Il π e il metodo Monte Carlo
Una piccola avventura matematica in C

Tempo fa, tra svariate ricerche sulla rete, mi è capitato di ritornare su di un argomento che in passato mi aveva particolarmente incuriosito: il metodo Monte Carlo.
Sono conscio del fatto che non sia molto professionale farlo, ma per avere una prima definizione di metodo Monte Carlo riporto quanto descritto da Wikipedia:

"L'algoritmo Monte Carlo è un metodo numerico che viene utilizzato per trovare le soluzioni di problemi matematici[…], per esempio il calcolo integrale."

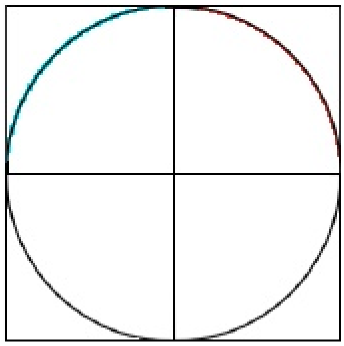
Incuriosito da tutto ciò ho provato a cimentarmi nella sua applicazione per ricavare qualche valore noto magari utilizzando un linguaggio di programmazione da me conosciuto.
Effettuando ulteriori ricerche sulla rete mi sono accorto che era possibile utilizzare il metodo per il calcolo del pigreco.
Una costante irrazionale, ricordiamo, che possiede infiniti numeri dopo la virgola, ma allo stesso tempo così tanto utilizzata nel calcolo di tutto ciò che concerne circonferenze, cerchi, sfere e affini.

Ebbene sì, il pigreco può essere calcolato mediante questa tecnica così particolare e pratica (come nel caso che vedremo) allo stesso tempo.

Ottenere una buona approssimazione del pigreco è proprio lo scopo che mi sono prefissato di raggiungere utilizzando questo metodo nella stesura dell'articolo.

Noterete quanto questo metodo abbraccia anche un altro settore della matematica: la statistica.

Partiamo prendendo due semplici figure: un quadrato di lato l e un cerchio con il diametro della stessa dimensione; disegniamoli come in figura, il cerchio inscritto nel quadrato:



Le aree delle due figure si possono facilmente calcolare:

Area del cerchio (chiamiamo con r il raggio del cerchio =(l/2)):

pigreco * r^2

Area del quadrato:

l^2=4 r^2

Ora calcoliamo il rapporto tra l'espressione dell'area del cerchio e quella del quadrato e procediamo con le dovute riduzioni:

(pigreco * r^2)/(4r^2)=pigreco/4=C

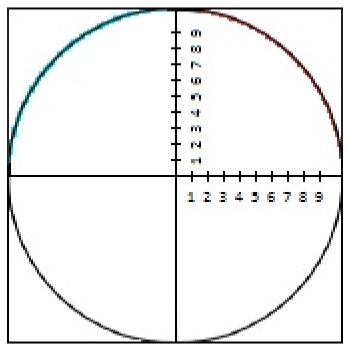
Chiamiamo il rapporto finale C. Quindi la nostra costante assume valore:

pigreco=4 C

Una volta conosciuto il valore di C si può pervenire al valore pigreco. Non ci resta che calcolare C.

Ora, immaginiamo di prendere della sabbia (granelli fittissimi) e di farla cadere in modo del tutto casuale sul nostro disegno: in questo modo avremo che una certa quantità di granelli si saranno depositati solo sulla superficie del cerchio e una parte sulla superficie del quadrato. Il numero è così fitto che possiamo effettuare il rapporto della quantità di granelli finiti nel cerchio e il numero dei granelli finiti nel quadrato e approssimarlo a quello delle due aree (come visto in precendenza).

Ma ora passiamo al calcolo in maniera pratica e cioè… codifichiamo il tutto in linguaggio C… non vi preoccupate… è più facile di quanto possa sembrare: occorre partire immaginando che i due diametri disegnati nel cerchio siano due assi cartesiani, di lunghezza, ad esempio, pari a 10 (non specifichiamo l'unità di misura, l'importante è che le suddivisioni siano di pari lunghezza):



www.webalice.it/lucianoporta/LEZIONIUNO/MONTECARLO.pdf

http://it.wikipedia.org/wiki/Metodo_Monte_Carlo

```
#include <stdio.h>
#include <stdlib.h>
#include <math.h>
#define PI 3.141592

int main() {

       double r, x, y, c = 0;
       int n, m, i, prove = 10000000;

       srand(time(0));

       for (i = 0; i < prove; i++)
       {
              n = rand();
              m = rand();

              x = ((double)n)/RAND_MAX*10;
              y = ((double)m)/RAND_MAX*10;
              r = sqrt(x*x + y*y);

              if (r < 10) c++;
       }

       printf("\nPi greco reale     --> %f\n", PI);
       printf("Pi greco calcolato --> %f\n", c /prove*4);
       printf("Scarto assoluto    --> %f \n", (PI - c/prove*4));

}
```

Per poter simulare i granelli immaginiamo di generare una oppia di numeri random da 0 a 10 in questo modo la coppia ci indica la sua posizione nel grafico. Siccome stiamo considerando solo numeri positivi, ci occuperemo solo del primo quarto di area (vedi figura). Ovviamente tutti i punti generati sono all'interno della superficie del quarto di quadrato, mentre per individuare i punti situati all'interno del quarto di cerchio utilizzeremo la formula relativa al calcolo della distanza del punto dato rispetto all'origine degli assi e controllando che tale distanza sia inferiore o uguale alla lunghezza del raggio del cerchio.

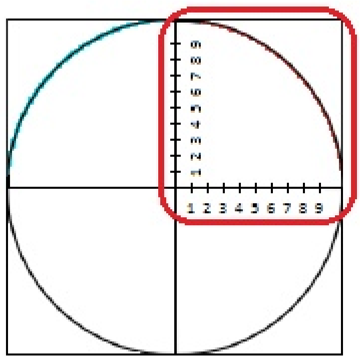
Chiamando con x e y i valori delle coordinate dei punti questi ultimi si troveranno all'interno del quarto di cerchio se viene rispettata questa condizione:

$$\sqrt{(x^2+y^2)} \le 10$$

Dal codice si evince l'utilizzo della funzione srand e della costante RAND_MAX per generare numeri quanto più casuali possibili.

Con un numero di "granelli"

(prove) pari a 10000000 si sono ottenuti i seguenti risultati:




```
Pi greco reale         --> 3.141592
Pi greco calcolato   --> 3.141074
Scarto assoluto        --> 0.000518
```