Projeto 2 - Sistemas Aleatórios

Instituto de Física de São Carlos
Universidade de São Paulo

Vinícius Bastos Marcos (12556715)

Introdução à Física Computacional
Prof. Francisco Castilho Alcaraz

Outubro, 2022

Tarefa A

A tarefa A pede para se calcular as médias de x, x^2, x^3 e x^4, com o intuito de testar um gerador de números aleatórios. Para isso, fiz uma função que fez isso, para que conseguisse colocar os quatro em um só programa, da forma mais compacta possível. Essa função pede duas variáveis, o expoente *nexp* e e o número de passos k, que é o número de números aleatórios a serem somados.

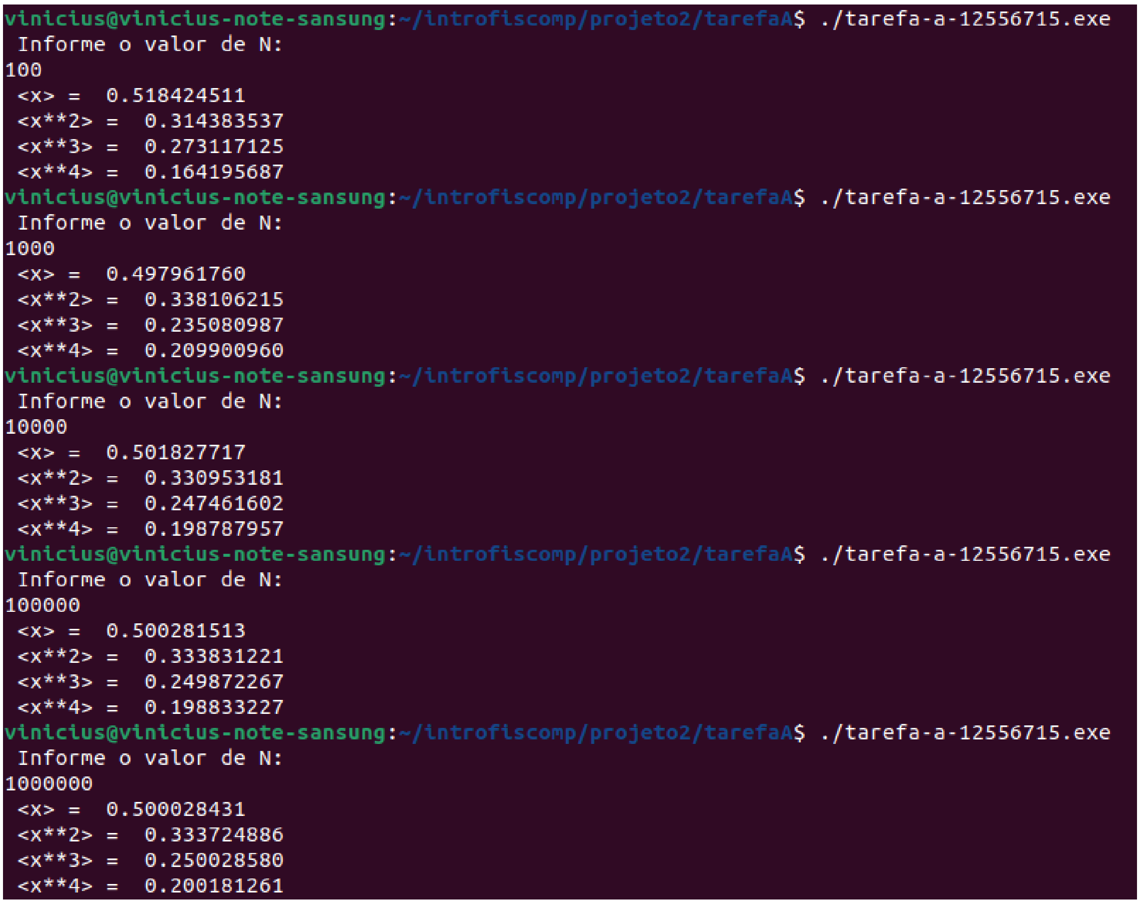
```
c       tarefa A

        function rmedia(nexp,k)
        rm = 0.e0
        do npassos = 1, k
           x = rand()
           rm = rm + x**nexp
        enddo
        rN = k
        rmedia = rm/rN
        return
        end function rmedia

        write(*,*) 'Informe o valor de N:'
        read(*,*) N

        write(*,*) '<x> =', rmedia(1,N)
        write(*,*) '<x**2> =', rmedia(2,N)
        write(*,*) '<x**3> =', rmedia(3,N)
        write(*,*) '<x**4> =', rmedia(4,N)

        end
```

Algoritmo 1: código para resolução da tarefa A

Assim, o usuário entra com o número N de passos e conseguimos calcular e escrever na tela os respectivos valores. Assim, vemos que conforme aumentamos o valor de N, chegamos perto do valor esperado que é

$$< x^n >= \int_0^1 x^n = \frac{1}{n+1}.$$

Pois a função *rand()* gera números aleatórios entre 0 e 1. Abaixo temos alguns teste que mostram os valores obtidos com a implementação do programa.



```
vinicius@vinicius-note-sansung:~/introfiscomp/projeto2/tarefaA$ ./tarefa-a-12556715.exe
 Informe o valor de N:
100
 <x> =  0.518424511
 <x**2> =  0.314383537
 <x**3> =  0.273117125
 <x**4> =  0.164195687
vinicius@vinicius-note-sansung:~/introfiscomp/projeto2/tarefaA$ ./tarefa-a-12556715.exe
 Informe o valor de N:
1000
 <x> =  0.497961760
 <x**2> =  0.338106215
 <x**3> =  0.235080987
 <x**4> =  0.209900960
vinicius@vinicius-note-sansung:~/introfiscomp/projeto2/tarefaA$ ./tarefa-a-12556715.exe
 Informe o valor de N:
10000
 <x> =  0.501827717
 <x**2> =  0.330953181
 <x**3> =  0.247461602
 <x**4> =  0.198787957
vinicius@vinicius-note-sansung:~/introfiscomp/projeto2/tarefaA$ ./tarefa-a-12556715.exe
 Informe o valor de N:
100000
 <x> =  0.500281513
 <x**2> =  0.333831221
 <x**3> =  0.249872267
 <x**4> =  0.198833227
vinicius@vinicius-note-sansung:~/introfiscomp/projeto2/tarefaA$ ./tarefa-a-12556715.exe
 Informe o valor de N:
1000000
 <x> =  0.500028431
 <x**2> =  0.333724886
 <x**3> =  0.250028580
 <x**4> =  0.200181261
```

Figura 1: alguns testes do código feito para a Tarefa A

Tarefa B

A Tarefa B pede que calculemos algumas médias a partir do andar do bêbado em uma dimensão. Para isso, nos é fornecida a probabilidade do andarilho andar para direita (p) e para esquerda (q), que é complementar a de andar para direita: $q = 1 - p$.

