Rappresentazione binaria dell'informazione

Lezione 5 di Fondamenti di informatica

Docente: Giuseppe Scollo

Università di Catania
Facoltà di Scienze Matematiche, Fisiche e Naturali
Corso di Laurea in Informatica, I livello, AA 2009-10

Indice

1. Rappresentazione binaria dell'informazione
2. rappresentazione binaria naturale
3. aritmetica in rappresentazione binaria
4. conversione di base per i naturali
5. rappresentazione binaria degli interi
6. rappresentazione in complemento a 2
7. conversione di base per gli interi
8. altre rappresentazioni binarie degli interi
9. rappresentazione binaria dei razionali
10. rappresentazione di testi, codici
11. rappresentazione di suoni
12. rappresentazione di immagini
13. temi per ulteriori approfondimenti

rappresentazione binaria naturale

rappresentazione **posizionale in base 2** (cioè: con 2 cifre) dei numeri naturali

significatività della posizione:
sequenza delle potenze intere della base, a partire da 0, crescente verso sinistra

introducendo la "**virgola binaria**", e le potenze ad esponente intero **negativo** della base per la significatività alla sua destra, si ottiene una rappresentazione binaria dei **numeri razionali** (non negativi)

rappresentazioni in una base che sia una potenza intera di 2 rendono più compatta la rappresentazione binaria, per la **semplicità della conversione di base** in tal caso, ad es.:

rappresentazione **ottale** ($b = 8 = 2^3$):
ogni cifra (0,...,7) rappresenta una sequenza di 3 bit

rappresentazione **esadecimale** ($b = 16 = 2^4$):
ogni cifra (0,...,9,A,B,...,F) rappresenta una sequenza di 4 bit

aritmetica in rappresentazione binaria

gli algoritmi per le operazioni aritmetiche nella rappresentazione posizionale decimale si generalizzano facilmente alla rappresentazione posizionale in una qualsiasi base $b > 1$

esempi:

ecco l'addizione di due numeri nella rappresentazione binaria naturale (in parentesi è indicato il riporto):

somma binaria naturale di 9 e 13

(1)			(1)		+
	1	0	0	1	
	1	1	0	1	
1	0	1	1	0	

la moltiplicazione binaria può risultare un po' scomoda, per la numerosità dei prodotti parziali da sommare (conviene scegliere il moltiplicatore col minor numero di 1):

prodotto binario di 13 e 9

			1	1	0	1	x
			1	0	0	1	
			1	1	0	1	
1	1	0	1				
1	1	1	0	1	0	1	

conversione di base per i naturali

come effettuare la conversione quando una delle basi *non* è una potenza intera dell'altra?
occorre distinguere due casi:

dalla base minore a quella maggiore:
si applica la **definizione** di rappresentazione posizionale, eseguendo **prodotti e somme** nella rappresentazione **in base maggiore**

esempio : $11010_2 = (0 \cdot 2^0 + 1 \cdot 2^1 + 0 \cdot 2^2 + 1 \cdot 2^3 + 1 \cdot 2^4)_{10} = (2+8+16)_{10} = 26_{10}$

dalla base maggiore a quella minore:
si itera la divisione, del numero dato e dei quozienti via via ottenuti, **per la base minore**, nella rappresentazione in base maggiore, fino a ottenere il quoziente zero
la **sequenza dei resti ottenuti**, rappresentati nella base minore, dà il risultato della conversione, con *significatività crescente* delle cifre nell'ordine in cui si ottengono

esempio : determiniamo la rappresentazione binaria naturale di 43_{10}
siano q_i, r_i rispettivamente i quozienti e i resti delle progressive divisioni per 2:

