che $1/b \leq |m| < 1$

  Se è rappresentata dalla sequenza di cifre

  $$c_1\ c_2\ c_3\ \cdots$$

  allora rappresenta il valore

  $$c_1 \cdot b^{-1} + c_2 \cdot b^{-2} + \cdots$$

Si noti che c_1 è diversa da zero, dato che $1/b \leq |m|$

Dal numero reale in base 10 al numero in virgola mobile in base 2

In un calcolatore la base di rappresentazione è ovviamente 2, e si allocano k bit per la mantissa (e si rappresenta solo la parte a destra della virgola), h bit per la caratteristica (che può essere rappresentata in complemento), ed un bit per il segno (0 positivo, 1 negativo)

Esempio: Fissiamo $k = 4$ e $h = 3$, consideriamo il numero -0.3125 e rappresentiamolo in virgola mobile in base 2

(1) Dato che il numero è negativo, il bit del segno viene posto pari ad 1.

(2) Si converte il valore assoluto del numero reale in base 2: $(0.3125)_{10} = (0.0101)_2$

(3) i primi k bit a partire da quello più significativo (il primo 1 da sinistra) diventano la mantissa (si aggiungono zeri se necessario, ed eventuali cifre significative dopo i primi k bit vengono perse). Quindi la mantissa di 0.0101 è 1010

(4) Per la caratteristica si conta il numero di posizioni di cui è necessario spostare la virgola verso destra per arrivare a sinistra del primo 1. La caratteristica di 0.0101 è -1 perché devo spostare la virgola a destra di una posizione. In complemento a 2 su 3 bit, -1 vale 111.

(5) Quindi -0.3125 viene rappresentato come $1\ 111\ 1010$ (segno 1, caratteristica 111 e mantissa 1010).

Calcolo del numero in base 10 a partire dal numero in virgola mobile in base 2

Esempio: : Fissiamo $k = 4$ e $h = 3$ e consideriamo il numero in virgola mobile $1\ 010\ 1010$ (segno 1, caratteristica 010, mantissa 1010). A quale reale in base 10 corrisponde?

(1) Si converte la caratteristica in base 10: 010 vale $+2$;

(2) Si aggiunge $0.$ (0 virgola) prima della mantissa. Quindi 1010 diventa 0.1010;

(3) Si sposta la virgola di un numero di posizioni pari alla caratteristica verso destra se positivo, verso sinistra se negativo. Poichè la caratteristica è $+2$, 0.1010 diventa 10.10;

(4) Si converte il numero frazionario in base 10: 10.10 vale 2.5;

(5) Se il bit del segno è 1, si considera l'opposto: nel nostro esempio il risultato della conversione è -2.5.

In base 2, fissati

k bit per mantissa, h bit per caratteristica (in complemento), 1 bit per il segno

l'**insieme di reali rappresentabili** è fissato (e limitato)

$$1/2 \leq |m| \leq \sum_{i=1}^{k} 2^{-i} = 1 - 2^{-k}$$
$$e \in [-2^{h-1}, 2^{h-1})$$

Questo fissa anche massimo e minimo numero rappresentabile.

Consideriamo solo i **positivi** (la rappresentazione è simmetrica rispetto allo 0) e poniamo, ad esempio, $k = 5$ e $h = 3$. Abbiamo che:

Massimo rappresentabile:

$m = |m| = 1 - 2^{-5}$

$e = 2^2 - 1 = 3$

quindi $X_{max} = 2^3 - 2^{-2}$, rappresentato come $0\ 011\ 11111$ (nel solito ordine segno-caratteristica -mantissa)

Minimo rappresentabile:

$m = |m| = 1/2$

$e = -2^2 = -4$

quindi $X_{min} = 2^{-1} \cdot 2^{-4} = 2^{-5}$, rappresentato come $0\ 100\ 10000$

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. $1/3, 1/5, \ldots$)
- 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

Limitazioni aritmetiche

Dovute al fatto che il numero di bit usati per rappresentare un numero è limitato $\Longrightarrow$

- perdita di precisione
- **arrotondamento**: mantissa non è sufficiente a rappresentare tutte le cifre significative del numero
- **errore di overflow**: caratteristica non è sufficiente (il numero è troppo grande in valore assoluto); può essere positivo (risp. negativo) se il numero eccede il più grande positivo (risp. negativo) rappresentabile.
- **errore di underflow**: numero troppo piccolo in valore assoluto e viene rappresentato come 0 (come per l'overflow, può essere positivo o negativo).

Formati standard definiti dalla IEEE (Institute of Electrical and Electronics Engineers) nello standard 754. Secondo questo standard, la mantissa m è interpretata come $1.m$ e la catteristica è espressa con *eccesso* (una variante della rappresentazione in complemento)

- singola precisione: 32 **bit**, (1 bit per il segno, 8 bit per la caratteristica, 23 bit per la mantissa)
- doppia precisione: 64 **bit** (1 bit per il segno, 11 bit per la caratteristica, 52 bit per la mantissa)
- quadrupla precisione: 128 **bit** (1 bit per il segno, 15 bit per la caratteristica, 112 bit per la mantissa)

Rappresentazione in virgola mobile in Python

- Python supporta **solo la rappresentazione in doppia precisione** (mentre altri linguaggi, come il C, consentono esplicitamente al programmatore di selezionare il tipo di dato in virgola mobile che si desidera usare – in genere ci si "accontenta" della singola precisione per risparmiare memoria e tempo di calcolo, se non serve il livello di precisione offerto dai 64 bit della doppia precisione).
- maggiori dettagli sui floating points in Python sono al link `https://docs.python.org/3.8/tutorial/floatingpoint.html`

Esercizio: Cosa restituiscono i seguenti confronti in Pyhton $.1 + .1 + .1 == .3$ e $4.35 * 100 == 435$? Verificate usando la shell Python e analizzate il comportamento

Suggerimento: per stampare a schermo un numero in virgola mobile facendo visualizzare il numero voluto di cifre dopo la virgola, usate la funzione `format`. Ad esempio `format(0.1, '.50f')` stampa 0.1 con 50 cifre dopo la virgola.