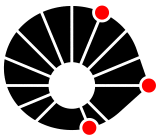
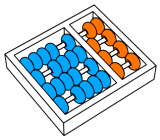




Clock Evolves: Algoritmo Evolutivo para Relógios Simulados

Nícolas Hecker Silva *Esther Luna Colombini*

Relatório Técnico - IC-PFG-25-31
Projeto Final de Graduação
2025 - Dezembro

UNIVERSIDADE ESTADUAL DE CAMPINAS
INSTITUTO DE COMPUTAÇÃO

Clock Evolves: Algoritmo Evolutivo para Relógios Simulados

Nícolas Hecker Silva* Esther Luna Colombini†

Resumo

Este trabalho consiste na implementação de um algoritmo evolutivo para obtenção de um relógio mecânico simulado. O surgimento de complexidade nos seres vivos através de mecanismos como seleção natural, reprodução com mutação e deriva genética é muito bem descrito na literatura científica. Como forma de rebater um exemplo anedótico típico de discursos relacionados ao design inteligente, este trabalho utilizou dos mecanismos descritos para a evolução em um pequeno conjunto de peças mecânicas simuladas incluindo vigas, engrenagens, pregos e âncoras. Os resultados foram analisados para identificar a sua capacidade de medir o tempo, trajeto evolutivo e número de gerações para serem obtidos. A conclusão foi que foi possível criar relógios muito simplificados através desse método.

1 Introdução

Em 24 de novembro de 1859, na Inglaterra, Charles Robert Darwin publicava seu livro *"On the Origin of Species by Means of Natural Selection, or the Preservation of Favoured Races in the Struggle for Life"* [6] ou em um título simplificado em português, "A Origem das Espécies". Nesse livro ele relata o processo da evolução das populações das espécies como ele mesmo pôde verificar através de suas viagens pelo mundo. Apesar de ainda incompleta, por não saber explicar a origem da variabilidade de características em populações, a sua teoria era um contraponto importante para a teoria Lamarckista de Lei do uso e desuso [8] e principalmente para a teoria fixista, explicação que dominava o senso comum na época. Apesar da resistência inicial, a teoria da evolução foi aceita como a mais adequada para explicar a origem das espécies pela comunidade científica. De acordo com essa teoria, em todo o mundo, existem populações de seres vivos da mesma espécie. Dentro dessa população, naturalmente existem indivíduos diferentes entre si. Estando competindo pela sobrevivência e reprodução, se iguais, não haveria qualquer distinção para qualquer indivíduo gerar mais descendentes em detrimento a outro. Contudo, com diferenças, os indivíduos cujas características atrapalharem sua capacidade de gerar mais descendentes, sejam psicológicas como a ausência de medo de predadores, ou estéticas, como as penas de pavões, acabam por serem removidos da população. Já se essas características não atrapalharem a sobrevivência ou ainda auxiliarem sua reprodução, a população possuirá mais indivíduos com as mesmas características que seus pais. Uma mesma característica pode ser vantajosa ou danosa para

*Instituto de Computação, Universidade Estadual de Campinas, Campinas, SP.
†Instituto de Computação, Universidade Estadual de Campinas, Campinas, SP.

os indivíduos dependendo do ambiente em que estão inseridos. Darwin argumentou que esse era o mecanismo por trás da existência de todas as espécies atuais, o que foi corroborado com a presença de fósseis intermediários de espécies vivas hoje.

Ao mesmo tempo, nos Estados Unidos, as perseguições religiosas que levaram muitos europeus a migrarem para a América motivaram a inclusão de emendas na Constituição, entre elas a Primeira Emenda, que estabelece a separação entre Igreja e Estado [14]. Essa disposição dificultou o ensino do fixismo nas escolas públicas. Como forma de contornar essa restrição, surgiu a corrente de pensamento do "Design Inteligente", que se apoia na ideia de complexidade irredutível, como explicado em 2012 por *Pirulla* em seu vídeo "Criacionismo [9] - Complexidade Irredutível" [11]. A complexidade irredutível é o ponto de falseabilidade da teoria da evolução. Se em algum momento for demonstrado que uma estrutura biológica não pode ter se formada por meio de modificações graduais em organismos preexistentes, então a teoria da evolução proposta por Darwin seria considerada insuficiente. Até o momento, todas os casos apontados como complexidade irredutível foram explicados como mudanças graduais em outros organismos, algumas mais abruptas como a hipótese de endossimbiose da mitocôndria [12] — explicando seu surgimento através da convivência proto-cooperativa de uma célula eucariota e uma procariota — outras baseadas em mudança de função, como a explicação do surgimento do flagelo bacteriano, que ocorreu no processo jurídico, o caso *Dover* [7] — cujo objetivo era permitir o ensino do criacionismo nas escolas públicas como contraponto a evolução.

Se existir tal complexidade irredutível, uma outra teoria ganharia espaço: O design inteligente [10]. Uma anedota conhecida motivadora para o entendimento dessa teoria é a seguinte:

"Imagine que ao andar sobre a praia de uma ilha deserta você encontra uma pedra. Ao questionar sobre sua origem, você imagina possibilidades. Ela poderia vir de uma rocha maior, sendo partida pelo vento, mar ou animais. Poderia ter sido trazida pelo mar de uma terra distante. Poderia ainda ter sido formada ali mesmo por uma atividade vulcânica a muito tempo no passado, e permanecida ali para ser encontrada. Satisfeito você continua sua jornada. Mais a frente você identifica um relógio de bolso. Ao percebê-lo você se enche de esperança, pois sabe que não está sozinho, aquele artefato pode ter sido trazido pelo mar, mas você sabe que possui origem humana. Isso ocorre, pois o relógio é muito complexo e não poderia ser gerado por processos naturais como a rocha. Ele portanto, precisa de um criador, um projetista. O mesmo pode ser dito sobre os seres vivos no mundo. Nós somos muito complexos para sermos gerados por processos naturais, e portanto precisamos de um projetista, sendo a complexidade irredutível evidência disso."

Essa anetoda utiliza da falácia do espantalho que reduz o argumento contrário a um argumento mais fácil de combater, entretanto as simplificações são grandes. Diferente do relógio de bolso, os seres vivos possuem três características que favorecem o surgimento de complexidade:

1. **A reprodução**. Essa propriedade possibilita gerar um novo indivíduo idêntico ao anterior, dando continuidade às suas características, mesmo quando o próprio indivíduo original deixou de existir. Essa reprodução pode ser sexuada, combinando as características dos indivíduos originais nos seus descendentes.

2. **A mutação**. Durante a vida dos indivíduos ou ainda durante o processo de reprodução, podem ocorrer mudanças nas características que serão passadas para os novos indivíduos, gerando variabilidade na população.

3. **A seleção**. Agindo sobre os indivíduos da população, a seleção discrimina-os sob um determinado critério, facilitando a sobrevivência ou reprodução de alguns em detrimento a outros.

A presença dessas 3 características move a população de seres vivos em direção ao critério apontado pela seleção. Os relógios podem sofrer mutações, se quebrando ou sendo alterados, podem sofrer seleção, sendo descartados os quebrados, muito confusos ou não atraentes, mas eles não podem se reproduzir, o que anula a evolução. Assim surge a pergunta: "Se relógios pudessem se reproduzir, sofrer mutações e seleção, eles seriam capazes de evoluir?".

O trabalho apresentado no vídeo "Evolution is a blind watchmaker" [4] na plataforma YouTube e traduzido no vídeo "A evolução É mesmo um relojoeiro cego" [9], apresenta um experimento em que ele garante essas três propriedades para estruturas constituídas de molas, âncoras, engrenagens e ponteiros, além de conexões entre elas. Elas são capazes de se reproduzirem, copiando o indivíduo pai, sofrerem mutações, e seleção, discriminando-os com respeito a sua *capacidade me medir o tempo*. Para realizar esse processo, o autor utilizou de multiplicações de matrizes em um programa em matlab. Entretanto ele não simulou os relógios fisicamente, sua definição de qualidade de medir o tempo não é muito precisa e ele parte de um número fixo de peças, além de garantir uma fonte de energia externa. Esses problemas são corrigidos neste trabalho: Uma simulação física de peças de relógios para evolução de um relógio mecânico.

O funcionamento de um relógio é ilustrado no Website de Bartosz Ciechanowski [5]. Das estruturas apresentadas, foram escolhidas as componentes mais básicas para serem simuladas. Os detalhes sobre as peças e parâmetros da simulação são detalhados no corpo deste trabalho.

O repositório github.com/salocinrevenge/ClockEvolves contém todo o código utilizado.

2 Objetivos

O principal objetivo deste trabalho é *Analisar a possibilidade de evoluir relógios mecânicos simulados apenas com algoritmo evolutivo.*

Assim, as perguntas de pesquisa que norteiam este trabalho são:

1. Como construir um ambiente para evolução de relógios em simulação?

2. Qual a velocidade da evolução dos indivíduos em número de gerações?

3. Quais as condições iniciais mínimas para que um relógio possa evoluir?

4. Qual a capacidade dos relógios evoluídos de medir o tempo?

Para responder a essas perguntas, foram definidas algumas restrições:

1. O tempo real máximo de cada simulação é de uma semana.

2. Um indivíduo é considerado um relógio mínimo se houver peças com velocidades não nulas após 5 segundos de simulação a frequência padrão de 120 estados por segundo.

Dessa forma, para responder tais perguntas, foi:

1. Construído um ambiente computacional para simulação de peças mecânicas;
2. Desenhado e testado um modelo esperado para os relógios;
3. Definido funções de pontuação exigindo o mínimo de propriedades dos indivíduos;
4. Testado as diferentes funções de seleção;
5. Testado diversas condições iniciais;
6. Analisado os resultados obtidos.

