Introdução à Computação Bioinspirada

(Baseado no curso “Computação e Informação”)

Claus Aranha, Universidade de Tsukuba
caranha@cs.tsukuba.ac.jp
Twitter: caranha

Conteúdo

- Parte 1 — Computação Bioinspirada
- Parte 2 — Algoritmos Genéticos
- Parte 3 — Programação Genética e Vida Artificial

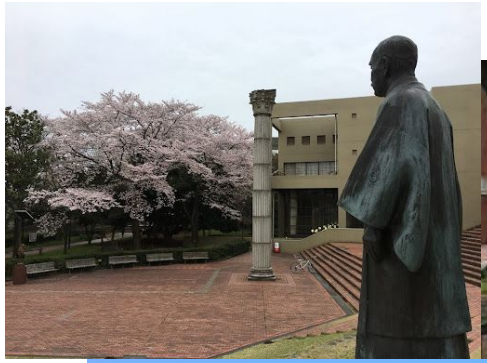
Quem sou eu?...

- **Nome:** Claus Aranha
- **Origem:** Maranhão
- **Aulas:**
  - *“Maratona de Programação”*
  - *“Matemática para a Computação”*
  - *“Experimentação em Cie. Comp.”*
- **Pesquisa:**
  - Teoria:Computação Evolutiva, Vida Artificial
  - Prática: Otimização, Jogos, Simulações
- **Interesses:**
  - Programação Competitiva, Jogos de Tabuleiro, Programação de Jogos

O que é a Universidade de Tsukuba?

筑波大学
University of Tsukuba

- “Cidade da Ciência”
- ~20.000 Alunos;
- 1 Hora de Tokyo;
- Membros famosos:
  - Varios Olímpicos
  - Matz (Ruby)
  - Jigoro Kano
  - 3 Nobels
- Parceria com a USP! (Venha fazer intercâmbio aqui!)



De onde vem os algoritmos e técnicas de CC?

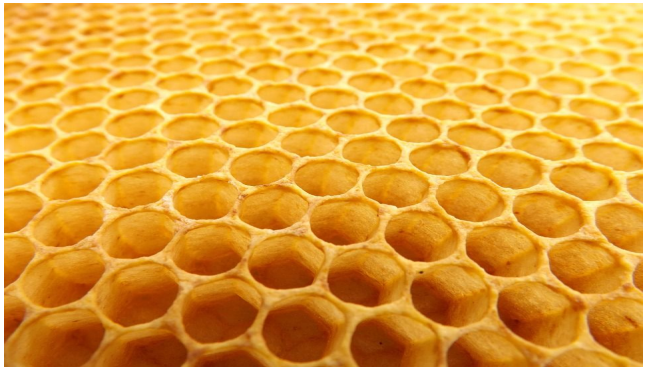
Na computação há vários algoritmos, estruturas de dados, conceitos matemáticos, entre vários outros conhecimentos necessários para formar nossos sistemas computacionais. Esses conceitos foram criados, em geral, por acúmulo de conhecimento em engenharia e matemática.

- Modelagem matemática de um problema a resolver;
- Álgebra linear para resolver o modelo matemático;
- Implementação do algoritmo através de engenharia de software, etc...

Na aula de hoje, conversaremos sobre outra maneira de se pensar sobre sistemas.

***Margareth Hamilton**, diretora de desenvolvimento de software no programa Apolo.*

Onde mais podemos buscar inspiração para o design the sistemas robustos e complexos ?



… Na Natureza!

Natureza Engenheira (1/2): Aranhas

Aprendendo um pouco dos animais incríveis da natureza.

- Teias de aranha como feitos de arquitetura;
- Vibrações na teia indicam a localização exata da presa.
- Assim, aranhas que predam outras aranhas desenvolveram um método de caça envolvendo “presas falsas”!
- As aranhas são muito mais espertas que nossos robôs atuais.

