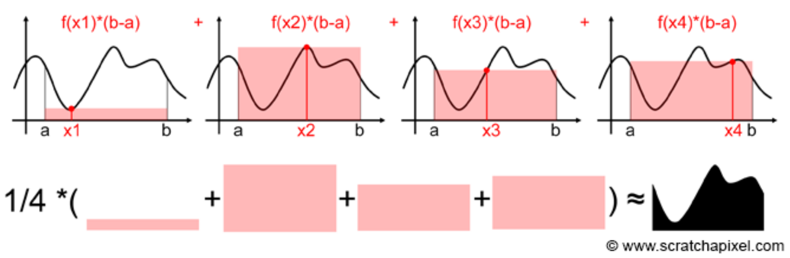
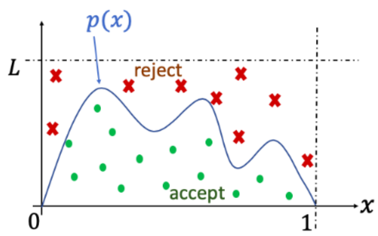
Metoda Monte Carlo. Alg. de aproximare a lui Pi.

1. Metoda Monte Carlo

Metodele Monte Carlo sunt algoritmi computaționali care se bazează pe eșantionarea aleatorie pentru a obține rezultate numerice. E vorba de tehnici matematice care permit cercetătorilor să facă estimări într-o lume probabilistică. Această tehnică, inițial dezvoltată pentru studiul fisiunii nucleare după Al Doilea Război Mondial, implică simulări statistice pentru modelarea sistemelor probabilistice și determinarea probabilităților diferitelor rezultate. Termenul "Monte Carlo" a fost ales de matematicianul Stanislaw Ulam, inspirat de unchiul său, pasionat de jocurile de noroc la cazinoul din Monte Carlo. Astăzi există o diversitate de simulări Monte Carlo, utilizate în diverse domenii, de la fizica particulelor până la inginerie, finanțe şi multe altele.





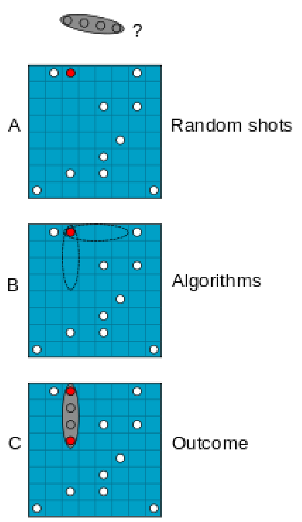
Nu există o singură metodă Monte Carlo; termenul descrie o clasă de abordări utilizate pentru o categorie mare de probleme. Cu toate acestea, aceste abordări tind să urmeze un anumit tipar:

i. Definiriea unui domeniu de date de intrare posibile.

ii. Generarea de intrări aleatorii cu o anumită distribuție de probabilitate.

iii. Efectuarea unui calcul determinist folosind datele de intrare.

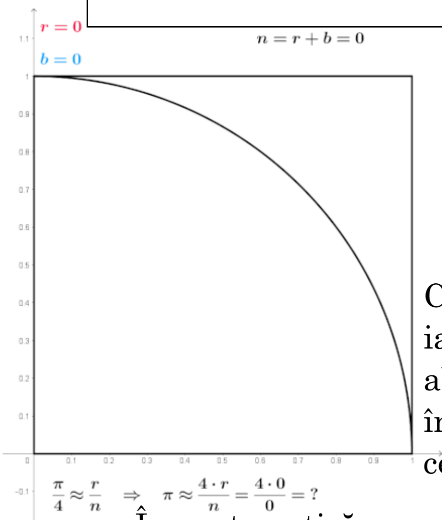
iv. Agregarea rezultatele calculelor individuale în rezultatul final.



Metoda Monte Carlo poate fi ilustrată ca o bătălie navală . Primul jucător trage lovituri aleatorii. Apoi, el aplică niște algoritmi (de exemplu, cuirasatul are patru puncte în direcția verticală sau orizontală). În cele din urmă, pe baza rezultatelor eșantionării aleatorii și a algoritmilor, jucătorul poate determina pozițiile probabile ale navelor celorlalți jucători.