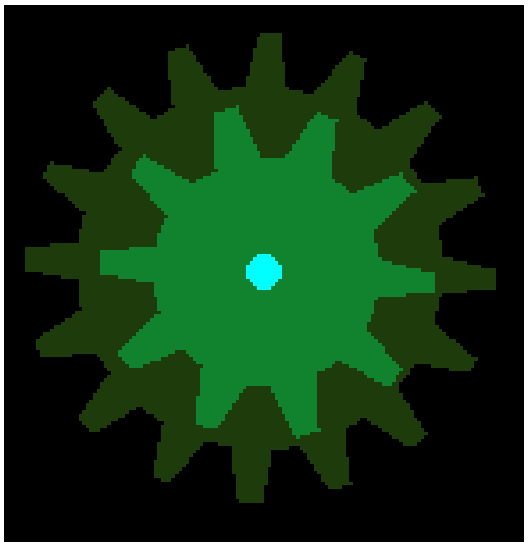
3 Metodologia

Para realizar as simulações, foi utilizada a biblioteca *Pygame*[13]. A execução de toda a simulação foi realizada utilizando apenas o processador, não sendo utilizado placas gráficas para acelerar a simulação ou qualquer etapa do código.

3.1 Ambiente de simulação

As primeiras tentativas de construir um ambiente para a simulação se basearam em escrever as equações de colisão e conservação de momento para os diversos polígonos utilizados. Devido a complexidade de resolução de colisões, essa abordagem foi abandonada e foi utilizado a biblioteca *Pymunk* [2] para controlar todas as peças e movimentos presentes. A construção de relógios mecânicos exige a utilização de estruturas complexas conectadas em diferentes camadas tridimensionais, como uma conexão simples entre duas engrenagens de diferentes tamanhos para alterar a velocidade linear de rotação (vide Figura 1a). Como, por simplicidade, é desejável a utilização de um ambiente bidimensional, mas há a necessidade de características tridimensionais, foi utilizado uma estrutura de dois planos, um a frente e outro atrás, em que todas as peças ocupam apenas um plano e colidem apenas com outras peças no mesmo plano.

(a) Duas engrenagens de tamanhos distintos conectadas por um eixo. Essa estrutura exige 3 dimensões. Imagem produzida com software Blender [1]



(b) Estrutura equivalente a Figura 1a construída dentro do ambiente de simulação.

Figura 1: Comparação entre o modelo das engrenagens em três dimensões e sua reprodução no ambiente de simulação.

Todas as simulações utilizaram apenas 4 tipos de peças:

1. Vigas: retângulos com uma dimensão muito maior que as outras. Ela pode ser rotacionada ou escalada até valores limite. Pode ser colocada em qualquer dos planos.

2. Pinos: Pinos ou pregos são pontos de conexão entre peças, podendo conectar com o fundo (pinos vermelhos) ou apenas nos pontos de peças sobrepostas (azul). Para conectar duas peças, é necessário que estejam em planos distintos. Não são adicionadas restrições re torque em um prego em particular, ou seja, duas peças conectadas por um único prego não necessariamente compartilharão da mesma velocidade angular.

3. Engrenagens: Círculo dentado que pode ser rotacionado ou escalado mantendo a mesma distância entre os dentes, podendo ser posicionado em qualquer um dos planos. As engrenagens também podem possuir seus dentes direcionados para alguma direção.

4. Âncoras: estruturas côncavas em formato de V. Pode ser escalado ou rotacionado e posicionado em qualquer um dos planos.

Além disso também são utilizado 4 segmentos estáticos para definir as bordas da simulação. A área da simulação é de 800 por 800 *pixels*.

Não foi utilizado nenhuma forma de fornecimento de energia externa. Todo energia da simulação deve ser proveniente da energia potencial gravitacional.

Para produzir as peças da simulação, a primeira ideia fora utilizar a função polígono do pymunk para shapes. Entretanto, ao projetar os pontos para a âncora, o polígono produzido foi convexo, ou seja, sem a abertura interna da peça. Por conta disso, foi necessário construir um corpo baseado em diversos polígonos menores. Para isso foi utilizado um algoritmo de triangularização. Esse algoritmo se baseia em remover pontas do polígono. Para isso, ele percorre todos os pontos e verifica os 2 posteriores a ele em ordem horária. Caso verifique que eles formam um triângulo externo (convexo), e que eles não contém nenhum outro ponto, percorrendo os demais, esses três pontos são retirados do vetor e anotados para retorno em um conjunto de triângulos de saída como um triângulo a parte. Sempre será possível realizar esse procedimento pois é possível triangularizar qualquer polígono cujas arestas não se sobrepõem. Como as peças nesse trabalho são previamente definidas, não ocorrerão casos de borda, logo este algoritmo é suficiente.

Para garantir que o ambiente de simulação permite a construção de um relógio mecânico funcional, foi construído um editor de relógios manual e a partir dele, projetado um relógio exemplo, como mostrado na Figura 2. Ele então foi testado com a primeira função de seleção (Apresentado na Seção 3.3.1) atingindo uma pontuação de 20905.

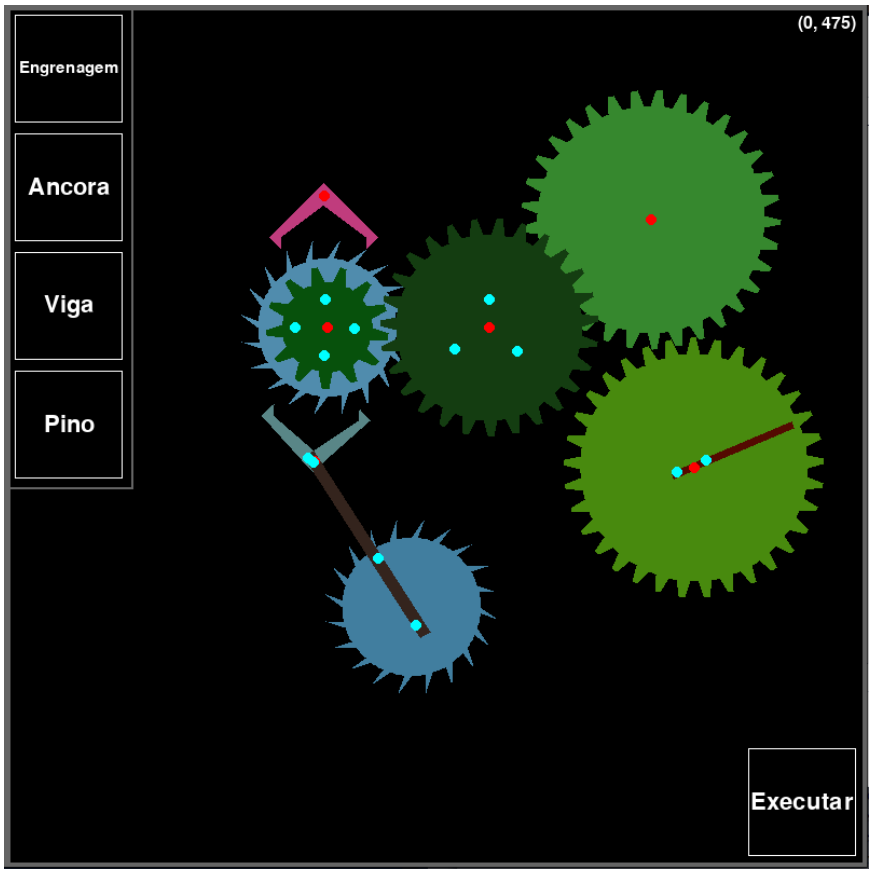


Figura 2: Relógio exemplo do que é possível ser construído com a simulação. Essa simulação atingiu pontuação 20905 para a primeira função de seleção 3.3.1.

Os comandos disponíveis nesse simulador são:

- Clicar em um espaço vazio com uma peça selecionada. Se for possível de adiciona-la nesse local com todos seus parâmetros, ela é inserida no espaço *pymunk* na posição selecionada. Caso haja colisões com outras peças no mesmo plano, ou, para o caso dos pinos, não esteja conectando duas peças ou uma peça e o fundo, o pino não é inserido. Se uma peça não puder ser inserida, ela continua selecionada. Se puder, a seleção passa a ser *None*.

- Clicar com o mouse sobre alguma peça da simulação já existente. Dessa forma a peça colocada primeiro, que cobre essa região, é removida do espaço e fica selecionada com o mouse.

- Clicar com o mouse sobre algum botão. Isso realizará a ação do botão, criando uma viga, uma âncora, um pino, uma engrenagem, ou ainda iniciando a simulação. Se uma peça for criada ela fica selecionada pelo mouse, sendo desenhada sobre ele.

- Se a roda do mouse for alterada para cima ou para baixo, o ângulo da peça selecionada irá aumentar ou diminuir no sentido horário.

- Mover o mouse com uma peça selecionada, altera a posição dela para a do mouse.

- Pressionar *backspace* remove a seleção de uma peça que, para todos os propósitos, deixa de existir.

- A tecla "Espaço"altera o plano sobre a qual a peça selecionada será inserida.

- A tecla − diminui a escala da peça selecionada.

- A tecla + aumenta a escala da peça selecionada.

- A tecla *o* aumenta a orientação dos dentes da engrenagem seleciona para o sentido horário.

- A tecla *i* aumenta a orientação dos dentes da engrenagem seleciona para o sentido anti-horário.

- A tecla *g* ativa a grade para posicionamento das peças. Ele varia pelas escalas: 1; 10; 50; 100. Isso significa que para a grade com escala g, as peças selecionadas só podem ser colocadas em coordenadas x e y tal que $mod_g(x) = 0$ e $mod_g(y) = 0$.

O comando p está disponível durante o período de simulação. O efeito é congelar a simulação até que p seja pressionado novamente.

Na tentativa de diminuir o tempo de simulação, cada simulação foi executada em um processo diferente para maximizar o uso dos recursos computacionais.

As cores das peças são escolhidas aleatoriamente como forma de identificar a mesma peça no passar das gerações. Entretanto, há uma tendência na escolha de cores para cada tipo. As engrenagens possuem cores mais próximas ao verde, as vigas possuem cores mais próximas ao vermelho e as âncoras possuem cores mais próximas ao azul. A Figura 3 apresenta exemplos de peças disponíveis, com suas propriedades sendo variadas.