Natureza Engenheira (2/2): Formigas

Aprendendo um pouco dos animais incríveis da natureza.

- Formigas não falam, mas podem se comunicar indiretamente, e dividir tarefas complexas entre si.
- Formigas influenciam as atividades umas das outras através de feromônios.
- Stigmergia: Comunicação indireta através de trabalhos parcialmente completados.
- Usando comunicação indireta, formigas:
  - Constroem pontes,
  - Buscam comida,
  - Classificam e organizam larvas...
- E se os drones pudessem colaborar da mesma maneira que as formigas?

Buscando inspiração nos sistemas complexos da natureza, podemos criar o futuro da computação?

“Sistemas bio-inspirados” são os sistemas computacionais criados a partir de uma inspiração no mundo natural.

Vamos começar com dois exemplos de sistemas bio-inspirados:

1) Um robô cujo método de movimento é inspirado em aranhas;

2) Uma simulação de computação gráfica baseada em flocos de pássaros;

Exemplo de computação bio-inspirada 1: “Tabbot”, o robô aranha

Dr. Ingo Rechenberg queria desenvolver um robô que funcionasse em ambientes extremos, tais como desertos.

Para isso, ele começou a pesquisar criaturas que vivem no deserto.

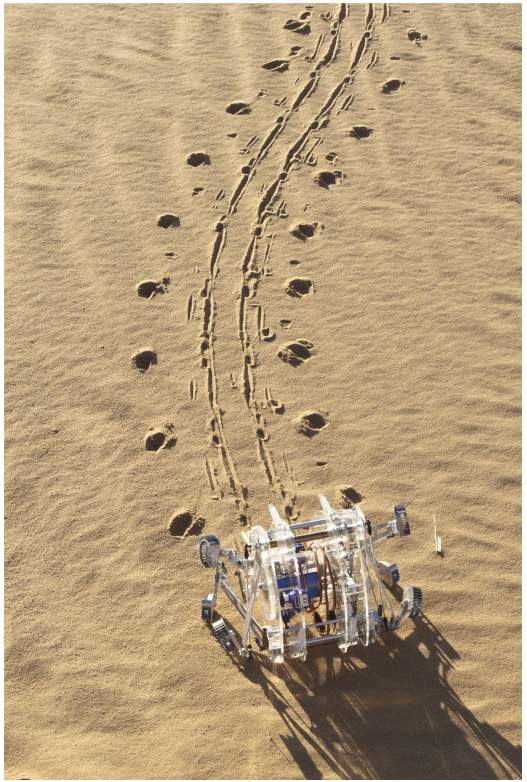
Dr. Ingo descobriu uma aranha no deserto do marrocos, capaz de dar “piruetas”.

Esta aranha, quando se sente ameaçada, faz saltos parecidos com piruetas para se deslocar rapidamente no deserto.