0	1	2	5	10	21	q_i
1	0	1	0	1	1	r_i

$43_{10} = 101011_2$

rappresentazione binaria degli interi

il segno di un numero intero può assumere due valori, dunque una plausibile rappresentazione binaria degli interi si ottiene adoperando

la rappresentazione binaria naturale per il **valore assoluto** del numero
un bit per la rappresentazione del **segno**

in tal modo con n+1 bit si possono rappresentare gli interi nell'intervallo
$[-(2^n-1), 2^n-1]$
tale rappresentazione ha due inconvenienti:

lo zero ha due distinte rappresentazioni
le operazioni di addizione e sottrazione hanno algoritmi distinti

soprattutto il secondo inconveniente risulta oneroso, perché complica la realizzazione dei circuiti aritmetici elementari dei calcolatori

la **rappresentazione in complemento a 2** risolve entrambi i problemi, ed è oggi largamente in uso

rappresentazione in complemento a 2

il bit più a sinistra rappresenta il **segno: 0 = +, 1 = -**
la rimanente sequenza di ***n*** bit rappresenta
- se il numero è **positivo o nullo:** il suo **valore assoluto** nella rappresentazione binaria naturale
- se il numero è **negativo: 1 +** il complemento della rappresentazione binaria naturale del valore assoluto (se $< 2^n$)

esempi:
- $25_{10} = (00011001)_{\overline{2}}$; $-25_{10} = (11100111)_{\overline{2}}$
- $0_{10} = (00000000)_{\overline{2}}$
- $1_{10} = (00000001)_{\overline{2}}$; $-1_{10} = (11111111)_{\overline{2}}$
- $127_{10} = (01111111)_{\overline{2}}$; $-127_{10} = (10000001)_{\overline{2}}$; $-128_{10} = (10000000)_{\overline{2}}$

con n+1 bit si possono così rappresentare gli interi nell'intervallo $[-2^n, 2^n-1]$
la **somma algebrica** si esegue come la somma naturale, bit di segno inclusi, e trascurando l'eventuale riporto oltre il bit di segno
esempio: $25_{10}-12_{10} = 25_{10} + -12_{10} = (00011001)_{\overline{2}} + (11110100)_{\overline{2}} = (00001101)_{\overline{2}} = 13_{10}$

conversione di base per gli interi

la conversione dalla rappresentazione binaria in complemento a 2 a quella decimale si esegue con essenzialmente **lo stesso algoritmo:**
- se il bit di segno è **0**, il numero è positivo, quindi si effettua la conversione naturale del valore assoluto
- se il bit di segno **1**, il numero è negativo, quindi
  - si complementa il resto della sequenza binaria
  - si aggiunge 1
  - si effettua la conversione naturale del valore assoluto così ottenuto

esempi:
- $(01011000)_{\overline{2}} = 88_{10}$
- $(11011000)_{\overline{2}} = -(0100111 + 1)_{\overline{2}} = -(0101000)_{\overline{2}} = -40_{10}$

altre rappresentazioni binarie degli interi

rappresentazione in **complemento a 1:**

- come quella in complemento a 2, eccetto per il "+1"
- non ha i vantaggi di quella in complemento a 2
- di interesse meramente storico (ad es., *mainframe* Univac degli anni '60-'70), praticamente abbandonata

rappresentazione in **eccesso 2^n** (con **n** bit per il valore assoluto):

- come quella in complemento a 2, eccetto che il valore del **bit di segno** è **opposto**
- ha gli stessi vantaggi di quella in complemento a 2
  - attenzione, però, al bit di segno nella **somma algebrica...**
  - che succede?
- perché è detta "in eccesso 2^n"...

rappresentazione binaria dei razionali

problema 1: la rappresentazione dei numeri razionali è finita, ma non limitata

- questo vale anche per i numeri interi...

problema 2: i numeri reali non hanno rappresentazione finita

- precisamente, ciò vale per i numeri irrazionali

è desiderabile, per la celerità di esecuzione delle operazioni aritmetiche in tali campi, disporre di una rappresentazione binaria di lunghezza fissa

- soluzione: rappresentazioni **approssimate**

in **virgola fissa** : sia la parte intera che quella frazionaria hanno lunghezze fisse

- precisione **assoluta** prefissata

in **virgola mobile** (ingl.: ***floating point***) : lunghezze fisse di **mantissa** intera ed **esponente** intero di una base b prefissata (di solito $b = 10$): $R = M \cdot b^E$