Também é pedido um histograma da quantidade de andarilhos em função da posição após N passos. O número de andarilhos utilizados nesta tarefa foi de 100000.

Tarefa B1

Nesta primeira parte, as probabilidades são iguais a $1/2$ e o número de passos é 1000. Para saber se ele vai para esquerda ou direita, fiz uso de condicionais e o gerador de números aleatórios introduzido na tarefa anterior.

A variável x, que é um número compreendido entre zero e um, vai definir para qual sentido ele anda, adicionando ou subtraindo uma unidade (um passo) na posição ipx dele, da seguinte forma: se $0 \leq x \leq p$ somamos um, e se $p < x \leq 1$ subtraímos uma unidade. Vemos isso implementado em ForTran77 abaixo.

```
c       tarefa b1

c       criando um vetor para guardar as informações
        dimension isaida(100000) !M = número de andarilhos
```

```
c      arquivo de saída para graficar
       iout = 10 !unidade arquivo de saída
       open(unit=iout, FILE='saida-b1.dat')

c      cálculo da posição de cada andarilho e soma das médias
       M = 100000
       N = 1000 !número de passos
       p = 0.5e0 !probabilidade direita
       soma1 = 0.e0
       soma2 = 0.e0
       do i = 1, M
           ipx = 0
           do j = 1, N
               x = rand()
               if (x <= p) then
                   ipx = ipx + 1
               else
                   ipx = ipx - 1
               endif
           enddo
           isaida(i) = ipx
           soma1 = soma1 + ipx
           soma2 = soma2 + ipx**2
       enddo

c      transformando o vetor no arquivo a ser plotado
       min = isaida(1)
       max = isaida(1)
       do i = 1, M
           if (isaida(i) < min) then
             min = isaida(i)
           endif
           if (isaida(i) > max) then
             max = isaida(i)
           endif
       enddo

       iamplitude = max - min
       njanelas = 14 !vão ser njanelas + 1
       np = iamplitude / njanelas
       np_atual = min - np
       icount = 0

       do while(np_atual <= max)
           np_atual = np_atual + np
           do i = 1, M
               if (isaida(i)<=np_atual .and. isaida(i)>np_atual-np) then
```

```
                  icount = icount + 1
                endif
            enddo
            write(iout,*) np_atual, icount
            icount = 0
        enddo

c       cálculo das médias
        write(*,*) 'O número M de andarilhos é', M
        write(*,*) 'As médias são:'
        write(*,*) '<x> =', soma1/M
        write(*,*) '<x**2> =', soma2/M

c       fechando o arquivo de saída
        close(iout)

        end
```

Algoritmo 2: código para resolução da tarefa B1

As médias calculadas estão dispostas logo abaixo, assim como o histograma.

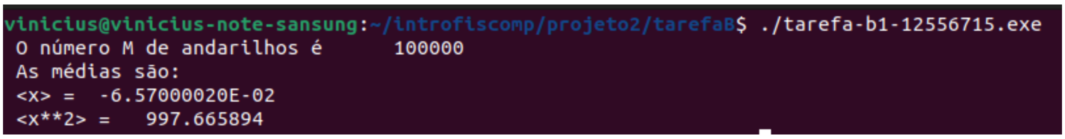


Figura 2: teste pedido para a Tarefa B1

Figura 3: histograma da Tarefa B1

Tarefa B2

Por outro lado, essa tarefa pede três casos quando p e q são diferentes, além de pedir a forma analítica de ser calcular as médias. A lógica utilizada é exatamente igual a anterior.

```
c      tarefa b2

c      criando um vetor para guardar as informações
       dimension isaida(100000) !M = número de andarilhos

c      arquivo de saída para graficar
       iout = 10 !unidade arquivo de saída
       open(unit=iout, FILE='saida-b2-15.dat')

c      cálculo da posição de cada andarilho e soma das médias
       M = 100000
       N = 1000 !número de passos
c      Neste próximo p mudei caso a caso na tarefa B2
       p = 1.e0/5.e0 !probabilidade direita (que vou mudando conforme o
     ↪ caso)
       soma1 = 0.e0
       soma2 = 0.e0
       do i = 1, M
           ipx = 0
           do j = 1, N
               x = rand()
               if (x <= p) then
                   ipx = ipx + 1
               else
                   ipx = ipx - 1
               endif
           enddo
           isaida(i) = ipx
           soma1 = soma1 + ipx
           soma2 = soma2 + ipx**2
       enddo

c      transformando o vetor no arquivo a ser plotado
       min = isaida(1)
       max = isaida(1)
       do i = 1, M
           if (isaida(i) < min) then
             min = isaida(i)
           endif
           if (isaida(i) > max) then
             max = isaida(i)
           endif
       enddo

```

```
      iamplitude = max - min
      njanelas = 14 !vão ser njanelas + 1
      np = iamplitude / njanelas
      np_atual = min - np
      icount = 0

      do while(np_atual <= max)
          np_atual = np_atual + np
          do i = 1, M
              if (isaida(i)<=np_atual .and. isaida(i)>np_atual-np) then
                icount = icount + 1
              endif
          enddo
          write(iout,*) np_atual, icount
          icount = 0
      enddo

c     cálculo das médias
      write(*,*) 'O valor de p é 1/5'
      write(*,*) 'O número M de andarilhos é', M
      write(*,*) 'As médias são:'
      write(*,*) '<x> =', soma1/M
      write(*,*) '<x**2> =', soma2/M
c     fechando o arquivo de saída
      close(iout)

      end
```

Algoritmo 3: código para resolução da tarefa B2

Os testes e os histogramas pedidos estão dispostos logo abaixo.

