Algoritmi Probabiliști

Bibliografie

- Horia Georgescu. **Tehnici de programare**, Editura Universităţii din Bucureşti 2005

- Gilles Brassard, Paul Bratley – **Algorithmics: theory and practice**, Prentice Hall, 1988

Algoritmi Probabilişti

- În timpul rezolvării unei probleme, putem ajunge la un moment dat în situaţia de a avea **de ales** între mai multe variante de continuare.

Algoritmi Probabilişti

- În timpul rezolvării unei probleme, putem ajunge la un moment dat în situaţia de a avea **de ales** între mai multe variante de continuare.
  - **se alege aleator una dintre variante**

Algoritmi Probabilişti

- În timpul rezolvării unei probleme, putem ajunge la un moment dat în situaţia de a avea **de ales** între mai multe variante de continuare.
  - **se alege aleator una dintre variante**
  - **la executări diferite ale unui algoritm probabilist, rezultatele sunt în general diferite.**

Algoritmi Probabiliști

Categorii

- Monte Carlo
- Las Vegas
- Algoritmi numerici

Algoritmi Monte Carlo

Algoritmi Monte Carlo

- Furnizează totdeauna un rezultat, care însă **nu este neapărat corect**
- **Probabilitatea ca rezultatul să fie corect creşte pe măsură ce timpul disponibil creşte**

Algoritmi Monte Carlo

Se consideră un vector cu *n* elemente distincte. Să se determine un element al vectorului care să fie mai mare sau egal cu mediana a celor n numere din vector

- n este foarte mare
- timpul avut la dispoziţie este mic

Algoritmi Monte Carlo – Mediana

$v = -\infty$

Repetă fără a depăşi timpul disponibil:

- alegem aleatoriu x un element al vectorului
- v = maxim(v, x) = cel mai mare element ales

scrie v

Algoritmi Monte Carlo – Mediana

- Care este probabilitatea ca un element ales x să fie mai mic decât mediana?

Algoritmi Monte Carlo – Mediana

- **Care este probabilitatea ca răspunsul să fie corect după k încercări?**

Algoritmi Monte Carlo – Mediana

- **Care este probabilitatea ca răspunsul să fie greşit după k încercări?**
  - Care este probabilitatea ca un element ales x să fie mai mic decât mediana?
  - Care este probabilitatea ca **toate** cele k elemente alese (în timpul de rulare avut la dispoziţie) să fie mai mici decât mediana (deci v < mediana)?

Algoritmi Monte Carlo – Mediana

- Care este probabilitatea ca răspunsul să fie corect după k încercări?

  $1 - 1/2^k$

  - **Pentru k=20, această probabilitate este mai mare decât 0,999999**

- $1 - (1-p)^k$

Algoritmi Monte Carlo

Se consideră un vector cu *n* elemente. Să se determine dacă există un element majoritar în vector (cu frecvența > n/2)

Algoritmi Monte Carlo – Element majoritar

$v = -\infty$

Repetă fără a depăşi timpul disponibil:

- alegem aleator x un element al vectorului
- Calculam f = frecventa lui x
- Daca f>n/2 scrie DA; STOP

scrie NU

Algoritmi Monte Carlo – Element majoritar

Analiza

- **Problemă de decizie**
- Dacă scrie DA, raspunsul este corect
- Dacă scrie NU, este posibil ca să existe element majoritar (deci răspunsul să fie greșit)

Algoritmi Monte Carlo – Element majoritar

Analiza

- **Problemă de decizie**
- Dacă scrie DA, raspunsul este corect
- Dacă scrie NU, este posibil ca să existe element majoritar (deci răspunsul să fie greșit)
- Care este probabilitatea de a răspunde greșit NU după k pași?

