Algoritmi

Laboratorio del Piano Lauree Scientifiche – I.I.S. Gaetano De Sanctis (10.04.2017)

Stefano Finzi Vita
Dipartimento di Matematica - Sapienza Università di Roma

Avete ormai visto diversi esempi di algoritmi. Oggi vogliamo tornare sulla rappresentazione dei numeri, anche in basi diverse da 10 (ad esempio 2), e sulla loro approssimazione (come avviene quando li utilizziamo in un computer). L'insieme dei *numeri macchina*, cioè i numeri effettivamente rappresentabili, è profondamente diverso da quello del continuo dei reali, con conseguenze importanti (identità che diventano false, proprietà che non valgono più), operazioni 'pericolose' e possibili effetti disastrosi. Ci occorrono quindi procedure efficienti e ben strutturate, operazioni semplici ripetute magari migliaia di volte (*cicli*) e diverse strade da percorrere in base alle situazioni (*alternative*) [da tradurre poi in un opportuno *linguaggio di programmazione*]. Tra queste procedure hanno una notevole importanza quelle ricorsive, cioè quelle che richiedono un calcolo ripetuto a partire da uno o più dati iniziali. Vogliamo vederne qualche esempio.

1 La rappresentazione dei numeri: il caso continuo

Ogni numero reale può essere rappresentato come un numero decimale illimitato:

$$x = n.a_1a_2\dots a_k\cdots = n + \frac{a_1}{10} + \frac{a_2}{10^2} + \cdots + \frac{a_k}{10^k} + \dots$$

dove $n \in \mathbb{Z},\ a_i = 0, 1, \dots, 9$. In particolare:

- periodici → razionali : $2(.00000...),\ 3.5(00000...),\ 0.\overline{6},\ 7.02\overline{428}$
- non periodici → irrazionali : $\sqrt{2},\ \sqrt{3},\ \pi,\ e$

Ogni numero reale x può essere approssimato (con precisione arbitraria) mediante un numero razionale $\overline{x}$ [**densità di $\mathbb{Q}$ in $\mathbb{R}$**] e la retta reale *"non ha buchi"* [**continuità dei numeri reali**]. Definiremo:

- **errore assoluto**: $e_A(x) = |x - \overline{x}|$, **errore relativo**: $e_R(x) = \frac{|x-\overline{x}|}{|x|} = \frac{e_A}{|x|}$.
- **troncamento** (arrotondamento per difetto) di x a k cifre: $x_{(k)} = n.a_1a_2\dots a_k$
- **arrotondamento per eccesso** di x a k cifre: $x^{(k)} = n.a_1a_2\dots(a_k + 1) = x_{(k)} + 10^{-k}$
- **arrotondamento di x alla k-ma cifra** : $fl_k(x) = x_{(k)}$ se $a_{k+1} < 5$; $fl_k(x) = x^{(k)}$ se $a_{k+1} \geq 5$
- $x_{(k)} \leq x < x^{(k)},\ |x - x_{(k)}| < 10^{-k},\ |x - x^{(k)}| < 10^{-k},\ |x - fl_k(x)| < 0.5 * 10^{-k}$.

Per operare con i numeri decimali non nulli è spesso comodo rappresentarli in una forma normalizzata che ne mette in risalto l'ordine di grandezza, la **rappresentazione in virgola mobile**:

$$x = \pm m \cdot 10^z = (\pm m, z), \quad 0.1 \leq m < 1, \quad z \in \mathbb{Z}$$

con m = **mantissa**, z = **esponente**. La corrispondenza $x \to (m, z)$ è unica (*). Esempi:

$$9 \to (0.9, 1), \quad -123.81471 \to (-0.12381471, 3), \quad 0.0000715 \to (0.715, -4).$$

(*) se escludiamo i numeri decimali con periodo 9 (per esempio $0.\overline{9} = 1$), che d'altra parte non si ottengono mai dalla divisione di due interi.