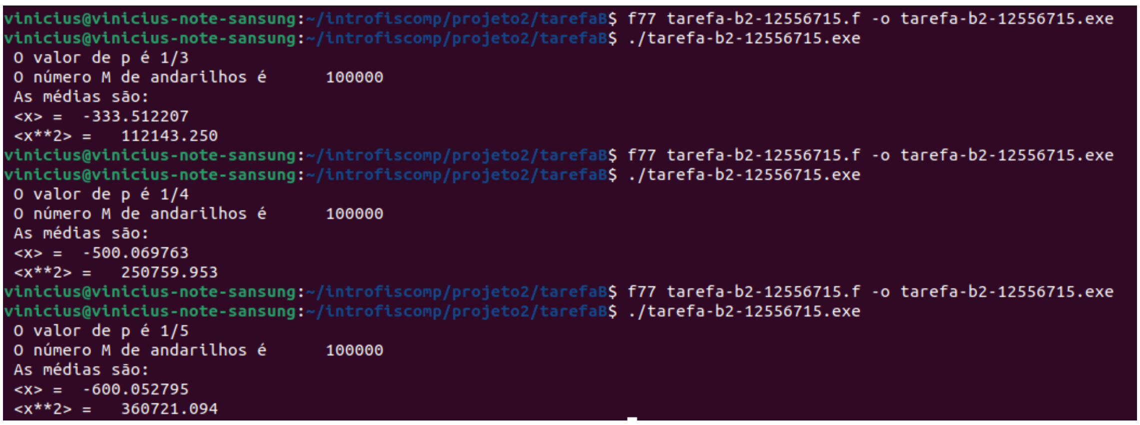


Figura 4: testes pedidos para a Tarefa B2

Figura 5: histograma da Tarefa B2, com p = 1/3

Figura 6: histograma da Tarefa B2, com p = 1/4

Figura 7: histograma da Tarefa B2, com p = 1/5

A forma analítica de se calcular $< x >$ e $< x^2 >$ é dada por:

$$< x >= N \cdot (p - q)$$

$$< x^2 >= [N \cdot (p - q)]^2 + 4pqN$$

Assim, vemos que os valores batem com os calculados a seguir e com os calculados na parte B1.

Tarefa C

A Tarefa C é uma extrapolação da anterior com mais dimensões, no caso duas. Para isso, nos é fornecido que as probabilidades do andarilho andar para qualquer sentido são iguais, e portanto 1/4. A lógica utilizada é igual a da Tarefa B, mas com duas dimensões, chamada por mim de *ipx* e *ipy*.

A posição do andarilho em cada dimensão é dada por: se $0 \leq x \leq p$ somamos um ao ipx, se $p < x \leq 2p$ subtraímos uma unidade no ipx, se $2p < x \leq 3p$ somamos um ao ipy, se $3p < x \leq 4p = 1$ subtraímos uma unidade no ipy.

Mas como o histograma pedido é de número de andarilhos em função da distância da origem, implementei uma função que calcula a distância, que é igual a $\sqrt{(ipx)^2 + (ipy)^2}$. Vemos o que foi explicado implementado no código de fortran abaixo.