Algoritmi Monte Carlo

Suplimentar – Algoritmul lui KARGER de determinare a unei tăieturi minime într-un graf

- D. R. Karger, Global min-cuts in rnc, and other ramifications of a simple min-out algorithm. In Proceedings of the Fourth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA, 1993
- D. R. Karger, S. Clifford, A new approach to the minimum cut problem, Journal of the ACM. 43 (4), 1996 doi:10.1145/234533.234534.
- https://en.wikipedia.org/wiki/Karger's_algorithm

Algoritmi Las Vegas

Algoritmi Las Vegas

- **Nu furnizează totdeauna un rezultat**, dar dacă furnizează un rezultat atunci acesta este **corect**

- **Probabilitatea ca algoritmul să se termine creşte pe măsură ce timpul disponibil creşte**

Algoritmi Las Vegas

Se dau n texte (n foarte mare) cu următoarele proprietăţi:

- **există un unic text t_0 care apare de cel puţin 10% ori;**
- **celelalte texte sunt distincte.**

Se cere determinarea textului t_0.

Algoritmi Las Vegas – Text

Algoritm probabilist

Idee

- Generăm aleatoriu doi indici i și j și testăm dacă

  $i \neq j$ și $t_i = t_j$

  până când testul se încheie cu succes

Algoritmi Las Vegas – Text

```
repeat
   if LVText()
        stop
until false


LVText()
      i ← random(1..n); j ← random(1..n);
      if i≠j and t_i=t_j
          write t_i; return true
      else
          return false
```

Algoritmi Las Vegas - Text

Probabilitatea p de succes la un pas (**`LVText`**() returnează true):

$=$ probabilitatea p ca $t_i = t_j = t_0$, $j \neq i$

p = ?

Algoritmi Las Vegas – Text

Probabilitatea p de succes (**`LVText`**() returnează true):

= probabilitatea p ca $t_i = t_j = t_0$, $j \neq i$

- probabilitatea ca $t_i = t_0$
- probabilitatea ca $t_j = t_0$, $j \neq i$

Algoritmi Las Vegas – Text

- Probabilitatea p de succes (**`LVText`**() returnează true):

  = probabilitatea p ca $t_i = t_j = t_0$, $j \neq i$

  - probabilitatea ca $t_i = t_0$
  - probabilitatea ca $t_j = t_0$, $j \neq i$

$$p = \frac{1}{10}\left(\frac{1}{10} - \frac{1}{n}\right) \geq \frac{9}{1000}$$ pentru $n \geq 100$

Algoritmi Las Vegas – Text

- Probabilitatea p de succes (**`LVText`**() returnează true):
  - = probabilitatea p ca $t_i = t_j = t_0$, $j \neq i$
    - probabilitatea ca $t_i = t_0$
    - probabilitatea ca $t_j = t_0$, $j \neq i$

$$p = \frac{1}{10}\left(\frac{1}{10} - \frac{1}{n}\right) \geq \frac{9}{1000} \text{ pentru } n \geq 100$$

- Teoretic sunt suficiente $t = 1/p \approx 112$ încercări (apeluri ale **`LVText`**())

Algoritmi Las Vegas

- Analiza timpului estimat pentru răspuns cu succes

```
repeat
   if LV()
        stop
until false
```

Algoritmi Las Vegas

- Analiza timpului estimat pentru răspuns cu succes

```
repeat
   if LV()
        stop
until false
```

- s = timpul mediu a unei rulări cu succes a `LV()`
- f = timpul mediu a unei rulări cu eșec a `LV()`
- p = probabilitatea de succes

Algoritmi Las Vegas

- **Analiza timpului estimat pentru răspuns cu succes**

```
repeat
   if LV()
        stop
until false
```

- s = timpul mediu a unei rulări cu succes a `LV()`
- f = timpul mediu a unei rulări cu eșec a `LV()`
- p = probabilitatea de succes
- t = **timpului estimat pentru răspuns cu succes** (repetând funcția `LV()`) = ?

Algoritmi Las Vegas

- **Analiza timpului estimat pentru răspuns cu succes**

```
repeat
   if LV()
        stop
until false
```