2 Sui cambiamenti di base

La scelta di usare la base 10 risponde all'esigenza di facilitare i calcoli, ma ovviamente non è l'unica possibile. Ad esempio si è scoperto che gli Inca usavano per i calcoli un abaco (la *Yupana*) basato sulla notazione posizionale in base 40. L'aritmetica dell'orologio (ore, minuti, secondi) è in base 60. E i computer lavorano in base 2. E' quindi utile poter passare da una base all'altra, convertendo la rappresentazione dei numeri.

Attenzione:

- un numero decimale finito in una base può restare finito o diventare periodico in un'altra
- un numero decimale periodico in una base può restare periodico o diventare finito in un'altra
- un numero decimale illimitato non periodico lo rimane in ogni base

Per capire se un numero razionale a in base 10 sia finito o meno in un'altra base b:

- Si calcola la sua frazione generatrice: $a = \left(\frac{n}{m}\right)_{10}$.
- Se m divide una qualche potenza di b (b^k multiplo di m per qualche $k \in \mathbb{N}$), il numero sarà finito in base b (con k cifre per la parte frazionaria), altrimenti no.

Esempi:
$0.75 = \frac{75}{100} = \frac{3}{4} \Rightarrow 4$ divide (coincide con) $2^2 \Rightarrow (0.75)_{10} = (0.11)_2$;

$2.7 = \frac{27}{10} = \frac{27}{2\times5} \Rightarrow$ in base due il numero diventerà periodico (nessuna potenza di 2 è multipla di 10).

Ci serve un algoritmo generale per il cambiamento di base, dalla base b alla base 10 e viceversa. In base alla rappresentazione posizionale non è difficile costruire gli algoritmi per il passaggio da una base a un'altra.

- **Dalla base b alla base** 10
  Se il numero in base b ha rappresentazione: $a_n a_{n-1}...a_1 a_0.a_{-1}...a_{-m}$, in base 10 sarà dato da:
  $$a_n b^n + a_{n-1} b^{n-1} + ... + a_1 b + a_0 + a_{-1} b^{-1} + ... + a_{-m} b^{-m}$$
- **Dalla base** 10 **alla base b**
  - **parte intera**: si divide per b finché il risultato non viene zero; i resti delle divisioni in ordine inverso danno la nuova p.i.
  - **parte frazionaria**: si moltiplica per b finché il risultato non ha parte frazionaria nulla o si riottiene una parte frazionaria già incontrata; le parti intere dei prodotti nell'ordine trovato danno la nuova p.f.

Esercizio.

- Se approssimiamo 9999 con 10000 e 0.99 con 1, in quale dei due casi l'approssimazione è più accurata?
- In quale base il numero periodico $0.\overline{3}_{10}$ diventa un numero con parte frazionaria finita e quale ne sarebbe la rappresentazione?
- In quali basi b (tra 2 e 9) il numero decimale 0.375 avrebbe rappresentazione finita?
- Trasformate in base 10 i numeri $(1101.01)_2$ e $(210.012)_3$.
- Trasformate il numero $(25.7)_{10}$ in base 2.

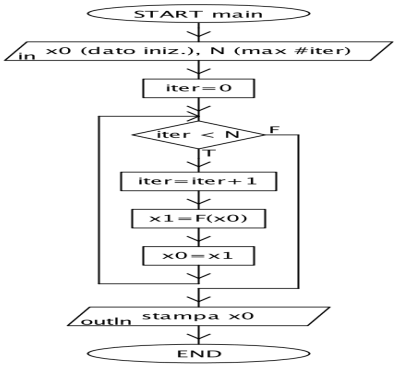
3 Algoritmi ricorsivi

Chiamiamo **metodo iterativo a un passo** quello definito a partire da un **dato iniziale** x_0 e da una **legge** F che trasforma (attraverso una formula o dei calcoli ben definiti) un dato in input in un dato in output, legge da applicare ripetute volte a partire da x_0. In formule:

$$\text{dato } x_0, \quad \text{calcolare} \quad x_{n+1} = F(x_n) \quad \text{per } n = 0, 1, 2, ...$$

finché un certo **criterio di arresto** non viene soddisfatto: ad esempio si può richiedere un numero fisso di iterazioni da raggiungere oppure di fermarsi appena il valore delle iterate soddisfa una data condizione. In figura un possibile diagramma di flusso con arresto basato sul numero massimo di iterazioni.