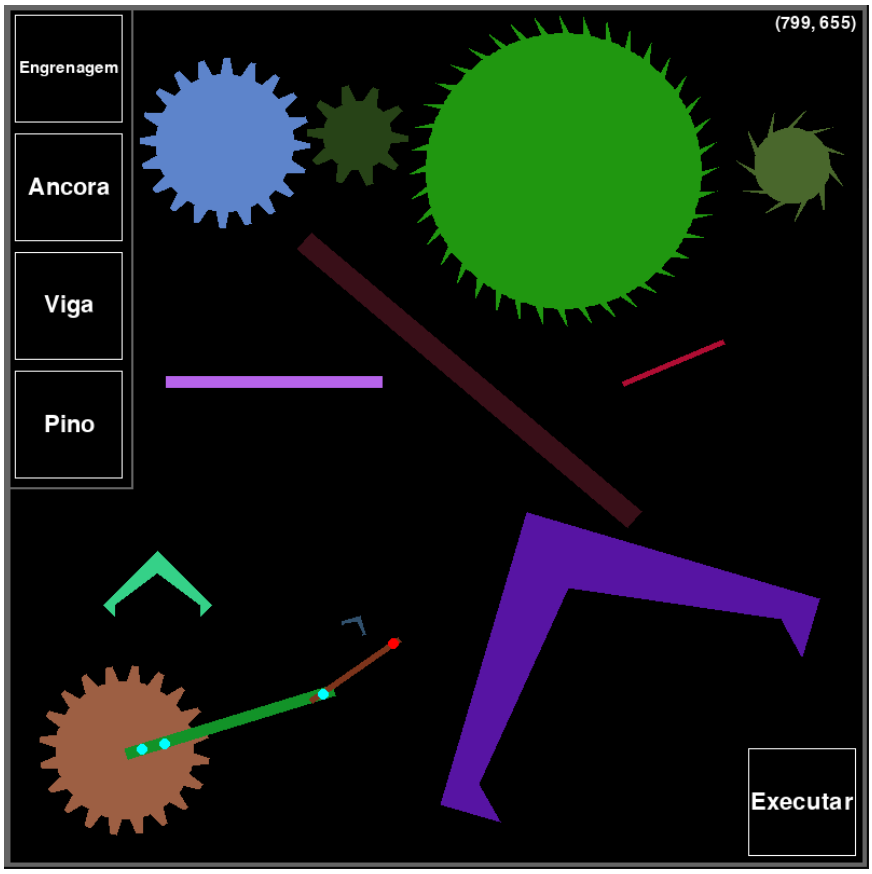


Figura 3: Exemplo de estados para cada peça do simulador

3.2 Algoritmo Evolutivo

O algoritmo evolutivo é o responsável pelo surgimento de complexidade nos indivíduos. Um indivíduo é todo o conjunto de peças com suas propriedades e conexões em uma simulação. Dessa forma, cada um dos indivíduos em uma população é testado durante o período de simulação. O algoritmo evolutivo é constituído de diversas gerações cada uma com 4 etapas.

Primeiro são executadas todas simulações dos indivíduos da geração atual. O fim de cada simulação é determinado pela sua respectiva função de seleção (Seção 3.3).

Quando a última simulação se encerra, a segunda etapa se inicia. Utilizando as pontuações de cada indivíduo computadas pela sua função de seleção (Seção 3.3), são selecionados os 3 melhores para se reproduzirem, com uma ponderação de 70% das caraterísticas do melhor indivíduo, 20% das caraterísticas do segundo melhor indivíduo e 10% das caraterísticas do terceiro melhor indivíduo. Além disso, o melhor indivíduo é reinserido na população como forma de manter suas características presentes. Essa estratégia é denominada *elitismo* e pode acarretar dificuldades na evolução. Essas dificuldades são detalhadas em breve.

A terceira etapa é a reprodução. Com as probabilidades de cada um dos 3 pais definidas,

é selecionado um deles segundo essa probabilidade (pai j) para cada peça no primeiro pai. Se essa peça existir em j, todas as suas características são copiadas para o indivíduo final.

A quarta e última etapa é a mutação. É definido um número de mutações para os indivíduos, nesse caso 1 e a taxa de mutação, nesse caso 1. O número de mutações define quantas mutações ocorrerão por indivíduo, enquanto a taxa de mutação define a intensidade dessa mutação. Para realizar a mutação, primeiro é escolhido o parâmetro a ser alterado dentre as seguintes opções: {"posição", "ângulo", "escala", "orientação", "parede", "plano"}. Cada um deles possui restrições sobre valores máximos, quais objetos podem ser aplicados e como se relacionam com a taxa mutagênica. Os atributos são detalhados a seguir:

- O atributo "*posição*"é aplicado a todos os tipos de peças. A sua interação com a taxa mutagênica é dada pela atribuição: $pos_i := pos_i \cdot random.uniform(1 - 100t, 1 + 100t)$, em que pos_i é um dos valores do par x, y da posição, t é a taxa mutagênica e $random.uniform(a, b)$ é uma função de gerador de números aleatórios segundo a distribuição uniforme limitada ao intervalo $[a, b)$. Após essa atribuição, o valor é limitado ao intervalo $[50, 750]$.

- O atributo "*escala*"é aplicado a todos os tipos de peça com exceção dos pinos. A sua interação com a taxa mutagênica é dada por: $escala := escala \cdot random.uniform(1 - t, 1 + t)$ limitados para o intervalo $[0.5, 2]$.

- O atributo "*angulo*"é aplicado a todos os tipos de peça com exceção dos pinos. A sua interação com a taxa mutagênica é dada por: $escala := escala \cdot random.uniform(1 - t, 1 + t)$ limitados para o intervalo $[0, 360]$.

- O atributo "*orientation*"é aplicado apenas a engrenagens. A sua interação com a taxa mutagênica é dada por: $orientation := orientation \cdot random.uniform(1 - t, 1 + t)$ limitados para o intervalo $[-0.6, 0.6]$. Ele modifica o nível de orientação dos dentes das engrenagens.

- O atributo "*parede*"é aplicado apenas a pinos. Uma mutação ocorrida nesse parâmetro troca o seu valor booleano, equivalente a porta *not*.

- O atributo "*categoria*"é aplicado a todos tipos de peça com exceção dos pinos. Uma mutação ocorrida nesse parâmetro troca o seu valor entre 1, 2. Equivalente a seguinte equação: $categoria := 3 - categoria$. Esse parâmetro representa o plano no qual as peças estão localizadas.

Há diversas formas de definir a população inicial dos indivíduos. Nesse trabalho, a população inicial começa com 0 peças cada uma. Todo novo indivíduo durante a fase de cruzamento, sorteia uma chance para *adicionar* e uma para *remover* peças. Para remover, é necessário que, sorteado um valor R do intervalo $[0, 50]$, satisfaça $R > 50 - n$ em que n é o número de peças atuais nesse indivíduo. Para adicionar, é necessário que, sorteado um valor A do intervalo $[0, 50]$, satisfaça $A < 50 - n$ em que n é o número de peças atuais nesse indivíduo. As operações são executadas na ordem em que foram descritas. Essa função

possibilita que indivíduos variem os números de peças que possuem, mas adiciona uma pressão em torno de cada um para possuir 25 peças.

Um dos mecanismos fundamentais da evolução em seres vivos é a *deriva genética*. Ele consiste em diversas mutações acumuladas em uma população que não garantam a ela uma vantagem evolutiva sobre as demais. Em dado momento, uma nova mutação pode surgir e se beneficiar do conjunto acumulado para garantir uma vantagem no futuro. O *elitismo* atrapalha esse mecanismo por garantir que o melhor indivíduo continuará inalterado na população, funcionando como uma "âncora"para a população. Para resolver esse problema, foi observado que muitos indivíduos diferentes na população possuíam a mesma pontuação. Dessa forma, para garantir a presença da deriva genética, basta embaralhar os indivíduos e utilizar essa ordem como critério desempate para seleção dos 3 melhores indivíduos. Assim, as características vantajosas não são perdidas, mas libera as outras características para evoluírem.

O número de gerações é grande o suficiente para que a simulação dure por 1 semana. No geral é escolhido um milhão de gerações.

3.3 Funções de Seleção

Uma das 3 propriedades necessárias para a evolução é a seleção. A hipótese inicial desse trabalho é a de que "é possível evoluir um relógio mecânico sem a presença de um projetista". Dessa forma, é desejável que a função de seleção seja a mais branda possível para não tendenciar o relógio evoluído transparecendo a presença de um projetista. Foram testadas diversas funções de seleção, aumentando gradualmente a complexidade de cada uma delas. Elas são detalhadas a seguir.

3.3.1 Função de seleção 1: Repetição global

A primeira função de seleção é baseada na repetição global das peças. Um bom relógio humano possui 2 propriedades desejáveis: A precisão e a amplitude no número de estados. A precisão é garantida por se tratar de um ambiente computacional determinístico. Assim, quando todas as peças passarem por um exato estado que já percorreram antes, todos os estados posteriores serão iguais aos já visitados. Aqui, um estado é o conjunto das propriedades "posição", "velocidade linear", "ângulo", "velocidade angular"para cada peça. Fornecido os objetos e essas propriedades para cada um deles, é possível reproduzir completamente uma simulação. Já a amplitude do número de estados pode variar entre indivíduos diferentes. Essa é a primeira função de seleção: "O número de estados distintos que uma simulação percorre até atingir um estado posteriormente visitado. Em termos computacionais, o algoritmo percorre os estados em ordem e para ao encontrar um ciclo.

Para implementar esse algoritmo, é utilizado um dicionário E, representando os estados da simulação. Para cada novo estado é computado o seu Hash. O conjunto de todos os objetos é então adicionado em E com a chave sendo o Hash computado. Se em algum momento já existir um elemento ocupando essa posição no dicionário, então a simulação está repetindo os estados e é encerrada, com a pontuação sendo o número de estados até esse momento. O Hash é computado com o modo 0, detalhado na subseção 3.4.

3.3.2 Função de seleção 1: Repetição local

A segunda função de seleção tenta resolver o problema do tempo de simulação de relógios evoluídos. Após algumas gerações, o tempo de simulação de uma geração torna-se muito elevado. Para resolver esse problema, foi testado uma outra função de seleção e de parada da simulação. Ao invés de contabilizar um estado somo sendo todas as peças de um indivíduo, contamos os estados de cada peça individualmente. Todas as peças começam sendo consideradas que não repetiram. É computado o hash modificado para receber apenas uma peça ao invés do indivíduo como um todo. Cada peça possui um dicionário de hashs presenciados. Se uma peça percorrer um estado de um hash já visitado, é considerado que essa peça já repetiu alguma vez. A simulação acaba quando todas as peças tiverem repetido pelo menos uma vez. Quando isso ocorrer, a pontuação do indivíduo será o número de estados percorridos na simulação até ela acabar. A implementação dessa abordagem é semelhante a anterior, mas com um dicionário externo englobando todas as peças.