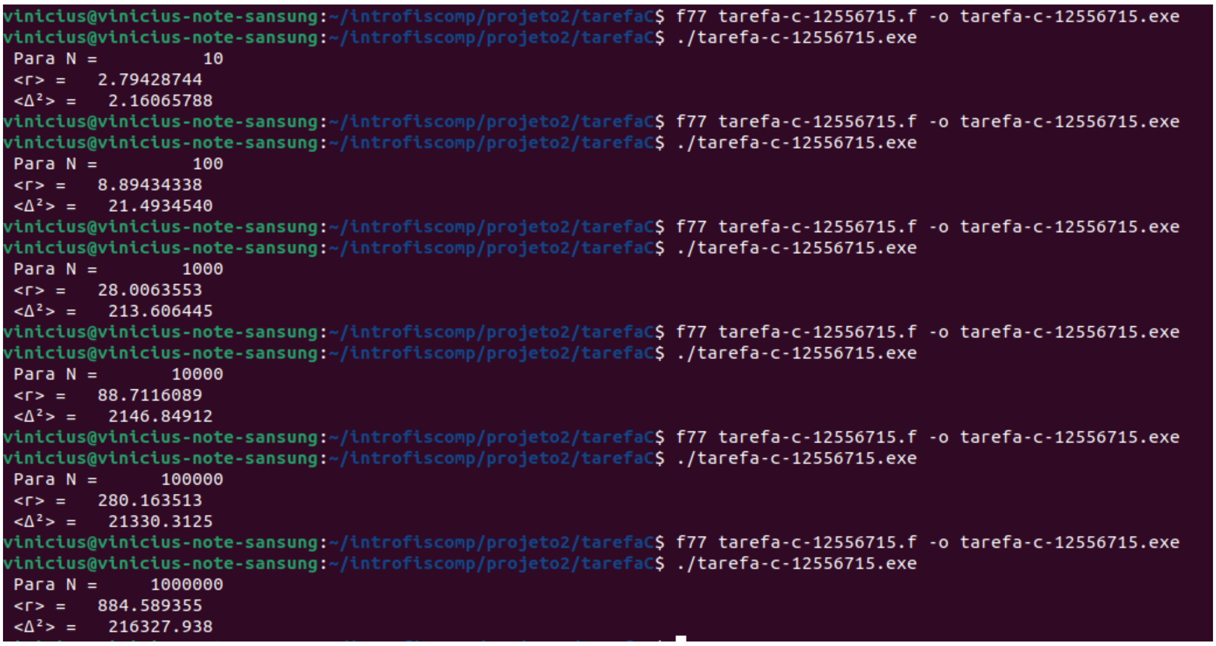
```
c      tarefa c

```

```fortran
c       função distância
        function dist(ix, iy)
            vx = ix
            vy = iy
            dist = sqrt(vx**2 + vy**2)
        return
        end function dist

c       criando um vetor para guardar as informações
        dimension saida(100000) !M = número de andarilhos

c       arquivo de saída para graficar
        iout = 10 !unidade arquivo de saída
        open(unit=iout, FILE='saida-c.dat')

c       cálculo da posição de cada andarilho e soma das médias
        M = 100000
        N = 1000000 !número de passos (que vou ir mudando)
        pdir = 1.e0/4.e0 !probabilidade direita
        pesq = 2.e0/4.e0 !probalidade esquerda
        pcima = 3.e0/4.e0 !probalidade cima
        pbaixo = 4.e0/4.e0 !probalidade baixo
        soma1 = 0.e0
        soma2 = 0.e0
        do i = 1, M
            ipx = 0
            ipy = 0
            do j = 1, N
                x = rand()
                if (x <= pdir) then
                    ipx = ipx + 1
                else if (pdir < x .and. x <= pesq) then
                    ipx = ipx - 1
                else if (pesq < x .and. x <= pcima) then
                    ipy = ipy + 1
                else if (pcima < x .and. x <= pbaixo) then
                    ipy = ipy - 1
                endif
            enddo
            write(iout,*) ipx, ipy
            d = dist(ipx, ipy)
            saida(i) = d
            soma1 = soma1 + d
            soma2 = soma2 + d**2
        enddo

c       cálculo das médias
        write(*,*) 'Para N =', N
```

```
      write(*,*) '<r> =', soma1/M
      write(*,*) '<²> =', (soma2/M - (soma1/M)**2)

c     fechando o arquivo
      close(iout)

      end
```

Algoritmo 4: código para resolução da tarefa C

A tarefa requisita que troquemos o valor dos passos, N, para que conseguíssemos observar a evolução de uma difusão ao longo do tempo. Os valores das médias calculados para cada N e cada histograma estão dispostos abaixo. Conseguimos com isso, observar que com a evolução do tempo, as moléculas tender a ser espalhar pelo espaço cada vez mais.



```
vinicius@vinicius-note-sansung:~/introfiscomp/projeto2/tarefaC$ f77 tarefa-c-12556715.f -o tarefa-c-12556715.exe
vinicius@vinicius-note-sansung:~/introfiscomp/projeto2/tarefaC$ ./tarefa-c-12556715.exe
 Para N =           10
 <r> =    2.79428744
 <Δ²> =    2.16065788
vinicius@vinicius-note-sansung:~/introfiscomp/projeto2/tarefaC$ f77 tarefa-c-12556715.f -o tarefa-c-12556715.exe
vinicius@vinicius-note-sansung:~/introfiscomp/projeto2/tarefaC$ ./tarefa-c-12556715.exe
 Para N =          100
 <r> =    8.89434338
 <Δ²> =    21.4934540
vinicius@vinicius-note-sansung:~/introfiscomp/projeto2/tarefaC$ f77 tarefa-c-12556715.f -o tarefa-c-12556715.exe
vinicius@vinicius-note-sansung:~/introfiscomp/projeto2/tarefaC$ ./tarefa-c-12556715.exe
 Para N =         1000
 <r> =    28.0063553
 <Δ²> =    213.606445
vinicius@vinicius-note-sansung:~/introfiscomp/projeto2/tarefaC$ f77 tarefa-c-12556715.f -o tarefa-c-12556715.exe
vinicius@vinicius-note-sansung:~/introfiscomp/projeto2/tarefaC$ ./tarefa-c-12556715.exe
 Para N =        10000
 <r> =    88.7116089
 <Δ²> =    2146.84912
vinicius@vinicius-note-sansung:~/introfiscomp/projeto2/tarefaC$ f77 tarefa-c-12556715.f -o tarefa-c-12556715.exe
vinicius@vinicius-note-sansung:~/introfiscomp/projeto2/tarefaC$ ./tarefa-c-12556715.exe
 Para N =       100000
 <r> =    280.163513
 <Δ²> =    21330.3125
vinicius@vinicius-note-sansung:~/introfiscomp/projeto2/tarefaC$ f77 tarefa-c-12556715.f -o tarefa-c-12556715.exe
vinicius@vinicius-note-sansung:~/introfiscomp/projeto2/tarefaC$ ./tarefa-c-12556715.exe
 Para N =      1000000
 <r> =    884.589355
 <Δ²> =    216327.938
```

