Vítor Paixão Damasceno

Prática 18
Laboratório de AEDS

Belo Horizonte, Brasil

2024

1 Introdução

Neste laboratório, atualizamos a simulação desenvolvida na aula 17 implementando reprodução sexuada. Cada organismo terá um sexo definido (masculino ou feminino) codificado no DNA. Quando dois organismos de sexos diferentes estiverem próximos, eles podem se reproduzir, onde o filho será composto pela metade do DNA de um pai combinado com a metade do DNA do outro pai (além de alguma possível mutação).

Espera-se, com esse experimento, observar de que maneira diferentes métodos de reprodução influenciam aspectos de uma simulação com computação evolucionária.

2 Desenvolvimento

Iniciamos alterando a classe `Organismo` para cada indivíduo agora tenha um atributo `sexo`, que é inicializado com 0 ou 1, simbolizando masculino e feminino.

Em seguida, alteramos o método `reproduzir()` para que agora ele receba outro indivíduo como parâmetro.

Optamos por mesclar o DNA dos pais de forma randômica, ao invés de determinar uma metade para cada parente.

Para evitar que o filho nasça próximo ao pai e assim tenha chance de se reproduzir imediatamente novamente, o filho gerado aparece em um local aleatório no mapa.

Listing 1 – Método `reproduzir()`

```
1  Organismo reproduzir(Organismo outro) {
2    // Reproduz com uma probabilidade baseada na saude
3    if (random(1) < 0.1 && this.vida > 50) {
4      float[] novoDna = new float[3];
5
6      // Mescla DNA dos pais
7      for(int i = 0; i < 3; i++){
8        if((int)random(2) == 1)
9          novoDna[i] = this.dna[i];
10       else
11         novoDna[i] = outro.dna[i];
12     }
13
14     // Mutacao
15     for(int k = 0; k < novoDna.length; k++)
16       if(random(1) < 0.05)
17         novoDna[k] = constrain(novoDna[k] + random(-0.1, 0.1), 0,
               1);
```

```
18
19      vida -= 10;
20
21      // Filho nasce num local aleatorio
22      PVector novaPosicao = new PVector();
23      novaPosicao.y = (int)random(width);
24      novaPosicao.x = (int)random(height);
25
26      return new Organismo(novaPosicao, novoDna);
27    } else {
28      return null;
29    }
30  }
```

Feitas as mudanças no método `reproduzir()`, algumas atualizações foram necessárias para sua implementação na função `draw()`.

A variável global `populacaoMax` foi criada para evitar que, em algum ponto da simulação, haja uma explosão exponencial da população. Esse recurso serve como uma forma rápida de balancear a simulação, visto que é uma tarefa extremamente difícil encontrar um ponto de equilíbrio nos parâmetros da reprodução onde não haja superpopulação e nem extinção dos organismos.

O laço for com o contador `j = i + 1` nos permite iterar sobre todos os possíveis parceiros de reprodução apenas uma vez, sem que haja repetição de combinações.

Em seguida, a tentativa de reprodução só ocorre em indivíduos que estão próximos e possuem sexos opostos.

Listing 2 – Chamada do método `reproduzir()`

```
1  // Tenta reproduzir se houver espaco
2  if(populacao.size() < populacaoMax)
3    // Itera sobre os possiveis parceiros
4    for (int j = i + 1; j < populacao.size(); j++){
5      Organismo outro = populacao.get(j);
6
7      // Verifica se sao passiveis de reproducao
8      if(o.posicao.dist(outro.posicao) < o.tamanho * 2 && o.sexo !=
           outro.sexo){
9        Organismo filho = o.reproduzir(outro);
10       if (filho != null) {
11         populacao.add(filho);
12         break;
13       }
```

```
14      }
15    }
```

Além disso, para uma melhor visualização no contexto desse experimento, uma distinção visual entre os sexos foi implementada no método `mostra()`.

Listing 3 – Método `mostra()`

```
1  void mostra() {
2      stroke(0);
3      colorMode(HSB, 360, 100, 100);
4      fill(cor(map(velocidadeMax, 2, 5, 0, 100)));
5      // Desenha forma de acordo com o sexo
6      if(sexo == 0){
7        ellipse(posicao.x, posicao.y, tamanho, tamanho);
8      } else {
9        square(posicao.x, posicao.y, tamanho);
10     }
11     colorMode(RGB, 255, 255, 255);
12 }
```

3 Resultados

Feitas as alterações, observou-se uma maior variabilidade genérica quando comparada à prática 17, onde a reprodução era assexuada. Ao se avançar diversas gerações na simualção, agora é possível encontrar uma maior variedade de cores nos indivíduos evoluídos. Essas tonalidades geralmente esão entre o vermelho e o amarelo, visto que os organismos de cor verde são energéticamente ineficientes, e, portanto, morrem logo no início da simulação.