3.3.3 Função de seleção 3: Repetição frequente

A terceira função testada resolve uma falha nas premissas estabelecidas na seção 3.3.1. Ao finalizar uma simulação quando a última peça repete seu estado, não é garantido que o estado global do relógio está se repetindo, ou ainda, não é possível prever o próximo estado dessa peça, podendo nunca voltar para esse último estado novamente. Isso ocorre por não levar em consideração todas as outras peças que podem colidir e interagir com ela. Dessa forma, não é garantida a precisão dos relógios. Por conta disso, é adicionado na função de seleção, uma pontuação dependente dos períodos da repetição de cada peça. Aqui, a simulação acaba quando a última peça se repetir um número $numero\_max\_rep$ de vezes. Ou seja, são anotados para cada hash de cada peça, todos os tempos de simulação que a peça o encontrou. Para os experimentos desse trabalho, $numero\_max\_rep = 6$. A pontuação aqui é o máximo das pontuações de cada peça. A pontuação de cada peça tem o objetivo de premia-la por se manter na mesma frequência de oscilação. Dado os tempos do hash que possui mais anotações de tempo, é removido, em ordem, amostras que estejam 1 de distância da anotação anterior. Por exemplo, dado as anotações $1, 2, 3, 4, 6, 7$, o resultado final após a remoção é $1, 3, 6$. Após isso, percorrendo todos os elementos a partir da posição 2 e começando $pontuacao$ com 0, tempos que $intervalo\_atual = anot[i] - anot[i-1]$, e $intervalo\_passado = anot[i-1] - anot[i-2]$ e por fim $pontuacao := pontuacao + max(intervalo\_passado - |intervalo\_passado - intervalo\_atual|)$. Aqui $anot$ é a lista de anotações temporais do hash atual. Essa expressão está computando a diferença das variações nos tempos mais frequentes (aceleração atual) e incrementando a pontuação em velocidade atual e essa aceleração computada. Assim, é esperado que os indivíduos evoluam sob pressão seletiva para manterem uma frequência constante em suas oscilações, mas oscilações muito longas. Aqui estamos buscando relógios que levam muito tempo para dar uma volta e que mantém essa volta a todo momento. Há ainda uma outra consequência para essa função. Se o indivíduo atrasar constantemente o seu relógio, ou seja, a aceleração é constante, o indivíduo é beneficiado. Entretanto esse ainda seria um relógio aceitável para essa evolução.

3.3.4 Função de seleção 4: Repetição paciente

A mudança da função de repetição frequente (subseção 3.3.3) para a repetição paciente, é a paciência de 100 instantes de tempo. A todo momento que alguma peça acessar um estado novo, a paciência é alterada para 100. A cada instante de tempo que nenhuma peça assuma um novo estado, a paciência diminui em 1. Caso ela chegue em 0, a simulação acaba. A pontuação começa em 0 e sempre que uma peça visitar um estado já visitado anteriormente, ela aumenta em 1. Isso força que peças ou fiquem paradas, ou ciclem para ganhar pontos, mas para aumentar o tempo da simulação é necessário que alguma peça do indivíduo encontre frequentemente novos estados, garantindo uma pressão seletiva para peças se movendo. A implementação da verificação dos estados por peça é o mesmo da função de seleção "Repetição local" 3.3.2 e "Repetição frequente" 3.3.3.

3.3.5 Função de seleção 5: Repetição colisão

A última função de seleção testada, foi um acréscimo sobre a "Repetição paciente" 3.3.4. Agora colisões entre as peças são consideradas peças somam 1 nos pontos do indivíduo. Isso cria uma pressão seletiva para que peças colidam mais frequentemente para ganhar pontos, mas ainda mantém as pressões da "Repetição paciente".

3.4 Hash

Durante a etapa de simulação com diferentes funções de seleção, é utilizado um hash para resumir em um único número, o indivíduo por inteiro ou suas partes. Para isso é utilizada uma função com perdas, mas experimentalmente muito eficaz. Como há diferentes necessidades para as diferentes funções de seleção, os hashs possuem modos, especificando seus comportamentos. Entretanto, todos eles utilizam 3 etapas: a **aproximação**, a **limitação** e a **escala**.

A aproximação é a parte mais importante da função de hash. Apesar de computacional, os elementos na simulação possuem valores racionais de precisão limitada. Entretanto, os quadros da simulação possuem uma taxa de amostragem muito baixa. Dessa forma, para um movimento circular simples, devido a taxa de amostragem, é possível passar muitas rotações antes de ser detectada a repetição. A discretização ou aproximação resolve esse problema. Assumindo uma velocidade máxima, um objeto em movimento de forma cíclica vai passar em pelo menos uma casa de discretização se os intervalos forem construídos corretamente. Cada modo possui suas constantes de aproximação. O modo 0 possui: {"rotation": 0.1, "position": 0.1, "linear_velocity": 5, "angular_velocity": 0.1 }, enquanto o modo 1 possui: {"rotation": 0.1, "position": 0.1, "linear_velocity": 1, "angular_velocity": 0.1 }, e por fim o modo 2: {"rotation": 0.5, "position": 0.05, "linear_velocity": 0, "angular_velocity": 0 }. Para computar os valores para um parâmetro é utilizado o seguinte procedimento: $valor := round(valor/aproximador) * aproximador$, em que *valor* é a entrada a ser aproximada e *aproximador* é a constante aproximativa da respectiva propriedade do objeto. Se o valor de *aproximador* for 0, valor é definido como 0 ($valor := 0$). Se valor é bidimensional, o processo é realizado com cada uma das coordenadas.

A limitação é importante para impedir extrapolações muito grandes de velocidade e casos extremos de posição. Se alguma peça conseguir de alguma forma escapar da região de simulação, a posição dela será limitada antes de computar a hash. Dessa forma os indivíduos não são beneficiados por conseguirem escapar de região de simulação. Além disso, também é utilizado a limitação de velocidade máxima razoavelmente baixa. Isso ajuda a diminuir o número de hashs possíveis e facilitar encontrar um estado já percorrido. os valores para limitações são para todos os modos de hash: {"rotation": 360, "position": 1000, "linear_velocity": 6, "angular_velocity": 360 }. Se valor é bidimensional, o processo é realizado com cada uma das coordenadas. A expressão utilizada para definir o valor é: $valor := max(min(valor, limitardor), -limitador)$, com limitador sendo a constante definida pelo tipo de objeto, min e max são as funções nativas para encontrar o menor valor e o maior valor entre os argumentos respectivamente.

A escala possui uma função mais técnica sobre o hash final. Existe um número finito de estados possíveis dada as limitações e as discretizações. Além disso, seria bom se os hashs não resultassem em colisões mas ocupassem o menor espaço de memória possível. Por conta disso, para cada propriedade dos objetos disponíveis é multiplicado os valores atuais por uma constante dependente do modo e da propriedade: para o modo 0 as escalas são {"rotation": 10^5, "position": 10^7, "linear_velocity": 10^2, "angular_velocity": 10^2 }, para o modo 1: {"rotation": 10^4, "position": 10^6, "linear_velocity": 10^2, "angular_velocity": 1 } e para o modo 2: {"rotation": 10^5, "position": 10^6, "linear_velocity": 0, "angular_velocity": 0 }.

Após computar as alterações de aproximação, escala e limitação para cada peça do indivíduo, o valor final de cada uma é agregado ao valor do hash através da operação xor. O hash inicialmente assume 0. Se a propriedade a ser agregada for bidimensional, é somado os valores inteiros de cada uma das duas propriedades.

4 Resultados encontrados

Respeitando as restrições estabelecidas na seção de Objetivos 2, todas as simulações executaram por no máximo uma semana, cada um gerando uma quantidade de gerações diferentes com base no custo dessa função de seleção. Pelo menos um resultado relevante foi separado de cada função de seleção e são mostrados a seguir.

4.1 Função de Seleção: Repetição global

Em um primeiro momento, a simulação foi executada sem restrições na localização das peças para que elas não colidissem entre si ou com a borda. Dessa forma, frequentemente peças eram colocadas sobrepostas, o que faziam com que elas entrassem em um estado impossível e começassem a vibrar rapidamente para resolver as colisões. Como elas são construídas por vários polígonos menores como detalhado na seção sobre o ambiente de simulação 3.1, quando um polígono interno escapa da colisão, outro polígono passa a colidir, fazendo a peça como um todo vibrar e ficar presa. Isso causa pequenas mudanças de posição, mas uma grande mudança na velocidade, o que no próximo momento, altera a posição da peça de maneira considerável. Esse comportamento cria relógios caóticos e não está alinhado

com relógios reais. A figura 4 mostra um caso em que uma engrenagem fica presa no topo da simulação.

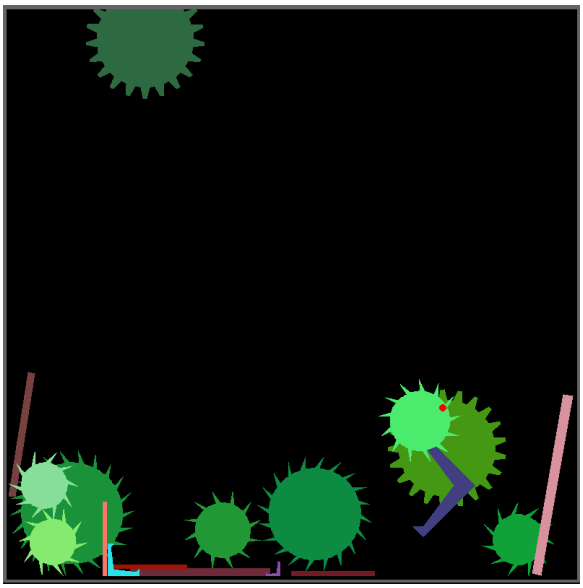


Figura 4: Esse é o resultado da simulação com função de seleção de repetição global com um erro. A engrenagem fica preso na borda superior da simulação e vai lentamente se movendo para direita. Isso faz com que a simulação dure muito tempo.