Figura 8: testes pedidos para a Tarefa C

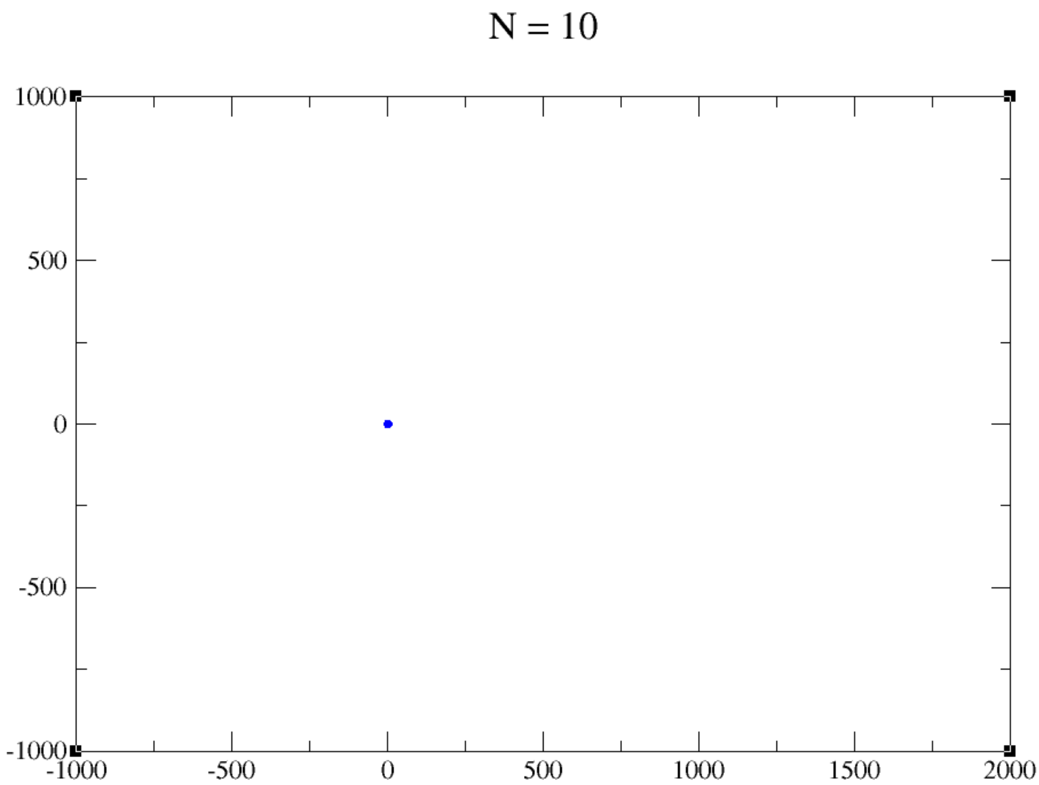


Figura 9: histograma da Tarefa C, com N = 10

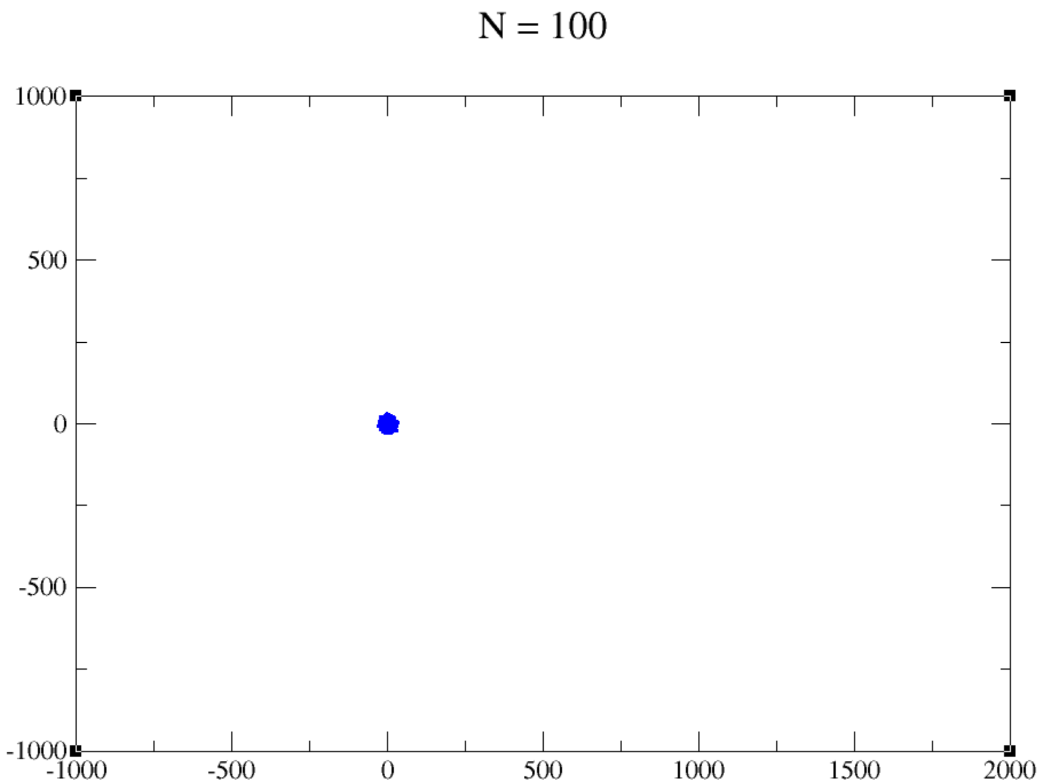


Figura 10: histograma da Tarefa C, com N = 100

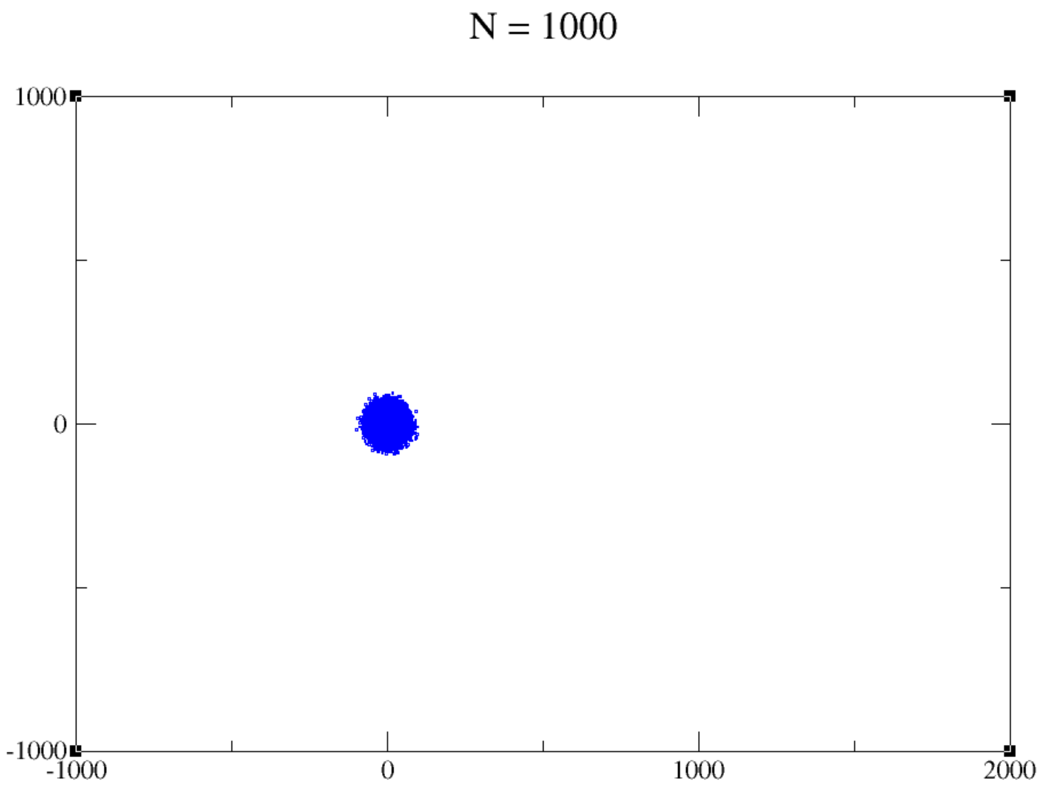


Figura 11: histograma da Tarefa C, com N = 1000