Metoda este utilizată pe scară largă în diverse discipline:

Fizică și Inginerie Nucleară:	Simularea reactorului nuclear și evaluarea parametrilor critici de siguranță.
Finanțe și Economie:	Evaluarea opțiunilor financiare și modelarea riscului în piețele financiare.
Inginerie Structurală:	Analiza fiabilității structurilor sub diferite condiții de încărcare
Medicină și Biologie:	Simularea interacțiunilor moleculelor și a proceselor biologice.
Inginerie Electrică și Electronică:	Evaluarea performanței și fiabilității circuitelor electronice.
Științe Sociale și Economie:	Modelarea comportamentului social și economic și analiza politicii publice
Meteorologie și Climatologie:	Prognozarea vremii și înțelegerea schimbărilor climatice.
Jocul Go și șah:	Simularea variantelor posibile și calcularea probabilităților de câștig sau pierdere.
IT:	Simulări și modelare, analiză de risc și siguranță optimizare și învățare automată, criptografie și securitate:



2. Aproximarea lui Pi

Calculul valorii lui Pi este o problemă clasică în matematică, iar metoda Monte Carlo oferă o modalitate elegantă de a o aborda. Prin generarea unui număr mare de puncte aleatoare într-un pătrat și numărarea celor care cad în interiorul unui cerc inscris, putem obține o estimare numerică a lui Pi.

În matematică, π este un număr irațional, iar pentru majoritatea aplicațiilor practice, aproximarea sa la 152 de zecimale este suficientă.(De ce? De exemplu dacă vrem să calculăm circumferința unei sfere și diametrul ar fi de 93 de miliarde de ani-lumină (adica diametrul universului observabil), o aproximare la 152 de zecimale a lui π ar fi suficientă pentru a obține o eroare cel mult egală cu lungimea Planck, cea mai mică unitate de măsură pentru lungime care are vreo semnificație.)

Informaticienii dezvoltă algoritmi pentru aproximarea lui π la miliarde sau chiar trilioane de zecimale. De exemplu, în ianuarie 2020, un super-calculator a stabilit un nou record, determinând primele 50 de trilioane de zecimale ale lui π, după 300 și ceva de zile de lucru intens, depășind recordul anterior de 31.4 trilioane, stabilit de Google cu un an înainte. Deci este o metodă foarte bună de a pune la încercare puterea computațională a celor mai noi super-calculatoare

- Metoda seriilor(suma inverselor pătratelor perfecte nenule – the Basel problem)

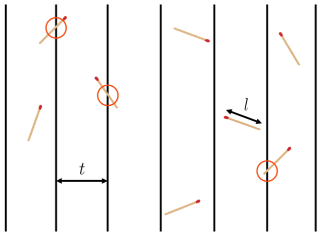
$$\sum_{n=1}^{\infty} \frac{1}{n^2} = 1 + \frac{1}{4} + \frac{1}{9} + \frac{1}{25} + \cdots = \frac{\pi^2}{6}$$

$$\frac{2\sqrt{2}}{9801} \sum_{k=0}^{\infty} \frac{(4k)!(1103+26390k)}{(k!)^4 396^{4k}} = \frac{1}{\pi}$$