A seguir algumas imagens de diferentes estágios da simulação.

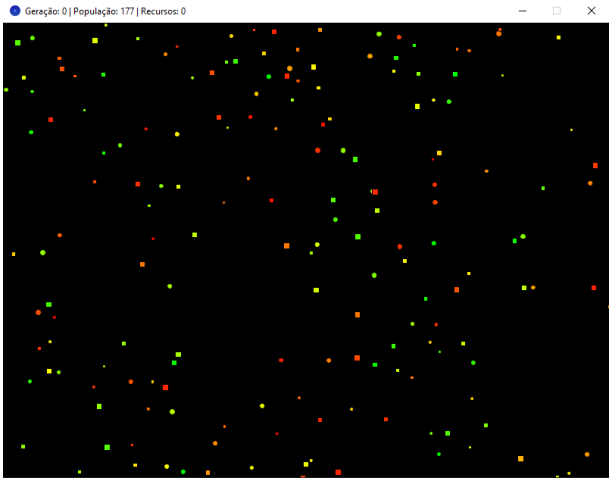

Figura 1 – estágio inicial da simulação

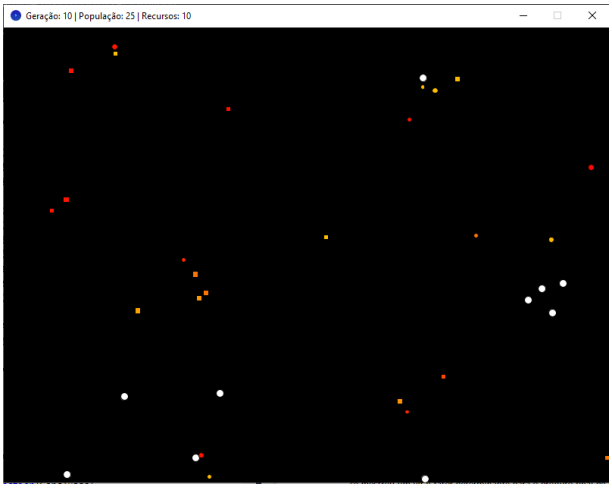

Figura 2 – Estágio intermediário da simualção

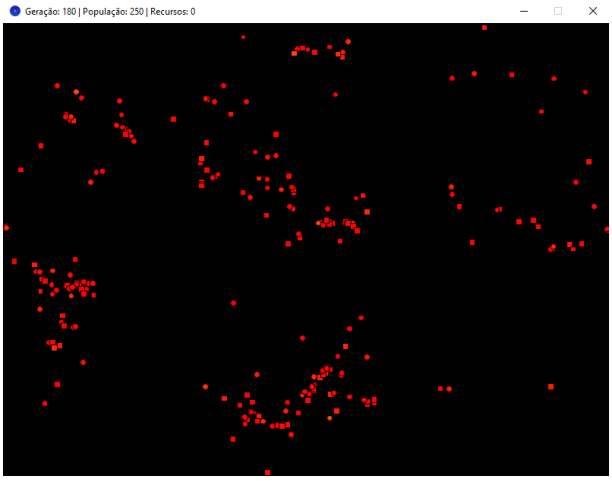

Figura 3 – Estágio avançado da simualção

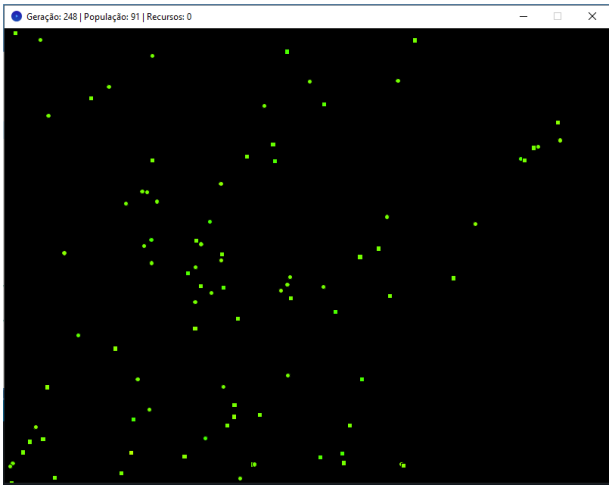

Figura 4 – Eságio avançado de outra iteração da simulação

4 Conclusão

Os resultados alcançados foram de acordo com os esperados, e a forma de reprodução se mostrou um vital fator determinante para o produto final de uma simulação com computação evolucionária.

A maior dificuldade encontrada durante a execução do experimento foi encontrar formas de balancear o crescimento populacional dos organismos de forma natural. A abordagem utilizada para resolver o problema foi a definição de um limite absoluto para a população, solução que, apesar de não ser ideal, é suficiente no contexto do experimento.

A prática se mostrou ter um grande valor prático e teórico, à medida que diferentes aspectos da computação evolucionária foram evidenciados durante a execução do experimento.