Para resolver esse problema, durante o cruzamento e criação das peças, é verificado se elas colidem com algo já presente no ambiente, como outra peça já inserida ou a borda. Se isso ocorrer, a mutação inserida sobre essa alguma peça causou um estado inválido, logo o indivíduo criado é destruído e um novo é criado em seu lugar com uma nova mutação e cruzamentos. Isso garante que, se os pais forem válidos, os filhos também serão. Como os pais começam com nenhuma peça, os pais sempre são válidos. Caso não seja possível adicionar uma nova peça, os indivíduos que tentarem, não poderão sobreviver, permitindo apenas a existência de indivíduos que não adicionaram novas peças. Assim, sempre é possível gerar um novo indivíduo válido diferente do pai, já que é possível remover peças. Isso faz com que a evolução não fique travada e possa continuar.

Com a adição dessa restrição na reprodução, foi possível evoluir a primeira linha de indivíduos sob a função seletiva de “Repetição global”. O último indivíduo após o tempo de treinamento adequado é apresentado na Figura 5. Ele foi gerado com 75 gerações e 100 indivíduos por geração.

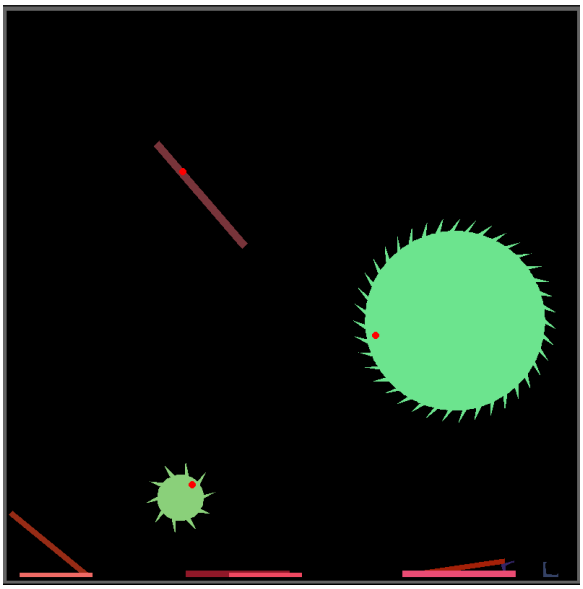


Figura 5: Esse é o resultado da simulação com função de seleção de repetição global. Há 3 pêndulos: a viga vermelha e as duas engrenagens verdes presas ao fundo com pinos vermelhos. Uma engrenagem é muito grande, outra pequena e a viga tem tamanho intermediário, fazendo com que os tempos de oscilações sejam distintos.

O resultado final dessa função de seleção foram 3 pêndulos de tamanhos distintos oscilando. Isso ocorreu pois essa distribuição das peças cria muitos estados antes de repetir completamente o estado da simulação. Como cada pêndulo possui tamanhos de peças diferentes e a gravidade é a mesma para todos, o tempo de oscilação difere entre eles. Dessa forma, assuma que o período de oscilação de cada um dos três pêndulos é respectivamente: a, b, e c. Assim, o tempo de oscilação do conjunto como um todo é de $MMC(a, b, c)$, ou seja, o mínimo múltiplo comum entre todos os períodos de oscilação individuais. Se eles forem coprimos, o período total é a multiplicação entre eles. Isso faz com que o tempo de simulação fique muito prolongado, principalmente quando simulados diversos indivíduos com peças não utilizadas. Por conta disso poucas gerações foram simuladas.

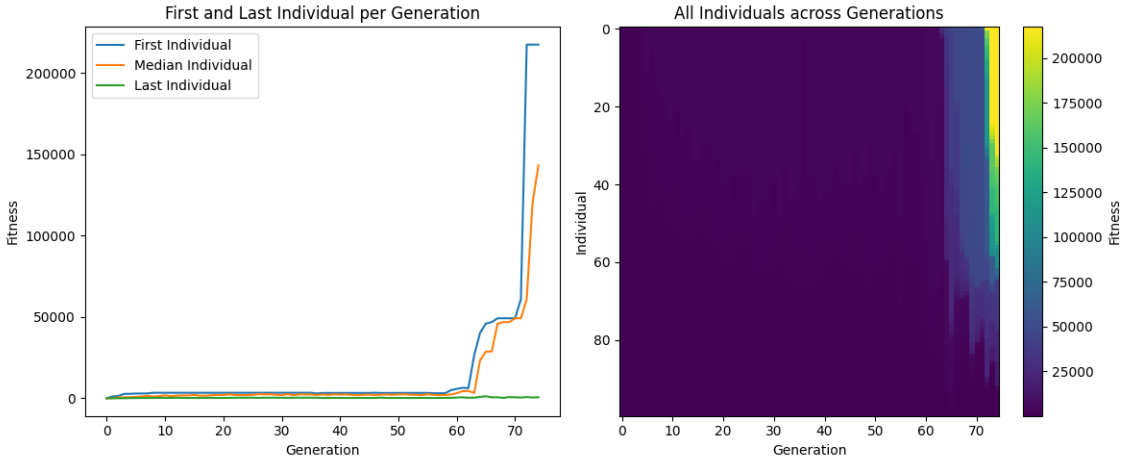


Figura 6: Esse é o resultado numérico da simulação com função de seleção de repetição global.

Os gráficos na Figura 6 apresentam as recompensas dos indivíduos durante as gerações. A esquerda temos 3 tipos de indivíduos em especial, os indivíduo com maior pontuação em cada geração (em azul), os indivíduos que representam a mediana nos pontos de cada geração (em laranja), e os indivíduos com a pior pontuação em cada geração (em verde). Podemos verificar um grande período com nenhuma mudança considerável. Próximo à geração 60 há uma subida sutil na recompensa dos indivíduos. Esse é o momento do surgimento do primeiro pêndulo. Ele foi conectado à pequena viga no topo, o que garantiu uma vantagem, mas por se tratar de uma viga muito pequena, o período dela não agregou muito mais que o período das engrenagens rolando nas demais partes da simulação. Com o surgimento do segundo pêndulo (a engrenagem mais abaixo), que ocorreu na geração 63, pudemos identificar uma melhora considerável com relação ao restante da simulação. Entretanto, na geração 71 ocorreu o surgimento do terceiro pêndulo, sendo a maior engrenagem. Com ela a simulação passaria a repetir muito mais, o que é verificado no gráfico. Esse gráfico, porém, mostra o desempenho apenas de 3 indivíduos por geração.

Para poder apresentar todos de modo geral, é utilizado o gráfico a direita. Nele é mostrado de cima para baixo os indivíduos de cada geração do melhor para o pior e da esquerda para direita as gerações. As cores de cada quadrado são as pontuações desses indivíduos. Esse gráfico nos possibilita adquirir uma intuição sobre o que está acontecendo com todos os outros indivíduos. Podemos verificar que logo antes de uma nova mancha, há um pequeno quadrado isolado, mostrando a dificuldade no surgimento de uma nova característica benéfica surgir. Podemos verificar também que logo após esse ponto solitário, há um grande trecho vertical, apresentando a rápida alteração da população para os indivíduos com essa característica. Apesar disso, ainda há muitos descendentes com uma pontuação muito maior. Isso ocorre por conta da chance de uma mutação ser maléfica ser muito maior que ela ser benéfica, assim, a maior parte das mutações acabam por destruir os pêndulos bem formados. Nesse gráfico há um caso em que não foi identificado apenas um ponto isolado, mas dois ao mesmo tempo com uma pontuação maior. Esse caso é a adição do

segundo pêndulo. Possivelmente o que ocorreu foi a criação de um outro pêndulo menor, ao mesmo tempo que esse foi criado. Por possuírem uma vantagem menor, foram superados pelo melhor pêndulo. Como a reprodução é feita com três indivíduos, as características deles não são totalmente perdidas e possivelmente contribuíram para o surgimento do terceiro pêndulo.

4.2 Função de Seleção: Repetição local

A função de seleção por repetição local gerou o indivíduo final apresentado na Figura 7. Esse indivíduo foi coletado da geração 425, o que indica que a evolução possuiu muito mais tempo para trabalhar, mas também que a simulação é muito mais leve, possibilitando para o mesmo tempo, mais gerações.

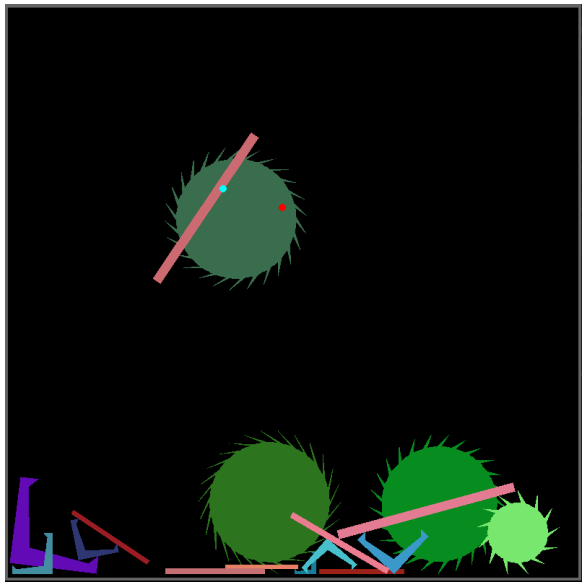


Figura 7: Esse é o resultado da simulação com função de seleção "Repetição local". Diversas peças são não utilizadas, mas duas são presas a parede. A engrenagem é presa diretamente através de um pino vermelho. Já a viga rosa é presa a engrenagem com um pino azul. Essa composição define um pêndulo duplo, com a viga rosa oscilando de forma muito difícil de prever.

O melhor indivíduo nessa linha evolutiva convergiu para um pêndulo duplo. Isso ocorre pois essa foi a estrutura mais simples encontrada que fornece diversos estados para uma peça antes de repetir qualquer um deles. O pêndulo duplo é um problema conhecido na física [3] por ser difícil de descrever, dependendo muito das variáveis iniciais. A viga conectada na engrenagem passa por muitos estados distintos, garantindo uma grande pontuação para esse indivíduo.