600 termeni – 3 zecimale

2 termeni – 14 zecimale

- Metoda Buffon's Needle



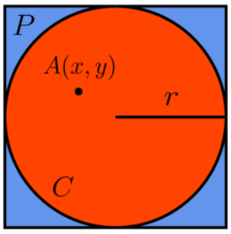
chibrirt = latime / 2,

raportul: intersecteaza / chibrituri = 1 / π

- Metoda MC

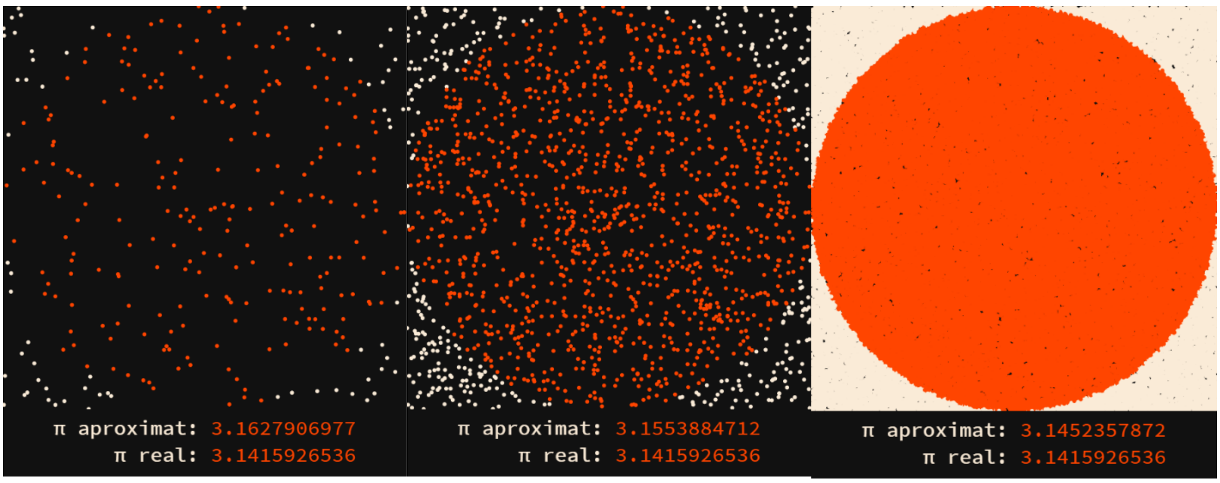
În informatică, un algoritm randomizat se numește Monte Carlo dacă de obicei produce un rezultat foarte apropiat de cel corect, sau Las Vegas dacă returnează mereu răspunsul exact. În continuare, voi prezenta un algoritm Monte Carlo foarte simplu, ce poate fi folosit pentru a calcula primele două-trei zecimale ale lui π.

Fie un pătrat oarecare P, de latură $2r$, și fie C cercul înscris în pătratul respectiv. Algoritmul se bazează pe ideea că, dacă alegem un punct aletoriu A din interiorul lui P, probabilitatea ca acesta să se afle și în interiorul lui C este egală cu π / 4.



De ce? Calculând ariile celor două figuri: aria cercului este $ArieC = \pi r^2$, iar aria pătratului este $ArieP = 4r^2$și raportul dintre cele două arii este π / 4. Deci raportul este chiar probabilitatea căutată de noi.

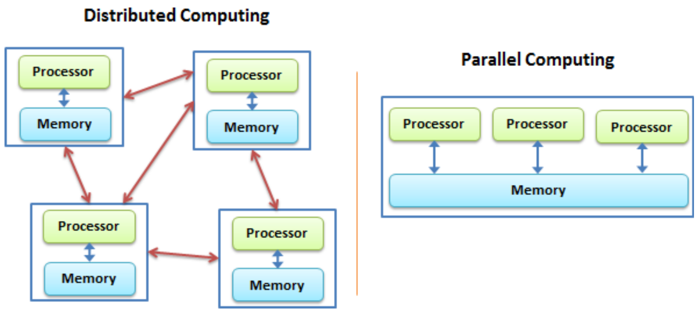
Algoritmul constă în a genera cât mai multe puncte random în interiorul unui pătrat și a determina câte dintre acestea se află la o distanță cel mult egală cu r de centrul pătratului. Apoi, împărțim acest număr la numărul total de puncte, înmulțim rezultatul cu 4 și putem spune că l-am aproximat pe π.

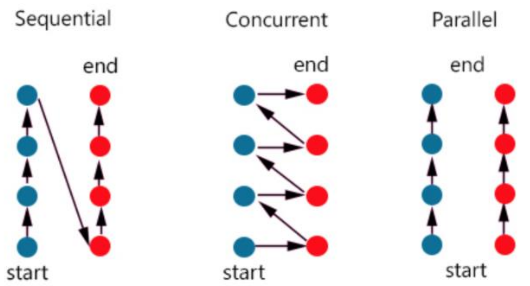


3. Paralelizare

Pentru a accelera procesul de aproximare a lui π, se pot distribui sarcinile de calcul către mai multe fire de execuție. Câteva idei:

- Utilizarea programării paralele pentru a împărți generarea de puncte aleatorii între mai multe fire de execuție.
- Implementarea unei abordări de paralelizare bazate pe divizarea domeniului de calcul, cum ar fi împărțirea pătratului în regiuni și distribuirea acestora către firele de execuție.
- Utilizarea unor tehnici de sincronizare și comunicare între firele de execuție pentru a asigura coerența și corectitudinea rezultatelor.
- Evaluarea performanței și scalabilității algoritmului paralelizat, luând în considerare numărul de nuclee de procesor disponibile și caracteristicile hardware-ului.