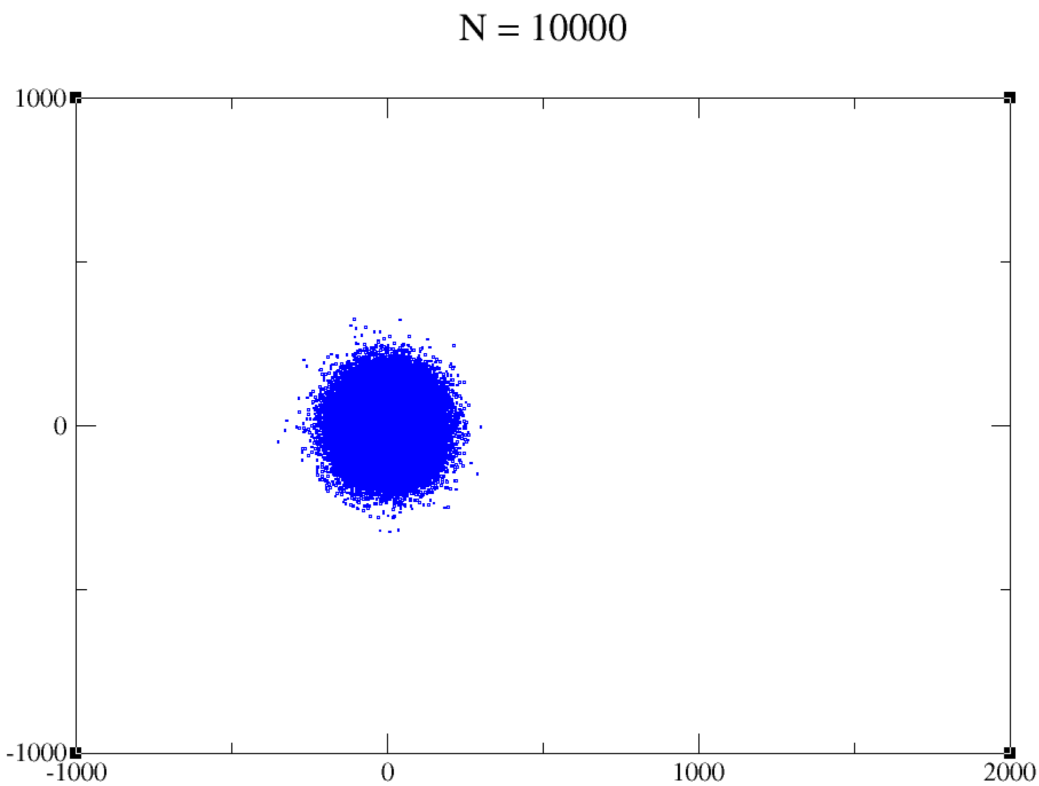


Figura 12: histograma da Tarefa C, com N = 10000

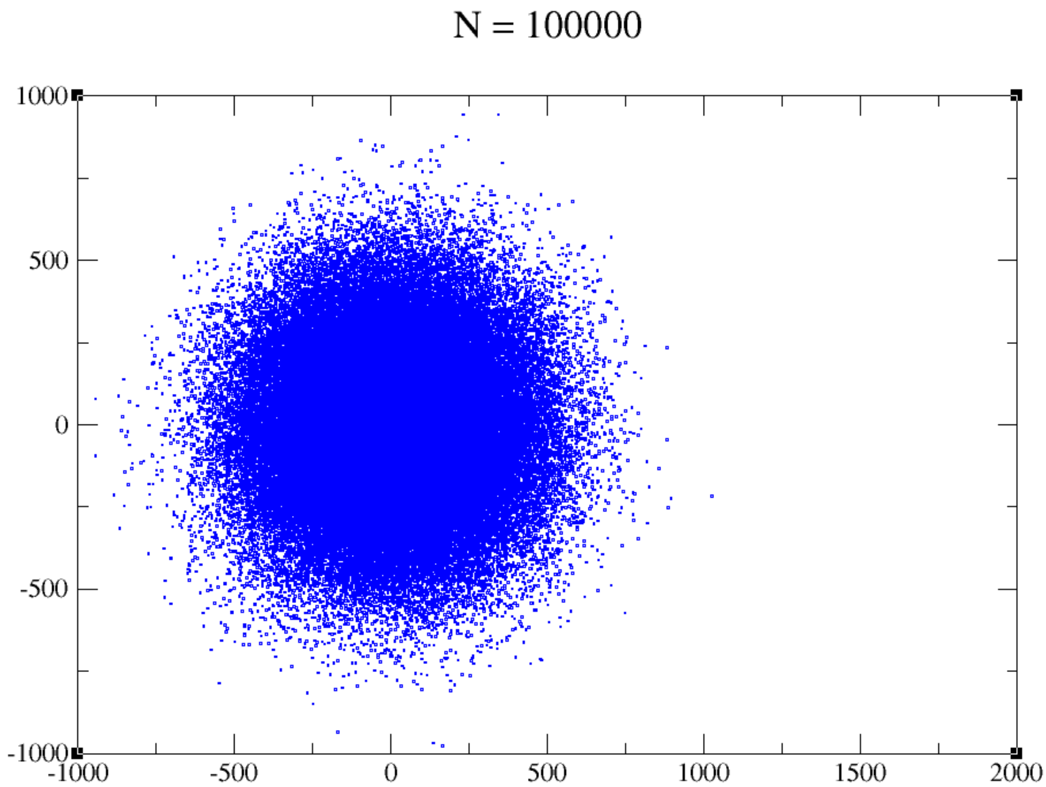


Figura 13: histograma da Tarefa C, com N = 100000

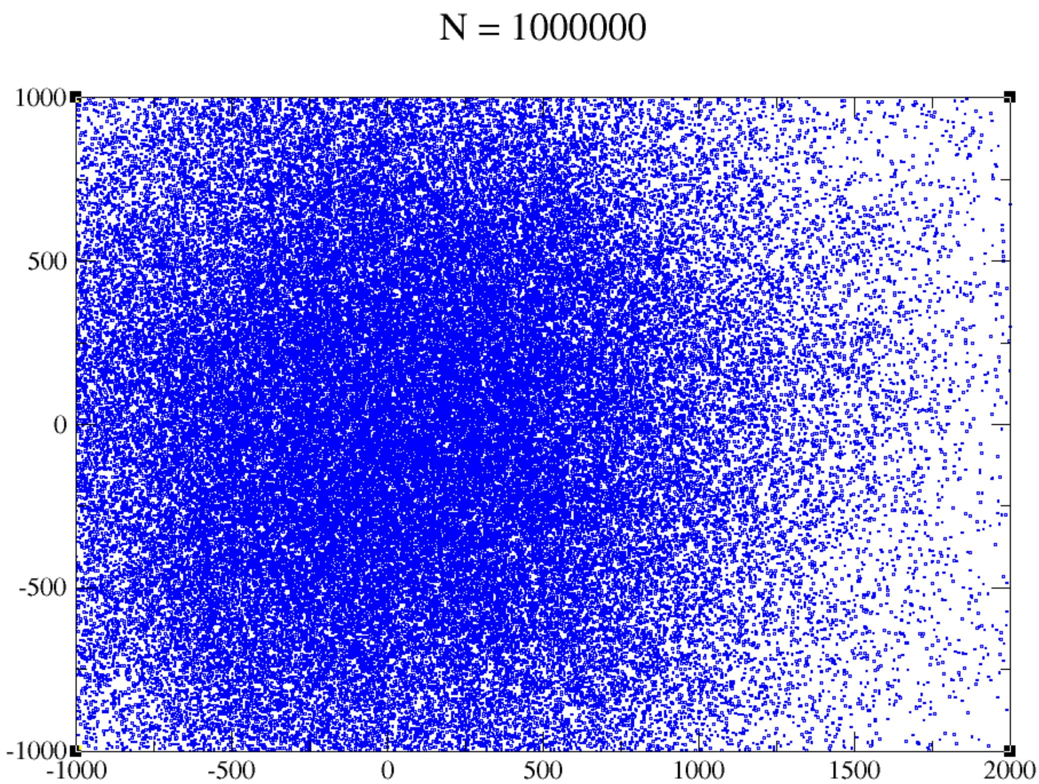


Figura 14: histograma da Tarefa C, com N = 1000000, na escala das anteriores

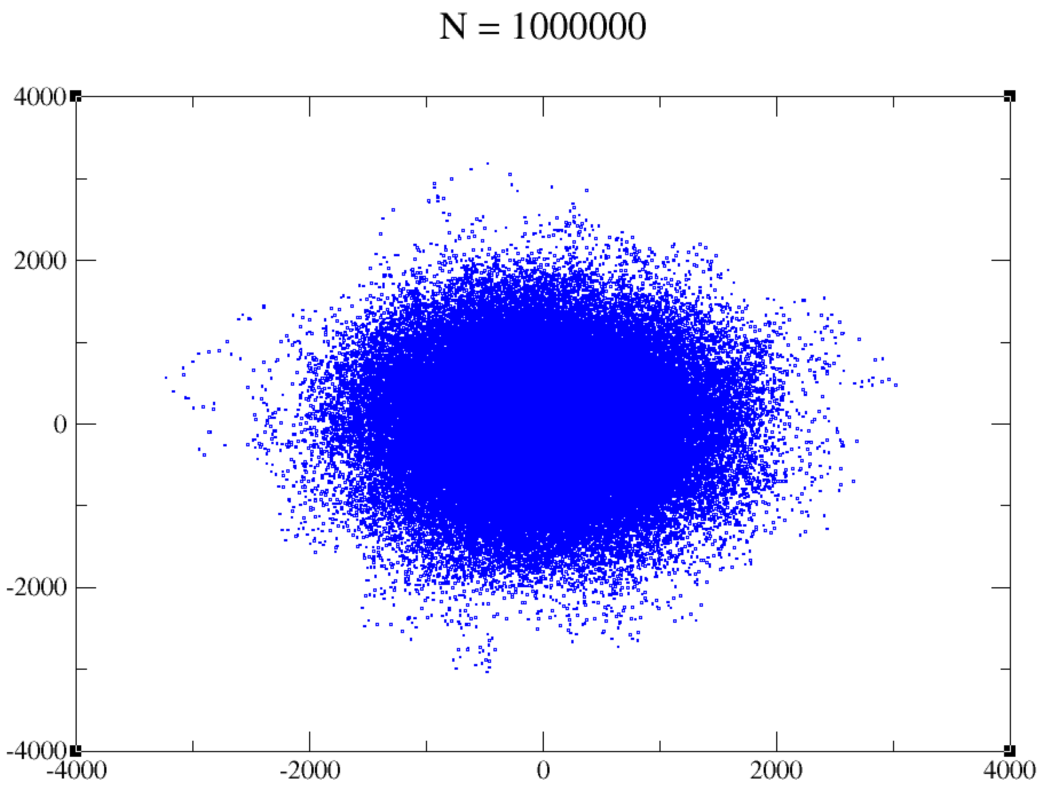


Figura 15: histograma da Tarefa C, com N = 1000000, em outra escala

Tarefa D

Na Tarefa D, temos que calcular a entropia relacionada com estado do problema estudado na tarefa anterior (Tarefa C), da seguinte forma:

$$S = -\sum_i P_i \cdot ln(P_i)$$

sendo P_i a probabilidade de se encontrar um andarilho no estado i.