Definizione. Chiamiamo valore di **equilibrio** (o **punto fisso**) della legge F un valore c tale che $F(c) = c$.



Mettetevi alla prova con i seguenti esercizi.

1. Data la legge $F(x) = 2x - 1$ calcolarne il valore di equilibrio. Scrivere le prime 5 iterazioni partendo da $x_0 = 0$ e da $x_0 = 2$. Come si comportano le sequenze rispetto all'equilibrio?
2. Scrivere le prime 6 iterazioni delle leggi $F_1(x) = x/2$ e $F_2(x) = -x/2$ partendo da $x_0 = 1$. Come si comportano le sequenze rispetto al valore di equilibrio?
3. Date le sequenze di valori seguenti, dedurre le leggi che le hanno rispettivamente generate:
   2 5 8 11 14 ...
   0 1 2 5 26 677 ...
   8 7 10 9 12 11 ...
4. Data la legge $F(x) = 3x + 4$, se $x_3 = 52$, quale è stato il dato iniziale x_0 da cui si è partiti?

4 L'algoritmo di Erone per la radice quadrata

Consideriamo un esempio celebre di algoritmo ricorsivo: una procedura molto efficiente per il calcolo di $\sqrt{2}$, attribuita ad Erone di Alessandria (I sec. d.C.). L'idea è quella di generare una sequenza di valori che si avvicinino sempre più al valore cercato, che data la sua irrazionalità potrà solo essere approssimato con la precisione desiderata.

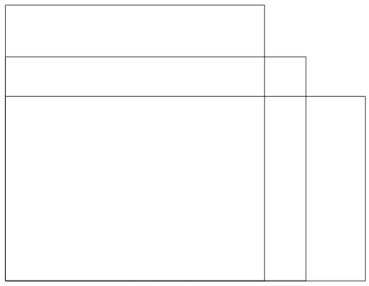
Idea geometrica: $\sqrt{2}$ è la misura del lato del quadrato di area 2. Partiamo allora da un rettangolo di dimensioni $x_0 > \sqrt{2}$ e $2/x_0$ (quindi di area 2) e cerchiamo di generare una sequenza di rettangoli di dimensioni via via più vicine tra loro, che si avvicini quindi sempre più al quadrato cercato.
Osserviamo che la media aritmetica tra i valori x_0 e $2/x_0$ sarà compresa tra di essi. Poniamo quindi:

$$x_1 = \frac{1}{2}(x_0 + \frac{2}{x_0})$$

Ad es. se $x_0 = 2$, allora $x_1 = 3/2 = 1.5$, e passeremo quindi da un rettangolo di dimensioni 2×1 a un rettangolo di dimensioni $3/2 \times 4/3$. Ripetendo il procedimento:

$$x_2 = \frac{1}{2}(x_1 + \frac{2}{x_1}) = \frac{1}{2}(\frac{3}{2} + \frac{4}{3}) = \frac{17}{12} = 1.4166666...$$

In figura il grafico delle prime due iterazioni. L'algoritmo di Erone per il calcolo di $\sqrt{2}$ è dunque il seguente:



$$\textbf{Assegnato } \mathbf{x_0} > \sqrt{2}, \ \textbf{calcola} \quad \mathbf{x_{n+1}} = \mathbf{F(x_n)} = \frac{\mathbf{1}}{\mathbf{2}}(\mathbf{x_n} + \frac{\mathbf{2}}{\mathbf{x_n}}) \textbf{ per n} = \mathbf{0, 1, 2}, ...$$

Partendo per esempio da $x_0 = 2$, i primi 5 valori saranno:
$x_1 = 1.5$, $x_2 = 1.416666666666667$, $x_3 = 1.414215686274510$, $x_4 = 1.414213562374690$, $x_5 = 1.414213562373095$
Con sole 5 iterazioni otteniamo una precisione di oltre 10^{-10}!