4. Aplicație C# - WFA

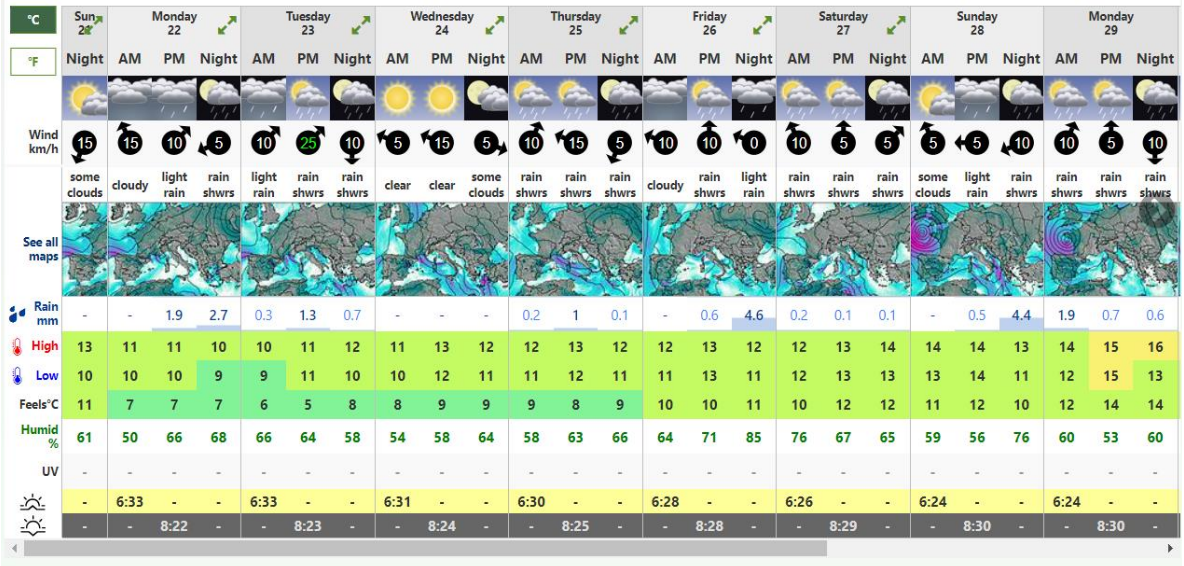
```
1 reference
private void timer1_Tick(object sender, EventArgs e)
{
    Draw();
    pictureBox1.Image = bmp;
}

1 reference
private void Draw()
{
    total++;
    int x = rnd.Next(0, 2 * radius);
    int y = rnd.Next(0, 2 * radius);

    double sqrDist = Math.Pow(radius - x, 2) + Math.Pow(radius - y, 2);
    if (sqrDist <= Math.Pow(radius, 2))
    {
        inside++;
        grp.FillEllipse(Brushes.Red, x - 3, y - 3, 7, 7);
    }
    else
        grp.FillEllipse(Brushes.White, x - 3, y - 3, 7, 7);
    textBox1.Text = $"π aprox: {(double)4 * inside / total:F15}";
    pictureBox1.Image = bmp;
}

1 reference
private void buttonRefresh_Click(object sender, EventArgs e)
{
    inside = 0;
    total = 0;
    grp.Clear(Color.Black);
    pictureBox1.Image = bmp;
}
No issues found
```

5. Prognoza meteo – metoda MC



	Sun 2[illegible]	Monday 22			Tuesday 23			Wednesday 24			Thursday 25		
°C / °F	Night	AM	PM	Night	AM	PM	Night	AM	PM	Night	AM	PM	Night
Wind km/h	15	15	10	5	10	25	10	5	15	5	10	15	5
	some clouds	cloudy	light rain	rain shwrs	light rain	rain shwrs	rain shwrs	clear	clear	some clouds	rain shwrs	rain shwrs	rain shwrs
Rain mm	-	-	1.9	2.7	0.3	1.3	0.7	-	-	-	0.2	1	0.1
High	13	11	11	10	10	11	12	11	13	12	12	13	12
Low	10	10	10	9	9	11	10	10	12	11	11	12	11
Feels°C	11	7	7	7	6	5	8	8	9	9	9	8	9
Humid %	61	50	66	68	66	64	58	54	58	64	58	63	66
UV	-	-	-	-	-	-	-	-	-	-	-	-	-
Sunrise	-	6:33	-	-	6:33	-	-	6:31	-	-	6:30	-	-
Sunset	-	-	8:22	-	-	8:23	-	-	8:24	-	-	8:25	-