- s = timpul mediu a unei rulări cu succes a `LV()`
- f = timpul mediu a unei rulări cu eșec a `LV()`
- p = probabilitatea de succes
- t = **timpului estimat pentru răspuns cu succes** (repetând funcția `LV()`)

$$t = p \cdot s + (1-p) \cdot (f+t) \Rightarrow t = s + f \cdot (1-p)/p$$

Algoritmi Las Vegas

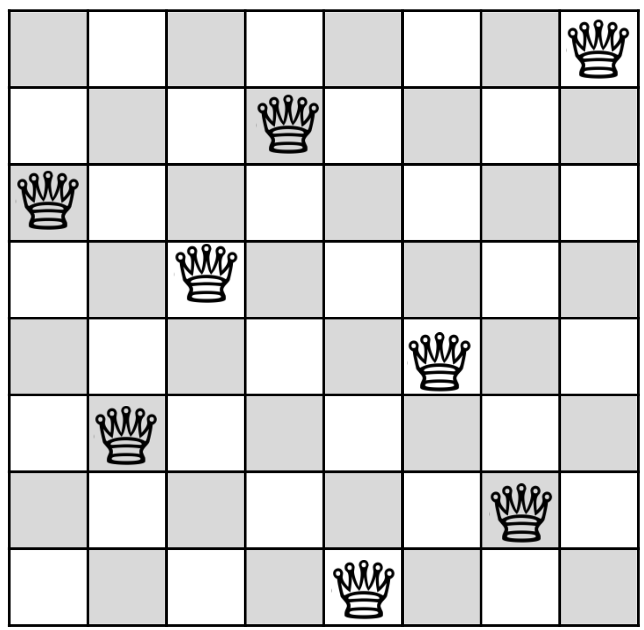
- **Analiza timpului estimat pentru răspuns cu succes**
  - Pentru s = f (cum este cazul `LVText()`) obținem

$$t = s \cdot 1/p$$

Algoritmi Las Vegas

Problema celor n dame

Se consideră un caroiaj n×n. Prin analogie cu o tablă de şah (n=8), se doreşte plasarea a n dame pe pătrăţelele caroiajului, astfel încât să nu existe două dame una în bătaia celeilalte



Algoritmi Las Vegas – Problema damelor

- **Reprezentarea soluţiei**

  $\mathbf{x} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\}$, unde

  $\mathbf{x}_k$ = coloana pe care este plasată dama de pe linia $\mathbf{k}$

  $\mathbf{x}_k \in \{1, 2, \ldots, n\}$

Algoritmi Las Vegas – Problema damelor

- **Backtracking**

  n = 8 – explorate 114 vârfuri (din 2057)

  din arborele asociat spațiului de căutare

  până când este găsită prima soluție

Algoritmi Las Vegas – Problema damelor

Algoritm probabilist

Idee

- pornim de la prima linie

repetă

- plasăm o damă aleatoriu pe linia curentă
- dacă dama nu atacă nicio damă deja plasată se trece la linia următoare

  altfel

până când se ajunge la linia n+1

Algoritmi Las Vegas – Problema damelor

Algoritm probabilist

Idee

- pornim de la prima linie

repetă

- plasăm o damă aleatoriu pe linia curentă
- dacă dama nu atacă nicio damă deja plasată se trece la linia următoare

altfel **eșec** (return false) **reluăm întreg algoritmul, nu facem doar un pas înapoi ca la Backtracking** (linia curentă devine prima linie)

până când se ajunge la linia n+1

Algoritmi Las Vegas - Problema damelor

Cum gestionăm diagonalele şi coloanele pe care s-au plasat deja dame?

Algoritmi Las Vegas – Problema damelor

Vectorii:

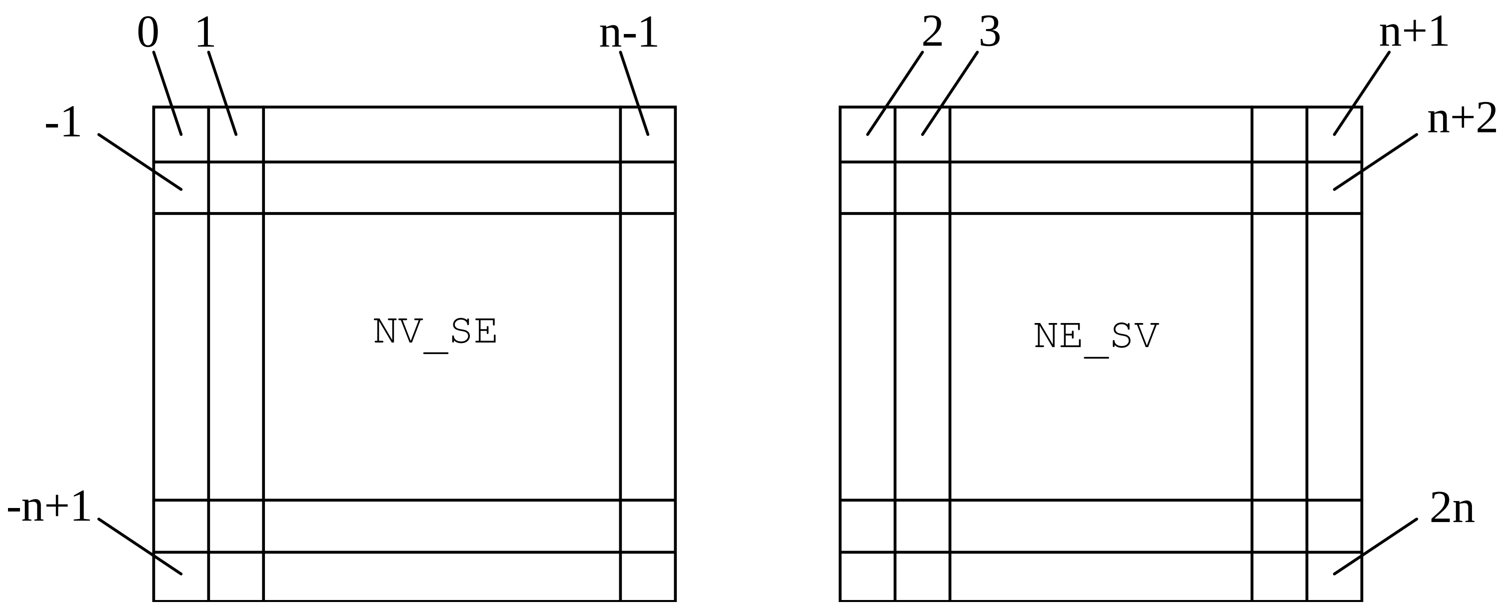
- Diagonale paralele cu diagonala principală (**j – i = constant**)

  NV_SE[–n+1..n–1]

- Diagonale paralele cu diagonala secundară (**j + i = constant**)

  NE_SV[2..2n]

- Coloane **C[1..n]**



Algoritmi Las Vegas – Problema damelor

```
repeat
    if LVDame() then
        stop
until false
```

LVDame()

- **inițializăm** C, NV_SE, NE_SV cu true
- k ← 1

repeat

-

-

until **k=n+1**

write(x)

return true

```
LVDame()
    • inițializăm C, NV_SE, NE_SV cu true
    • k ← 1
    repeat
      • formăm un vector a cu acele poziții i∈{1,...,n}
        cu C[i] = NV_SE[i-k] = NE_SV[i+k] = true
        şi notăm na lungimea vectorului obținut
      •





    until k=n+1
    write(x)
    return true
```

```
LVDame()
    • inițializăm C, NV_SE, NE_SV cu true
    • k ← 1
    repeat
      • formăm un vector a cu acele poziții i∈{1,...,n}
        cu C[i] = NV_SE[i-k] = NE_SV[i+k] = true
        şi notăm na lungimea vectorului obținut
      • if na>0 then
            aleg aleator i∈{1,...,na};  poz ← a_i
            x_k ← poz ;{plasam aleator dama pe o pozitie corecta}




    until k=n+1
    write(x)
    return true
```

```
LVDame()
    • inițializăm C, NV_SE, NE_SV cu true
    • k ← 1
    repeat
      • formăm un vector a cu acele poziții i∈{1,...,n}
        cu C[i] = NV_SE[i-k] = NE_SV[i+k] = true
        şi notăm na lungimea vectorului obținut
      • if na>0 then
            aleg aleator i∈{1,...,na};  poz ← a_i
            x_k ← poz ;
            NV_SE[poz-k] ← false; NE_SV[poz+k] ← false;
            C[poz] ← false;
            k ← k+1
        else

    until k=n+1
    write(x)
    return true
```

```
LVDame()
    • inițializăm C, NV_SE, NE_SV cu true
    • k ← 1
    repeat
      • formăm un vector a cu acele poziții i∈{1,...,n}
        cu C[i] = NV_SE[i-k] = NE_SV[i+k] = true
         şi notăm na lungimea vectorului obținut
      • if na>0 then
            aleg aleator i∈{1,...,na};  poz ← a_i
            x_k ← poz ;
            NV_SE[poz-k] ← false; NE_SV[poz+k] ← false;
            C[poz] ← false;
            k ← k+1
        else
            return false
    until k=n+1
    write(x)
    return true
```