Para isso, temos que dividir o espaço total máximo (quadrado NxN) em micro espaços, que considerei com tendo um lado igual a 10 unidades. A cada passo, que considerei como sendo um instante de tempo que passa, vou contar quantos andarilhos tem em cada micro espaço e calcular a entropia neste espaço. Assim, somo a de todos os micro espaços e tenho a do espaço todo, para cada N.

Dessa forma, coloco no arquivo de saída a entropia total do espaço e o instante de tempo em que ela foi calculada, o N.

O código implementado e o gráfio da entropia versus passos (tempo) estão dispostos logo abaixo.

```
c      tarefa d

c      constantes, vetores e arquivo de saída
       parameter(M = 10000) !número de andarilhos
       parameter(N = 1000) !número de passos
       dimension ipos(-N:N,-N:N) !matriz que guarda posição
```

```fortran
      dimension ipx(M)
      dimension ipy(M)
      open(unit=iout, FILE='saida-d.dat')

c     iniciando a matriz posição
      do i = -N, N, 1
        do j = -N, N, 1
            ipos(i,j) = 0
        enddo
      enddo

c     iniciando os vetores posição
      do i = 1, M
         ipx(i) = 0
         ipy(i) = 0
      enddo

c     cálculo da posição de cada andarilho e soma das médias
      pdir = 1.e0/4.e0 !probabilidade direita
      pesq = 2.e0/4.e0 !probalidade esquerda
      pcima = 3.e0/4.e0 !probalidade cima
      pbaixo = 4.e0/4.e0 !probalidade baixo
      ilado = 10 !lado do micro espaço
      do i = 1, N !tempo
          Stotal = 0.e0
          do j = 1, M !andarilho
              x = rand()
              if (x <= pdir) then
                  ipx(j) = ipx(j) + 1
              else if (pdir < x .and. x <= pesq) then
                  ipx(j) = ipx(j) - 1
              else if (pesq < x .and. x <= pcima) then
                  ipy(j) = ipy(j) + 1
              else if (pcima < x .and. x <= pbaixo) then
                  ipy(j) = ipy(j) - 1
              endif
              ipos(ipx(j),ipy(j)) = ipos(ipx(j),ipy(j)) + 1
          enddo
          do k = -N, N - ilado, ilado !percorre o eixo x
            icont = 0
            do l = -N, N - ilado, ilado !percorre o eixo y
                do im = k, k+ilado, 1
                    do in = l, l+ilado, 1
                      icont = icont + ipos(im,in)
                    enddo
                enddo
            enddo
            !cálculo da entropia total
```

```
          if (icont /= 0) then
            cont = icont !transformando em real
            rM = M !transformando em real
            prob = cont/rM
            S = prob * log(prob)
            Stotal = Stotal - S
          endif
        enddo
        !reiniciando a matriz posição
        do i2 = -N, N, 1
          do j2 = -N, N, 1
            ipos(i2,j2) = 0
          enddo
        enddo
        write(iout,*) i, Stotal
      enddo

c     fechando o arquivo de saída
      close(iout)

      end
```

Algoritmo 5: código para resolução da tarefa D

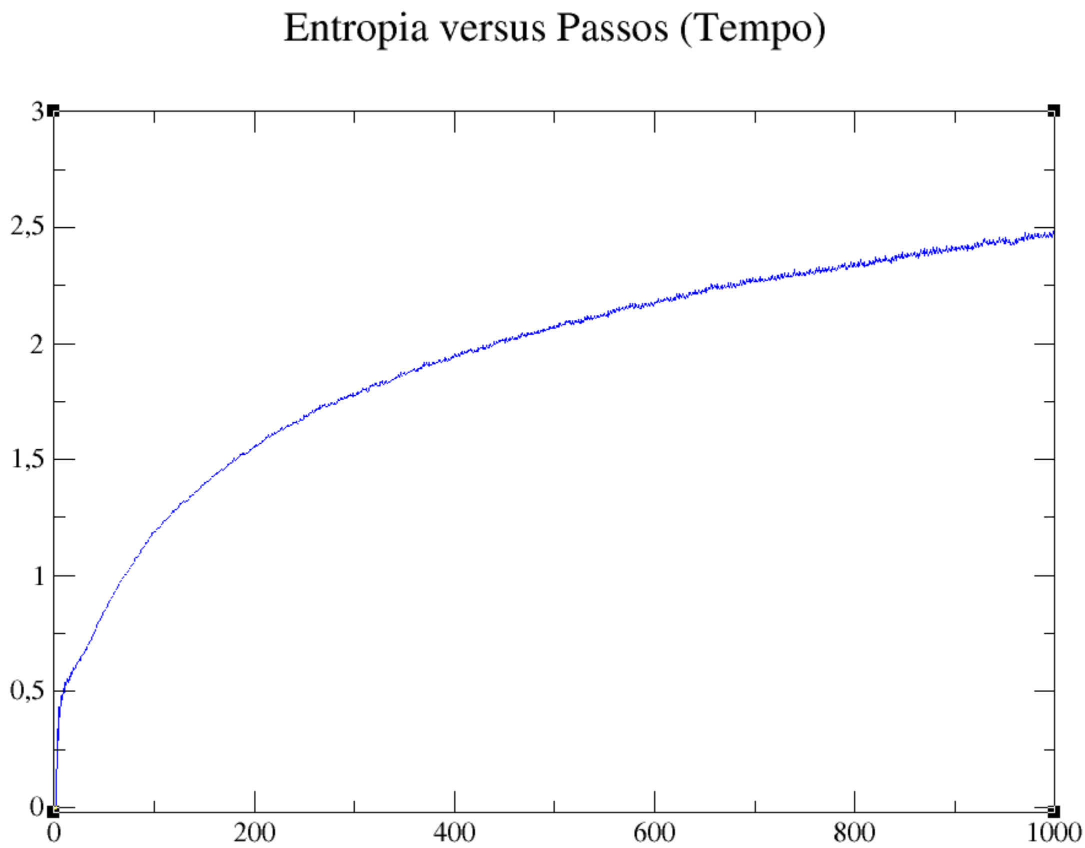


Figura 16: gráfico da Tarefa D