	Friday 26			Saturday 27			Sunday 28			Monday 29		
°C / °F	AM	PM	Night	AM	PM	Night	AM	PM	Night	AM	PM	Night
Wind km/h	10	10	0	10	5	5	5	5	10	10	5	10
	cloudy	rain shwrs	light rain	rain shwrs	rain shwrs	rain shwrs	some clouds	light rain	rain shwrs	rain shwrs	rain shwrs	rain shwrs
Rain mm	-	0.6	4.6	0.2	0.1	0.1	-	0.5	4.4	1.9	0.7	0.6
High	12	13	12	12	13	14	14	14	13	14	15	16
Low	11	13	11	12	13	13	13	14	11	12	15	13
Feels°C	10	10	11	10	12	12	11	12	10	12	14	14
Humid %	64	71	85	76	67	65	59	56	76	60	53	60
UV	-	-	-	-	-	-	-	-	-	-	-	-
Sunrise	6:28	-	-	6:26	-	-	6:24	-	-	6:24	-	-
Sunset	-	8:28	-	-	8:29	-	-	8:30	-	-	8:30	-

Metoda Monte Carlo nu dezvoltă un model pentru procese, ci este o metodă de simulare. De exemplu, putem reproduce/genera toate condițiile meteo posibile care ar putea să apară într-un interval de timp specific. Procesul de reproducere este într-un fel similar cu aruncarea unei monede sau a unui zar. Desigur, aceasta este o situație simplă și nu este nevoie să folosim o simulare. Toată lumea știe că șansele de a cădea pe față la o aruncare de monedă sunt 1/2 = 50%.

Dar pentru a calcula dacă o furtună se va transforma într-un uragan este o situație foarte complexă care implică un număr mare de variabile, inclusiv temperatura apei, temperatura aerului, umiditatea, locația, condițiile prevalente ale vântului, etc. Pentru a complica lucrurile, avem doar câteva date din unele locații, iar pentru aceste date există și o marjă de eroare.

Vremea este mult prea complexă pentru a fi calculată simplu din cap, iar toate variabilele sunt foarte incerte. Astfel, se folosește o simulare pe computer. Această simulare pe computer ia datele pe care le avem și încearcă diferite combinații. De exemplu:
Experiment 1: Temperatura de 22.2 grade Celsius, viteza vântului 9.66 km/h NV
Experiment 2: Temperatura de 21.1 grade Celsius, viteza vântului 11.27 km/h V
Experiment 3: Temperatura de 21.7 grade Celsius, viteza vântului 8.05 km/h N
Etc.

Prognozele meteo folosesc simulări Monte Carlo pentru a crea intervale de probabilitate. (ranges of linklihood)

Calculatorul poate face acest lucru de câteva mii de ori cu mii de variabile. Acest lucru creează acele previziuni ușor de înțeles, cum ar fi 70% șanse de ploaie. Aceste previziuni ne permit să luăm decizii importante, cum ar fi: luăm sau nu umbrela cu noi??

6. Bibliografie și webografie

1) https://thatsmaths.com/2020/05/28/the-monte-carlo-method/
2) https://koaha.org/wiki/Metodo_Montecarlo
3) https://www.scientia.ro/stiinta-la-minut/110-matematica/2995-ce-sunt-simularile-monte-carlo.html
4) Programming Distributed Computing Systems: A Foundational Approach, Carlos A. Varela, 2013, MIT Press
5) https://www.linkedin.com/advice/1/how-can-you-use-monte-carlo-methods-analyze-ohfnc
6) https://www.ibm.com/blog/monte-carlo-simulations-with-ibm-cloud-functions/
7) https://towardsdatascience.com/using-a-monte-carlo-simulation-to-forecast-extreme-weather-events-d17671149d3e