- precisione **relativa** prefissata
- spesso si rappresenta la mantissa in complemento a 2, e l'esponente in eccesso 2^n

rappresentazione di testi, codici

una semplice rappresentazione binaria di un testo consiste nella sequenza delle rappresentazioni binarie dei suoi caratteri, secondo un dato **codice di rappresentazione**

quest'ultimo è una funzione iniettiva che associa ad ogni carattere dell'alfabeto A del testo una sequenza di n bit, con $|A| \leq 2^n$

alfabeto: caratteri **stampabili** + caratteri **di controllo**

codice **ASCII (*American Standard Code for Information Interchange*)**: a 7 bit, + ottavo bit per varianti locali

la diffusione dell'informatica ha fatto emergere l'esigenza di codici più ricchi, per rappresentare testi in tutte le lingue (segni diacritici, alfabeti diversi da quello inglese, linguaggi di scrittura ideografica, etc.)

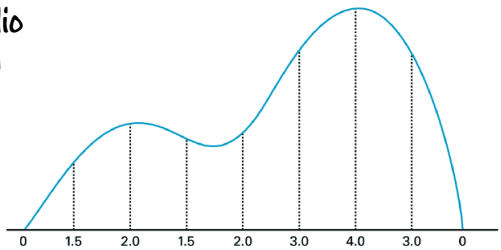
famiglia di codici **ISO 8859-x**: a 8 bit, estensioni del codice ASCII

codice **ISO/IEC 10646 (UCS: *Universal Character Set*)**: "universale", base per i codici (a 16 bit) **Unicode, UTF-8**

sono allo studio estensioni di Unicode (attualmente: a 20 bit)

rappresentazione di suoni

per la codifica digitale dell'informazione audio si effettua il **campionamento dell'ampiezza** dell'onda sonora ad intervalli regolari



la qualità della riproduzione dipende dalla **frequenza di campionamento**

8 kHz (ovvero un campione ogni ottavo di millesimo di secondo) è una frequenza adeguata alla riproduzione del parlato

per la riproduzione musicale di alta fedeltà la frequenza sale a **44.1 kHz**

il valore di ampiezza di ogni campione è codificato con **16 bit** (32 se stereo)

la codifica **MIDI (*Musical Instruments Digital Interface*)** è ben più parsimoniosa: rappresenta non il suono bensì l'informazione per la sua produzione da un **sintetizzatore**, ad es.: strumento emulato, altezza della nota, durata

rappresentazione di immagini

tecniche ***bitmap*** :

l'immagine è discretizzata in una **matrice di *pixel*** (***pic****ture's* ***el****ement*)
la matrice è sequenzializzata dal basso verso l'alto e da sinistra a destra
ciascun pixel è rappresentato da
1 bit per immagini in bianco e nero
una sequenza di bit per immagini a colori o con più livelli di grigio
la **codifica del colore**, basata sulla composizione cromatica di tre colori fondamentali, ad es. **RGB (Rosso, Verde, Blu)**, richiede **3*n*** bit per pixel, dove ***n*** è il numero di bit per la codifica del livello di ciascun colore fondamentale
la **risoluzione** indica la precisione della discretizzazione dell'immagine, ma è espressa da grandezze diverse in contesti diversi, ad es.:
dimensioni della matrice, in n. di pixel (ad es. 640X800)
densità dei pixel, ad es. pixel/mm, in ciascuna direzione

tecniche **vettoriali:**

molto più **efficienti** delle rappresentazioni bitmap, specialmente per alcune classi di immagini (diagrammi, glifi tipografici, etc.)
vantaggiose anche rispetto alla **variazione di scala** dell'immagine

temi per ulteriori approfondimenti

Rappresentazione di caratteri tipografici

I codici standard per la rappresentazione dei testi si limitano a fornire una codifica univoca ad ogni simbolo degli alfabeti considerati, e in generale prescindono dalla rappresentazione grafica dei simboli quali caratteri tipografici. Questo aspetto attiene alla rappresentazione di una particolare classe di immagini, i **glifi** (ingl. ***font***) tipografici, frequentemente disponibili sia in forma *bitmap* che vettoriale. Quest'ultima ben si presta alla variazione di scala dei glifi. Ulteriori approfondimenti del tema possono riguardare il rapporto fra rappresentazione dei glifi e caratteristiche (quali, ad es., risoluzione, resa dei colori, etc.) dei dispositivi di visualizzazione e/o stampa dei testi.