Idea algebrica: partiamo dall'equazione $x^2 = 2$; allora: $x = \frac{2}{x} \quad \rightarrow \quad 2x = x + \frac{2}{x} \quad \rightarrow \quad x = \frac{1}{2}(x + \frac{2}{x})$
L'algoritmo di Erone si ottiene trasformando l'ultima equazione (soddisfatta da $\sqrt{2}$) in un metodo iterativo innescato da un valore iniziale. Funziona sempre? Che succede se ci fermassimo al primo passo prendendo il metodo : $x_{n+1} = \frac{2}{x_n}$?
Proviamo a dimostrare che Erone funziona! Ci serviranno semplici disuguaglianze algebriche.

Esercizio.

- Calcolare il valore di equilibrio della legge F;
- mostrare che se $x_0 > \sqrt{2}$ vale sempre $x_n > 0$;
- provare che vale sempre $x_n^2 > 2$ (cioè i valori saranno sempre a destra di $\sqrt{2}$) ;
- provare che vale sempre $x_n > x_{n+1}$ (cioè la sequenza decresce);
- concludere che la sequenza si avvicina decrescendo proprio a $\sqrt{2}$.

In realtà si dimostra di più ! Per stimare la bontà dell'algoritmo possiamo infatti valutare la sua efficienza (o *velocità di convergenza*), e vorremmo anche un buon *criterio di arresto* (stop con la certezza di aver raggiunto la precisione desiderata). Vale

$$x_n - \sqrt{2} < \frac{x_0 - \sqrt{2}}{2^n} \ .$$

Infatti per ogni n si ha $x_n > \sqrt{2}$, quindi $2/x_n < \sqrt{2}$. Allora

$$x_{n+1} - \sqrt{2} = \frac{1}{2}\left(x_n + \frac{2}{x_n}\right) - \sqrt{2} < \frac{1}{2}(x_n + \sqrt{2}) - \sqrt{2} < (x_n - \sqrt{2})/2;$$

quindi la differenza tra x_n e il valore $\sqrt{2}$ tende a zero al crescere di n, e l'errore si dimezza almeno ad ogni passo. In realtà si può addirittura dimostrare che l'errore si riduce come il quadrato dell'errore precedente. In altre parole il numero delle cifre significative corrette di $\sqrt{2}$ si raddoppia all'incirca ad ogni iterazione! Abbiamo inoltre un utile criterio d'arresto:

$$x_{n+1} - \sqrt{2} < x_n - x_{n+1} \ .$$

Infatti dalla relazione vista al punto precedente: $2x_{n+1} < x_n + \sqrt{2}$, da cui, sottraendo a entrambi i membri $(x_{n+1} + \sqrt{2})$ si ottiene la tesi, che afferma in sostanza che se mi fermo dopo $n + 1$ passi, la mia distanza dal valore corretto di $\sqrt{2}$ è minore della differenza $x_n - x_{n+1}$. Ci riterremo soddisfatti, e quindi fermeremo le iterazioni, non appena questa differenza sarà diventata più piccola della precisione voluta. Ad esempio se vogliamo k cifre corrette basterà fermarsi appena

$$x_n - x_{n+1} < \frac{1}{2} \times 10^{-k}$$

Quanto visto per $\sqrt{2}$ si generalizza facilmente:

- L'algoritmo per approssimare la radice di un qualunque numero positivo α sarà dato da

$$x_{n+1} = F(x_n) = \frac{1}{2}(x_n + \frac{\alpha}{x_n}) \ \text{ per } n = 0, 1, 2, ...$$

- Possiamo passare alla radice cubica di un numero positivo estendendo l'idea geometrica: vogliamo trasformare un parallelepipedo di base quadrata (di lato x_0) e altezza α/x_0^2 (quindi di volume pari ad α) via via verso un cubo il cui lato sarà pertanto il numero cercato. Usando ancora la media aritmetica delle dimensioni, si ottiene la formula:

$$x_{n+1} = F(x_n) = \frac{1}{3}(2x_n + \frac{\alpha}{x_n^2}) \ \text{ per } n = 0, 1, 2, ...$$

- Quale sarebbe un algoritmo per calcolare $\sqrt[5]{\alpha}$?