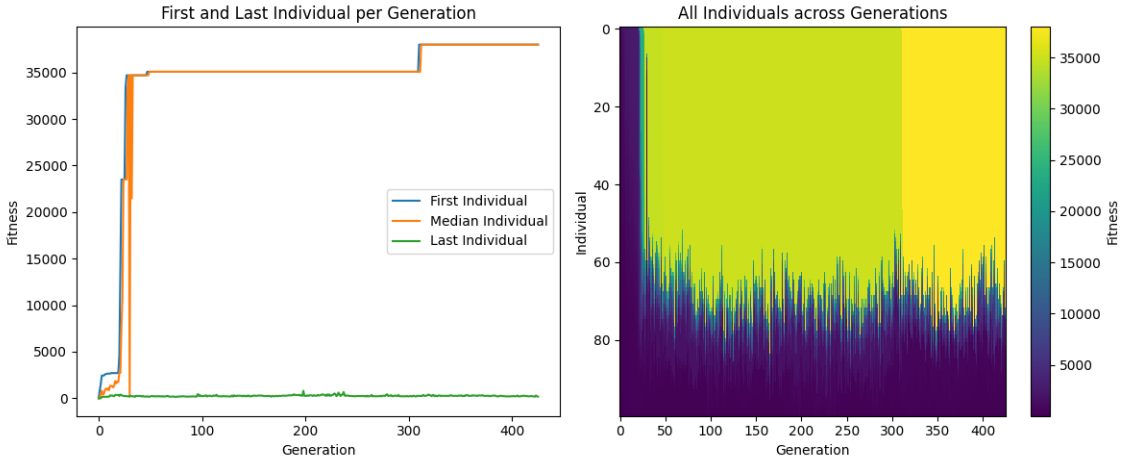


Figura 8: Esse é o resultado numérico da simulação com função de seleção de "Repetição Local".

Os gráficos da Figura 8 apresentam os dados da mesma forma que os da Figura 6. A esquerda verificamos que diferente do primeiro caso, o primeiro pêndulo foi criado muito cedo (na geração 19), com o pêndulo duplo já existindo na geração 40. Entretanto, essa estrutura é muito mais instável a mutações, já que qualquer mutação maléfica sobre o primeiro pêndulo, destrói o pêndulo duplo completo. Por conta disso, em alguns pontos, até os indivíduos medianos possuíram uma pontuação muito abaixo do ótimo, apresentando essa instabilidade. Depois da época 300 em ambos gráficos é possível identificar uma melhora no pêndulo. Isso se deve a uma posição inicial mais favorável, ou seja, que faz a viga conectada percorrer mais estados antes de repetir. Essa alteração ocorre alterando gradativamente uma peça por geração, o que limita o espaço de busca. Isso acontece pois qualquer alteração sobre essas peças modifica a pontuação final, sendo uma região muito sensível e é aplicado apenas uma mutação por vez. Se a maximização precisar de mais de uma mutação por vez, ela nunca será atingida. O gráfico da direita possui o mesmo comportamento de um trecho vertical estar seguindo um indivíduo ótimo isolado, mas por ser mais sensível, os picos de piores indivíduos são muito menos padronizados. Entretanto, por possuir menos peças que a estrutura da função de seleção anterior, o número de mutações prejudiciais como um todo é menor, já que modificar peças fora do pêndulo, não modifica a pontuação final. Assim, a região em amarelo é maior que o caso anterior.

4.3 Função de Seleção Repetição frequente

Esse foi o resultado mais diferente em relação aos demais. Como mostra a Figura 9, o relógio consiste na adição de uma engrenagem com movimento livre, sobre uma âncora de ponta cabeça presa pela extremidade por dois pinos vermelhos (conectados a parede). Essa construção força a âncora a permanecer parada, entretanto, por estar posicionada na extremidade e com os pinos muito próximos, é garantida alguma mobilidade angular a essa peça. Assim, quando a engrenagem cai sobre ela, é garantido momento, fazendo-

a rotacionar. Os pinos impedem que isso ocorra e voltam a posição inicial, lançando a engrenagem para cima. Esse foi o melhor indivíduo da geração 607, mostrando que é igualmente rápido em relação à função de seleção anterior.

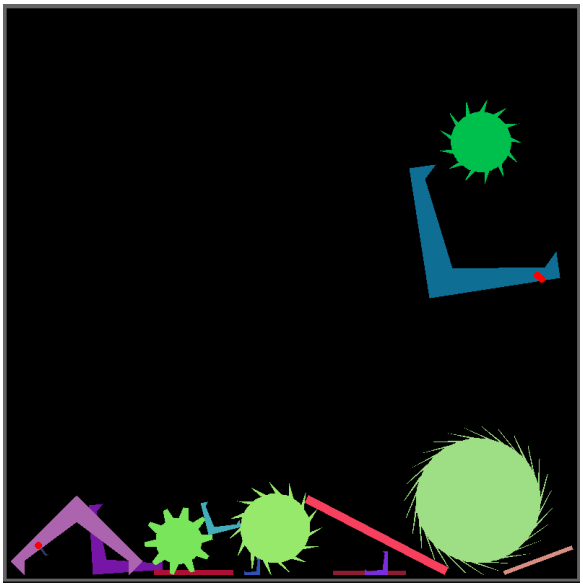


Figura 9: Esse é o resultado da simulação de seleção "Repetição frequente". A evolução levou esse indivíduo a posicionar uma âncora como um braço preso ao fundo com dois pinos vermelhos e uma engrenagem acima. A engrenagem cai sobre o braço e ele a arremessa para cima, colidindo com o teto e a parede lateral da simulação.

Esse indivíduo encontrado possibilita que a engrenagem que está sendo jogada passa por muitos estados antes de se repetir, pois não é jogada na mesma direção todas as vezes. Após alguns lançamentos, o braço perde energia e começa a lançar a engrenagem cada vez menos, até que uma pequena oscilação contínua ocorre. Nesse momento, após 6 oscilações a simulação termina. Apesar de satisfazer as restrições, esse é ainda mais diferente do relógio real que as outras linhas evolutivas.

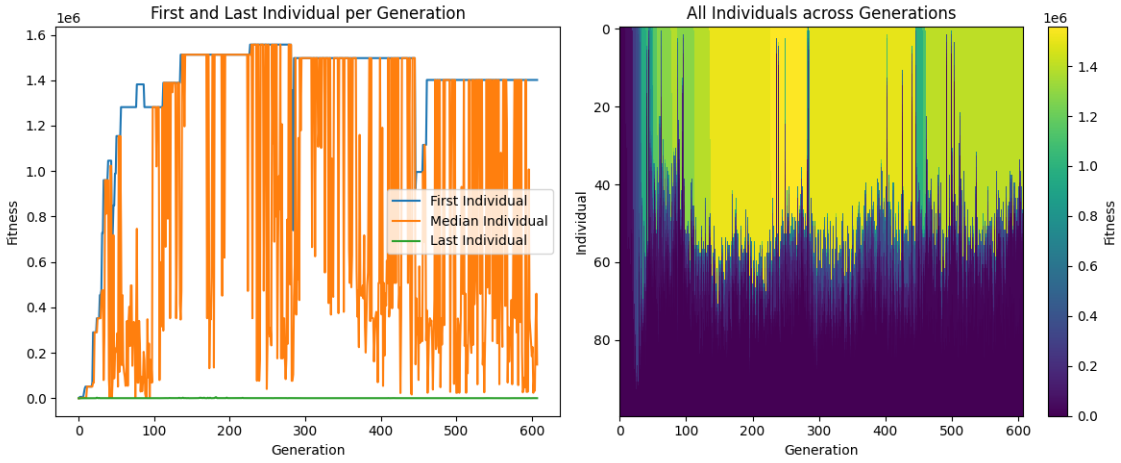


Figura 10: Esse é o resultado numérico da simulação com função de seleção de "Repetição Frequente".

Os gráficos da Figura 10 apresentam o caminho evolutivo das gerações. Aqui, a solução encontrada é ainda mais instável que a anterior. Na geração 19 surge o primeiro pêndulo e pouco tempo depois é adicionado um outro pino vermelho a esse pêndulo feito por uma âncora. Acima dela já estava uma engrenagem que quando cai sobre essa estrutura de âncora, é jogada para cima com velocidade. Muito cedo foi encontrada uma solução que oscila muito, percorre muitos estados e é extremamente instável. Esse cenário é tão instável que mesmo com elitismo, houve momentos visíveis no gráfico da esquerda, que a pontuação máxima decresceu. Isso evidencia algum erro sobre o determinismo do ambiente, uma vez que o mesmo indivíduo gerou pontuações diferentes. Mesmo após procura dessa diferença, não foi possível compreender a causa. Podemos ver em ambos gráficos esquerda e direita, que o cenário é muito instável. Além disso, na maior parte do tempo, a simulação está otimizando a posição inicial da engrenagem, encontrando um estado inicial na qual ela é solta para faze-la percorrer o maior número de estados de forma periódica. Qualquer pequeno movimento nesse sistema caótico pode levar a resultados muito diferentes.

4.4 Função de Seleção repetição paciente

O indivíduo 11 é o representante da geração 4814. Especialmente para esse método foi fornecido mais tempo, uma vez que ele possui o mecanismo de paciência. Surpreendentemente as gerações acompanharam o acréscimo, chegando ao maior número de gerações. Por outro lado, o resultado não foi diferente dos anteriores.

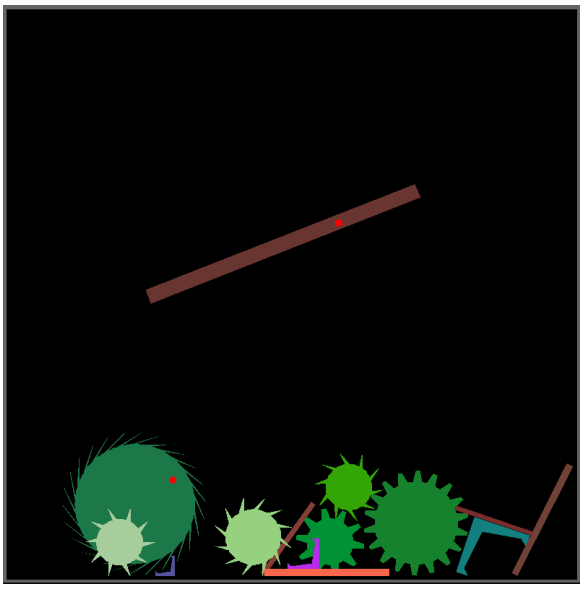


Figura 11: Esse é o resultado da simulação repetição paciente. Aqui verificamos a existência de dois pêndulos: uma engrenagem travada no chão e uma viga livre ao topo. Outras peças livres estão no chão da simulação.