Algoritmi Las Vegas - Problema damelor

- **Analiza probabilitate succes**
  - p = probabilitatea de succes
  - s = numărul mediu de vârfuri explorate la un succes
  - f = numărul mediu de vârfuri explorate la un eșec
  - t = numărul de vârfuri explorate până la încheierea cu succes (repetând funcția LVDame())

$$\mathbf{t = s + f \cdot (1-p)/p}$$

Algoritmi Las Vegas – Problema damelor

- **Experimente $n = 8$**
  - $p \approx 0.1293$
  - $f \approx 6.971$
  - $s = 9$
  - $\mathbf{t = s + f \cdot (1-p)/p} \approx 56$

Algoritmi Las Vegas – Problema damelor

- **Experimente $n = 8$**
  - $p \approx 0.1293$
  - $f \approx 6.971$
  - $s = 9$
  - $\mathbf{t = s + f \cdot (1-p)/p} \approx 56$
- **Backtracking –114**
- **Soluții mixte – k dame plasate aleatoriu, apoi backtracking**

Algoritmi numerici

Algoritmi numerici

- Urmăresc determinarea aproximativă a unei valori
- **Cu cât timpul alocat executării algoritmului este mai mare, precizia rezultatului se îmbunătăţeşte**

Algoritmi numerici

- **Aproximarea** lui π
- **Aproximarea** $\int_a^b f(x)\,dx$

$$f:[a,b]\to[c,d]$$

Aproximarea lui π

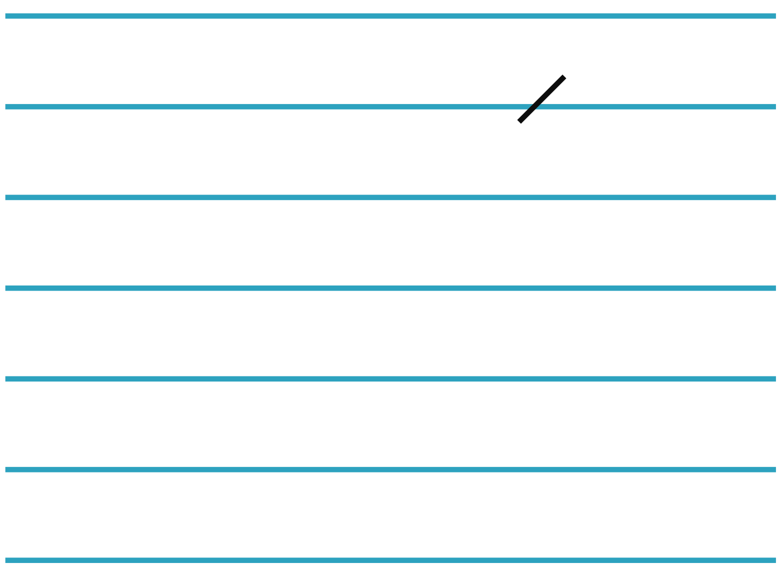
1. Acul lui Buffon

Se consideră o mulţime de linii paralele astfel încât oricare două linii vecine sunt la distanţă de o unitate.

Aproximarea lui π

1. Acul lui Buffon

Se consideră o mulţime de linii paralele astfel încât oricare două linii vecine sunt la distanţă de o unitate. Un ac de lungime o jumătate de unitate este aruncat aleator şi se numără de câte ori a intersectat vreo linie.