Esse resultado mostra apenas um pêndulo. Possivelmente isso ocorre pela amostragem do pêndulo oscilando. Quando o pêndulo percorre uma posição repetida pela primeira vez, ele deveria continuar repetindo todos os estados a partir desse momento, mas por conta da pequena perda de energia, gradualmente ele desacelera, levando a viga a visitar um novo estado. Quando isso ocorrer a contagem reinicia, estendendo a simulação e ganhando mais pontos por isso.

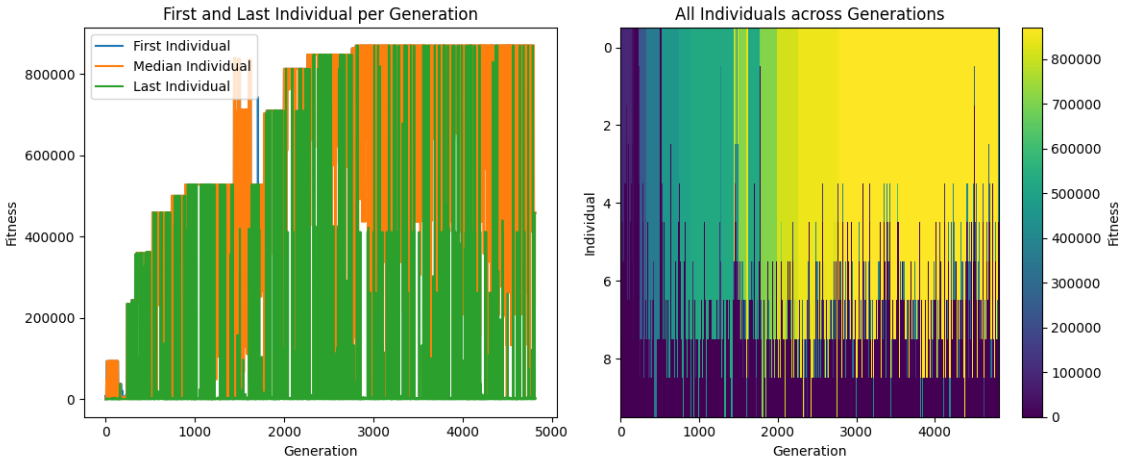


Figura 12: Esse é o resultado numérico da simulação com função de seleção de “Repetição Paciente”.

Os resultados apresentados na Figura 12 mostram um cenário em que pela primeira vez os piores indivíduos conseguem alcançar pontuações semelhantes aos melhores. Isso ocorreu pois o resultado encontrado um único pêndulo. Assim, as piores pontuações são ruins quando destroem o pêndulo, ou indiferentes quando ocorrem fora dele. A adição de uma outra estrutura rotacionando (o pêndulo feito pela engrenagem travado no chão), não auxilia no processo de aumentar o tempo da simulação, já que ele é completamente feito pelo pêndulo, e mantido conforme ele perde energia. Os gráficos mostram que a única otimização feita nesse cenário foi a posição inicial do pêndulo, alterada pouco a pouco para percorrer mais estados novos dentro do intervalo de paciência de 100 instantes. Por possuir muitas gerações, o gráfico muito cheio. Para resolver isso, foi utilizado uma média móvel de 50 amostras como apresentado na Figura 13.

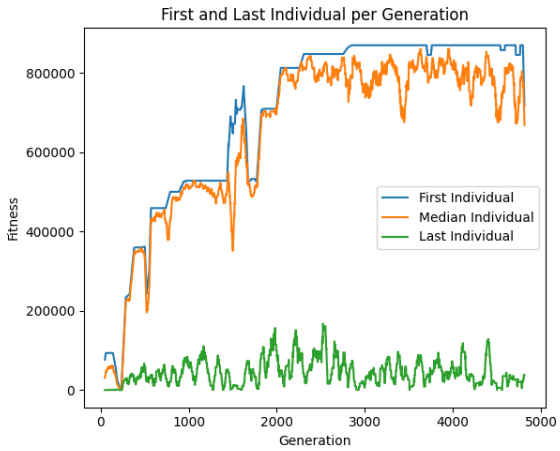


Figura 13: Esse é o resultado numérico da simulação com função de seleção de "Repetição Paciente"com uma média móvel de 50 amostras.

A Figura 13 apresenta o mesmo problema encontrado na simulação da função de perda "Repetição Frequente": Uma pontuação máxima que caiu em certo momento. Além disso é possível verificar que nesse caso os piores indivíduos estão melhores que os outros métodos, enquanto a mediana está muito próxima ao ótimo.

4.5 Função de Seleção Repetição colisão

O indivíduo apresentado na Figura 14 é o representante da geração 14 da função de seleção "Repetição colisão".

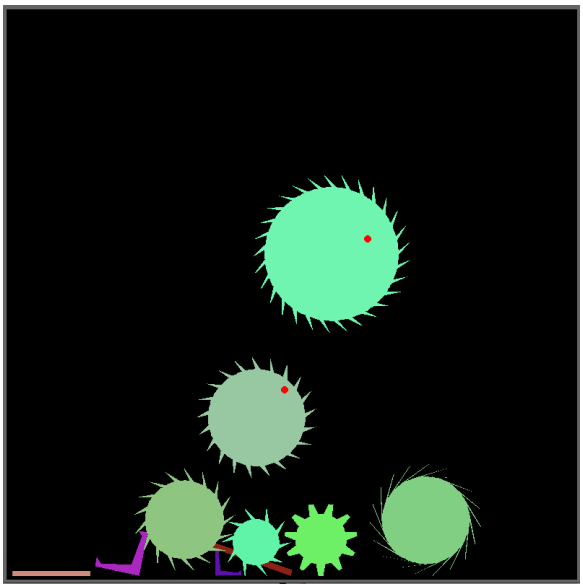


Figura 14: Esse é o resultado da simulação com função de seleção "Repetição colisão". É possível verificar que foram criados dois pêndulos oscilantes.

Aqui podemos verificar que a parte na pontuação que considera a colisão não foi relevante o suficiente. Além disso, o número de gerações está muito abaixo do esperado. Possivelmente isso ocorreu por conta de algum conjunto de gerações que muito cedo geraram uma simulação muito lenta, mas também um erro na estrutura computacional no momento da execução, o que diminuiu o processamento para esse processo e levou a simulações mais lentas.

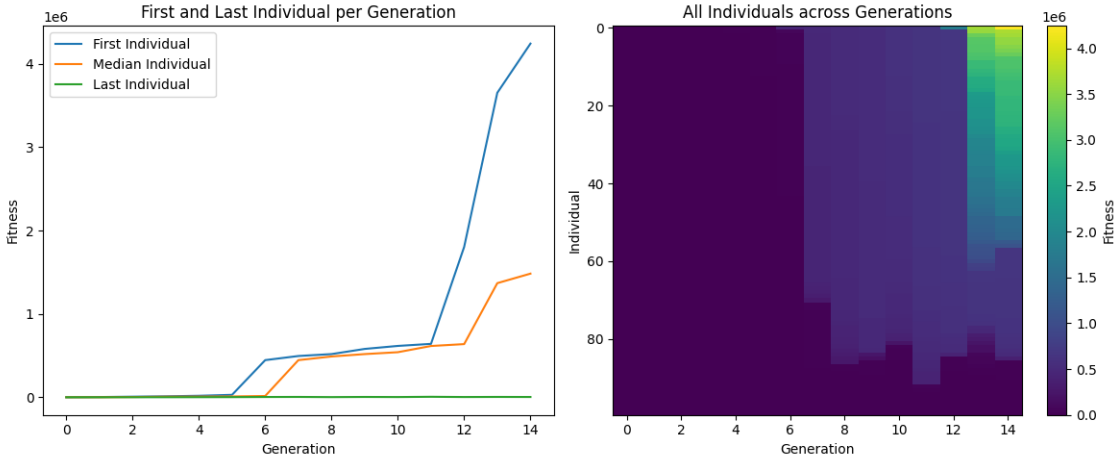


Figura 15: Esse é o resultado numérico da simulação com função de seleção de "Repetição Frequente".

A Figura 15 mostra o desempenho por gerações, dessa linha evolutiva. O gráfico a

esquerda mostra mais claramente o surgimentos dos pêndulos nessa simulação. O primeiro surgiu na geração 5 e suas características são passadas para os demais logo na próxima geração, também sendo possível de ver esse comportamento no gráfico da direita. Após otimizar o primeiro pêndulo, na geração 12 surge o segundo pêndulo, e suas características são passadas adiante logo em seguida.

4.6 Diminuir a população

Foi testado para a terceira função de seleção uma população de 10 indivíduos. O resultado após 3374 gerações é apresentado na figura 16. Devido a quantidade de indivíduos, foi possível percorrer mais gerações. Entretanto, devido ao elitismo e o baixo número de mutações nos indivíduos, a evolução não encontrou uma forma de evoluir, permanecendo em varias peças desconexas pelo espaço.

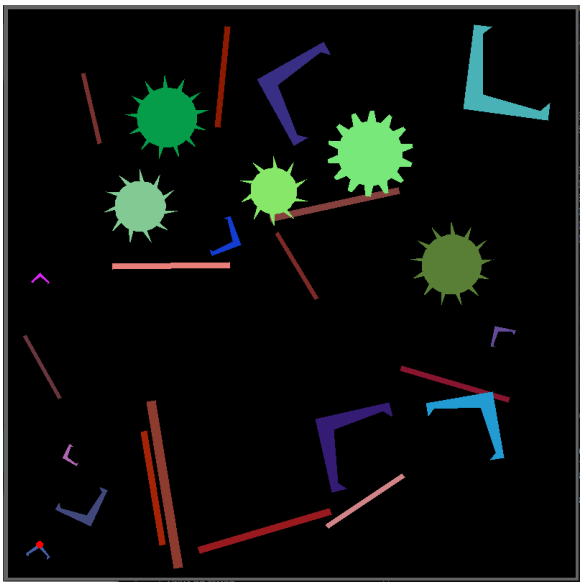


Figura 16: Esse é o resultado da simulação com função de seleção "Repetição frequente". É possível verificar que nenhuma estrutura coerente emergiu apesar da quantidade extensa de gerações dos indivíduos.

4.7 Resultados Gerais

Segundo a restrição de relógio apresentada na Seção 2, todos as simulações de 100 indivíduos foram capazes de gerar relógios funcionais. Cada um otimiza sua própria pontuação dependendo da função de seleção. Em alguns momentos foi necessário muitas gerações, em outros, estruturas consideradas relógios já surgiram com 5 gerações. Ao diminuir o número de indivíduos, mesmo com muitas gerações, não foi possível encontrar um relógio. O surgimento das primeiras estruturas complexas parecem dificultar o surgimento das próximas, uma vez que estas não estão sujeitos a algumas funções de seleção.

5 Limitações e trabalhos futuros

Esse trabalho possui diversas limitações em sua metodologia e resultados, grande parte delas devido ao tempo. Elas restringem as conclusões derivadas desse trabalho. São elas:

- **Estruturas Conexas**. Apesar de esse trabalho utilizar o fundo para realizar conexões, sendo possível interpretá-lo como uma peça, não há restrições nas funções de seleção testadas que motivem a evolução a unir as peças com os pinos. No mundo real, há relações de afinidade eletromagnéticas que aproximam átomos, moléculas e proteínas. Essa característica pode ser fundamental para o surgimento de complexidade e não foi abordada nesse trabalho.

- **Simulação**. A simulação nesse trabalho não é perfeita. Em diversos momentos ocorrem comportamentos inesperados nas simulações que a diferem do mundo físico. A principal fonte de problemas é a discretização do tempo, necessária para realizar qualquer simulação. Com a discretização, movimentos contínuos se tornam discretos, e sob altas velocidades, objetos que deveriam se colidir acabam se atravessando, objetos podem ficar presos nas estruturas de outros, ou ainda as colisões e forças deixam de conservar energia. Utilizar um motor de física mais apropriado poderia fornecer resultados mais próximos da realidade.

- **Custo computacional**. As simulações físicas e algoritmos evolutivos são computacionalmente custosos. Os programas utilizados apenas fazem uso da CPU. O tempo disponível para a simulação desses relógios é limitado. Unindo esses três fatores, pode ser possível que um relógio ideal fosse encontrado após gerações das simulações, mas devido às restrições, a simulação se encerrou muito antes disso ocorrer. Adaptar o motor físico e os algoritmos utilizados para a GPU poderia melhorar a eficiência computacional e percorrer mais gerações, garantindo mais liberdade à evolução.

- **Projetista**. Uma das premissas originais desse trabalho era a evolução de relógios sem a presença de um projetista. Apesar dos autores não guiarem cada passo da evolução desses relógios, as regras e funções de seleção foram desenhadas para expressar o entendimento de relógios por esses autores. Dessa forma, não é possível remover a necessidade mínima do projetista a partir desse trabalho. Apesar disso, as funções de seleção foram escolhidas de forma a minimizar o viés do projetista, e gradualmente premissas são adicionadas para auxiliar a evolução.

- **Peças**. A evolução dos relógios nesse trabalho, utilizou de peças pré desenhadas. Apesar de na natureza a evolução trabalhar com átomos e moléculas já definidos, elas são muito mais simples que as peças utilizadas nesse trabalho. Idealmente, os relógios deveriam partir ou de polígonos aleatórios ou moldáveis pela evolução, ou de pequenas peças mais simples (removendo a engrenagem e a âncora). Essas modificações expandiriam as conclusões finais, mas tornariam o processo evolutivo ainda mais custoso e complexo.

- **Reprodução**. Esse trabalho fez uma escolha peculiar para a reprodução: Uma reprodução sexuada com 3 indivíduos. Essa escolha permite impedir a dominação das populações pelo indivíduo com maior pontuação e simplificar o processo de reprodução, entretanto há duas diferenças fundamentais com a realidade que poderiam ser resolvidas em trabalhos futuros para expandir as conclusões: Devido ao custo energético, a maioria dos seres sexuados no mundo realizam reprodução com apenas 2 indivíduos. Além disso, a evolução de populações inclui a reprodução de diversos pares de indivíduos, não apenas os dominantes.

- **Dimensão**. Esse trabalho foi realizado em 2 planos bidimensionais. A vida e os relógios reais existem em um espaço tridimensional. Devido às diferentes restrições desses ambientes, não é possível extrapolar o comportamento de um sobre o outro, podendo ser mais fácil ou mais difícil evoluir relógios em 3 dimensões. Utilizar um motor físico adaptado a 3 dimensões resolveria essa limitação.

- **Mutação**. As fontes de mutação no mundo real são diversas e múltiplas. Nesse trabalho, o valor máximo de mutação e o número de mutações foi previamente definido. Utilizar outros valores poderia levar a evolução a encontrar atalhos ou explorar caminhos antes bloqueados por essas limitações.

- **Elitismo**. A estratégia usada para conservar as características benéficas de uma população, não estão presentes no mundo real, uma vez que não existem seres vivos imortais. O elitismo atrapalha no surgimento de derivas genéticas e no surgimento de variabilidade em uma população, o que limita o espaço da evolução. Adicionar um tempo máximo de vida para cada indivíduo como um parâmetro da evolução poderia ser uma solução para aproximar os resultados da realidade.

- **Motor**. Diferente de relógios reais, esse trabalho não possui nenhuma fonte externa de energia. Adicioná-la pode trazer resultados mais comparáveis ao mundo real.

Esperamos que essas limitações possam ser resolvidas em trabalhos futuros.

6 Conclusões

Esse trabalho definiu um subconjunto de peças iniciais, regras de evolução, e condições iniciais mínimas e funções de seleção para, através de um algoritmo evolutivo, construir relógios primitivos. Foram testadas 5 diferentes funções de seleção para guiar a evolução. Essas funções tentam definir relógios reais com o menor conjunto de premissas possíveis, de forma a dispensar a existência de um projetista na criação desses relógios. Apesar dos resultados estarem muito distantes do esperado, os relógios obtidos são capazes de medir o tempo e satisfazem muito bem todas as restrições definidas pelas funções de seleção. Esse trabalho também apresentou em sua metodologia uma forma de simular o ambiente para evolução dos relógios. O tempo de evolução dos relógios varia muito, sendo dependente das mutações aleatórias que ocorrem em cada geração, mas para populações com mais de 100 indivíduos, em todos os 5 casos apresentados, foi possível evoluir relógios mínimos em

até 300 gerações. Nesse trabalho foi utilizada uma condição inicial vazia para as peças, começando cada indivíduo com nenhuma peça, e adicionando a cada cruzamento, alguma alteração. Entretanto, os indivíduos partiram de 4 tipos de peças pré definidas e um limite no espaço da simulação, além de restrições também nos parâmetros de suas peças. A capacidade de medir o tempo de cada relógio é relativa a forma de definição de medir o tempo. Isso pode ser interpretado como a pontuação das funções de seleção em cada caso. Dessa forma, a seção 4 aponta individualmente a capacidade dos relógios de medir o tempo para cada geração e função de seleção. Mesmo não atingindo um relógio útil para seres humanos, relógios primitivos foram encontrados, mostrando o quão difícil é controlar o algoritmo evolutivo, sendo necessário uma função de seleção muito bem definida de forma a atingir os relógios mecânicos desejados.

Referências

[1] Blender Online Community: Blender - a 3D modelling and rendering package. Blender Foundation, Blender Institute, Amsterdam (2024), https://www.blender.org

[2] Blomqvist, V.: Pymunk: A easy-to-use pythonic rigid body 2d physics library. http://www.pymunk.org/ (2007–2025), versão **7.2.0**

[3] Calvão, A., Penna, T.: The double pendulum: a numerical study. European Journal of Physics **36**(4), 045018 (2015)

[4] cdk007: Evolution is a blind watchmaker. https://www.youtube.com/watch?v=mcAq9bmCeR0 (July 2007), vídeo do YouTube. Acesso em: 3 dez. 2025

[5] Ciechanowski, B.: Mechanical watch – bartosz ciechanowski (May 2022), https://ciechanow.ski/mechanical-watch/

[6] Darwin, C.: On the Origin of Species by Means of Natural Selection, or the Preservation of Favoured Races in the Struggle for Life. John Murray, London (1859)

[7] Kitzmiller, T., Rehm, B., Rehm, C., Fenimore, D.F., Leib, J.A., Stough, S., Eveland, B.A., Sneath, C., Smith, J., Callahan, B., Callahan, F.B., Rehm, B., Sneath, C., Smith, J., Callahan, B., Callahan, F.B.: Kitzmiller v. dover area school dist. F. Supp. 2d **400**, 707 (2005), https://law.justia.com/cases/federal/district-courts/FSupp2/400/707/2414073/, u.S. District Court, M.D. Pa., Dec. 20, 2005

[8] de Lamarck, J.B.d.M.: Philosophie zoologique..., vol. 1. F. Savy (1873)

[9] Multicambota: A evolução É mesmo um relojoeiro cego! https://www.youtube.com/watch?v=BWn92jYCcSk (March 2012), vídeo do YouTube. Acesso em: 3 dez. 2025

[10] Pennock, R.T.: Creationism and intelligent design. Annual Review of Genomics and Human Genetics **4**(1), 143–163 (2003)

[11] do Pirulla, C.: Criacionismo [9] - complexidade irredutível. https://www.youtube.com/watch?v=Q64bfjsa3E8 (Jan 2016), vídeo do YouTube. Acesso em: 3 dez. 2025

[12] Sagan, L.: On the origin of mitosing cells (1967)

[13] Shinners, P.: Pygame. http://pygame.org/ (2011)

[14] United States Congress: Constitution of the united states of america: Amendment i. Arquivos Nacionais dos EUA (1791), https://www.archives.gov/founding-docs/bill-of-rights-transcript, aprovado em 25 de setembro de 1789; Ratificado em 15 de dezembro de 1791