Aproximarea lui π

1. Acul lui Buffon

- Probabilitatea ca acul să intersecteze o linie este $1/\pi$

Aproximarea lui π

1. Acul lui Buffon

- **Probabilitatea ca acul să intersecteze o linie este $1/\pi$**
- După un număr "suficient de mare" de încercări, raportul între:

  – **`numărul total de încercări`**

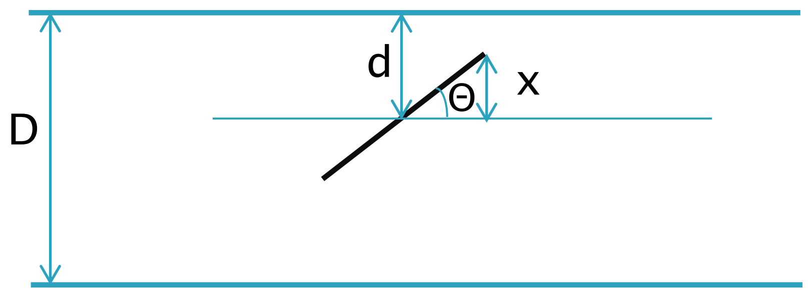
  `- numărul cazurilor în care acul a intersectat vreo linie`

va fi "suficient de aproape" de π.

Aproximarea lui π

1. Acul lui Buffon

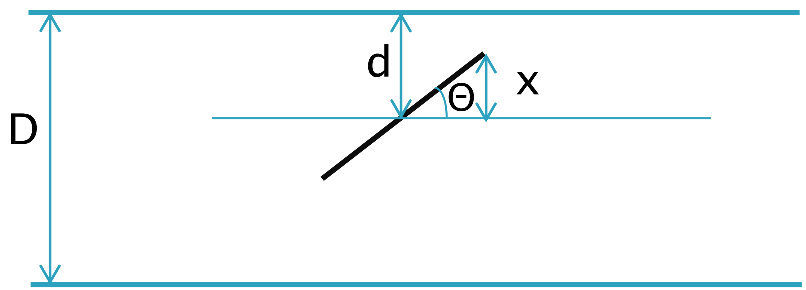
- **Justificare:**
  - L – lungimea acului (exp L=1/2)
  - D – distanța dintre drepte (L<D, exp D=1)
  - Acul – unic identificat de perechea (Θ,d), unde



Aproximarea lui π

1. Acul lui Buffon

- **Justificare:**
  - L – lungimea acului (exp L=1/2)
  - D – distanța dintre drepte (L<D, exp D=1)
  - Acul – unic identificat de perechea (Θ,d), unde
    - d = distanța de la centrul acului la cea mai apropiată dreaptă (linie) din mulțime
    - Θ = unghiul format de ac cu direcția dreptelor paralele

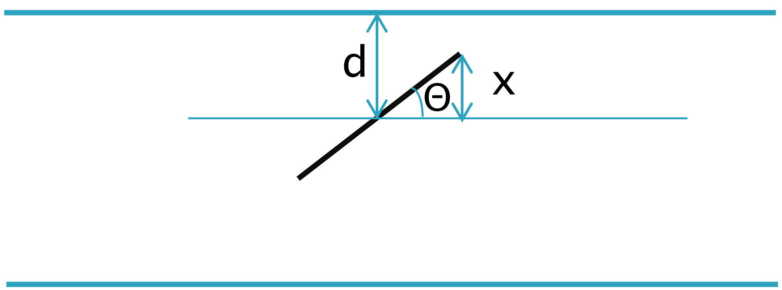


Aproximarea lui π

1. Acul lui Buffon

- **Justificare:**
  - Acul – unic identificat de perechea (Θ,d), unde
    - d $\in$[0, D/2]
    - Θ $\in$[0, 180º]
  - **"Aruncare ac" $\Leftrightarrow$ generare pereche (Θ,d)**
  - Acul intersectează dreapta cea mai apropiată $\Leftrightarrow$

    $d \le x = L/2 \sin(\Theta)$



Aproximarea lui π

1. Acul lui Buffon

- **Justificare:**
  - Poziții posibile ac:
    - $T = \{(\Theta, d) \mid d \in [0, D/2], \Theta \in [0, \pi]\}$
  - Poziții ac – care intersectează dreaptă:
    - $F = \{(\Theta, d) \mid \Theta \in [0, \pi], 0 \le d \le L/2 \sin(\Theta)\}$
  - Probabilitatea ca acul să intersecteze dreapta:

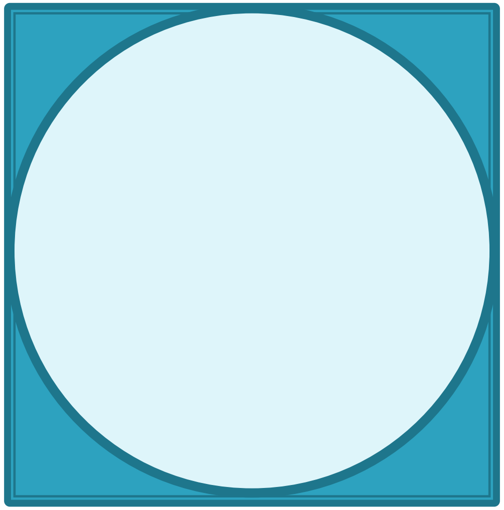
$$\frac{arie(F)}{arie(T)} = \frac{\left|\int_0^{\pi} \frac{L}{2} \sin(\Theta)\, d\Theta\right|}{\pi D/2} = \frac{2L/2}{\pi D/2} = \frac{2L}{\pi D}$$

Pentru L=1/2 și D=1 probabilitatea este $\frac{1}{\pi}$

Aproximarea lui π

2. Se aruncă repetat cu o săgeată într-un panou pătrat, cu ținta un cerc înscris în pătrat.

Se presupune că săgeata nimereşte totdeauna panoul.



Aproximarea lui π

2. Se aruncă repetat cu o săgeată într-un panou pătrat, cu ținta un cerc înscris în pătrat.

Se presupune că săgeata nimereşte totdeauna panoul.

Atunci raportul dintre:

- numărul cazurilor în care săgeata nimereşte în cercul înscris în pătrat

- numărul total de încercări

tinde la

$$\frac{arie\ cerc}{arie\ patrat} = \frac{\pi r^2}{(2r)^2} = \frac{\pi}{4}$$

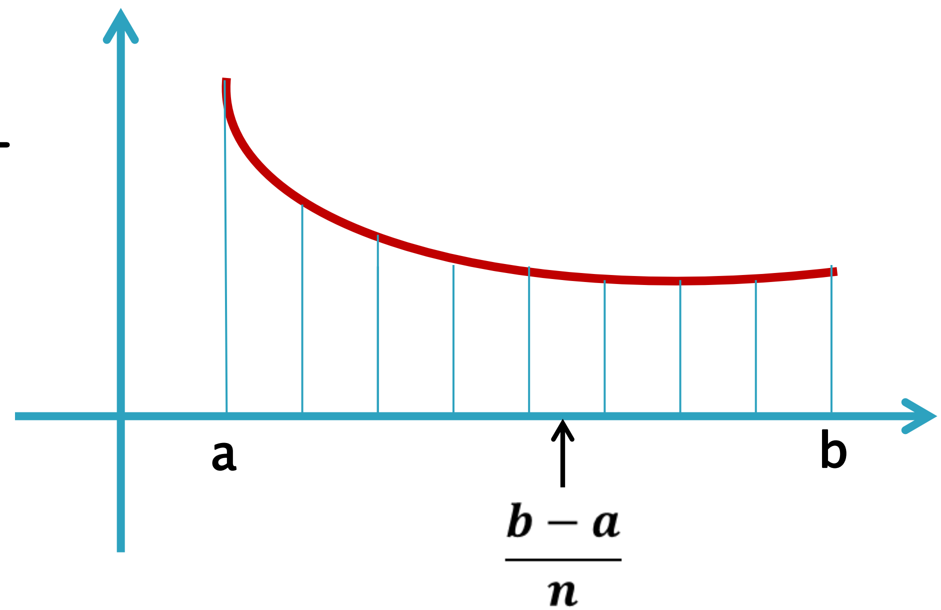
Aproximarea integralei

- $$\int_a^b f(x)\,dx, \quad f:[a,b]\to[c,d]$$

Aproximarea integralei

- $$\int_a^b f(x)\,dx, \quad f:[a,b]\to[c,d]$$

n încercări



Aproximarea integralei

$$\int_a^b f(x)\,dx, \qquad f:[a,b]\rightarrow[c,d]$$

```
s ← 0
for i=1,n
    x ← random([a,b]);
    s ← s+f(x)

s ← s·(b-a)/n
write(s)
```