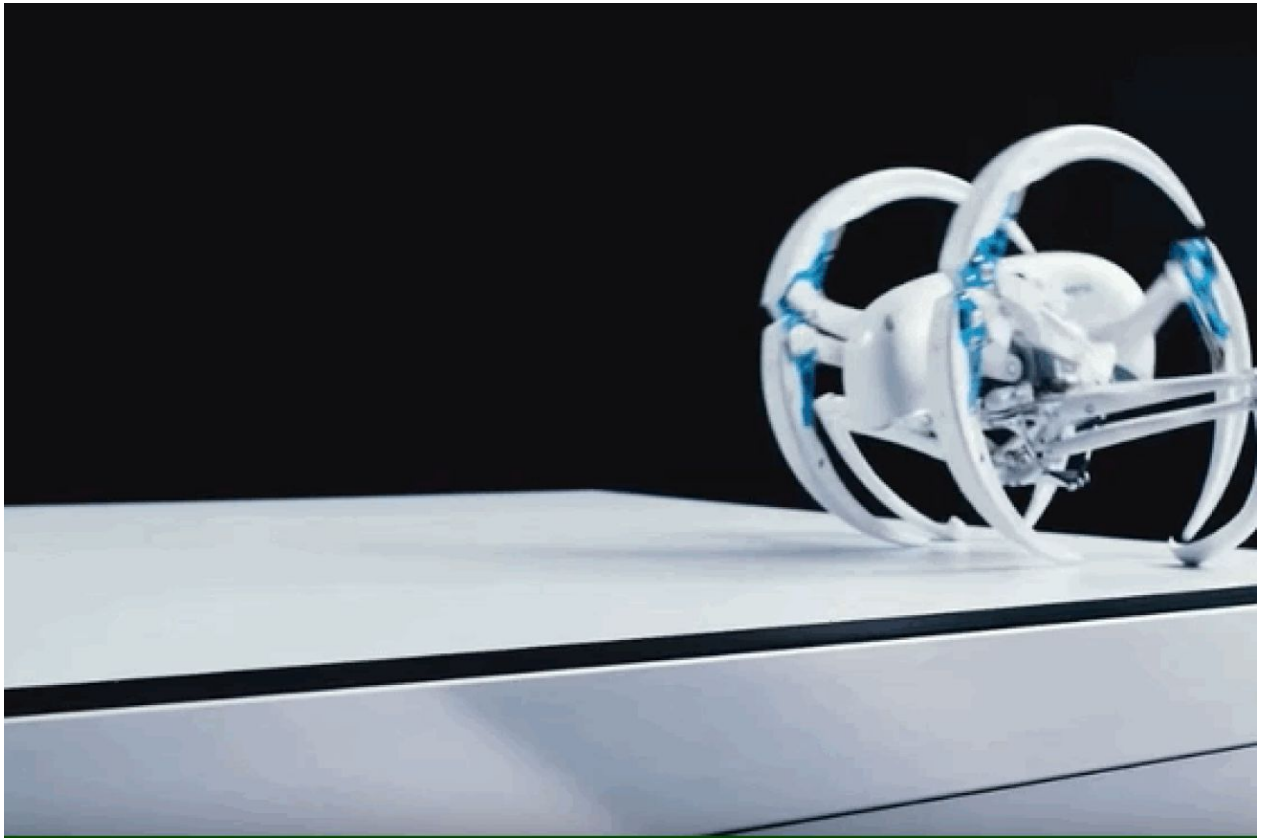
Exemplo de computação bio-inspirada 1: “Tabbot”, o robô aranha

Estudando o movimento em piruetas da aranha marroquina, Ingo desenvolveu o robô “Tabbot”.

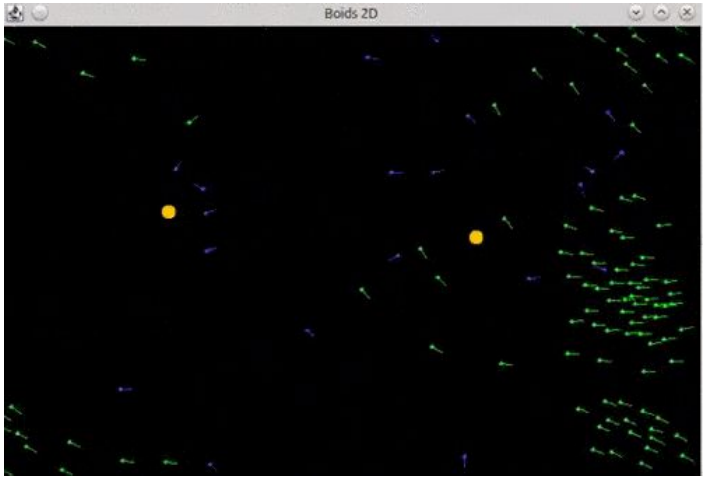
Tabbot é capaz de usar movimentos semelhantes a aranha para se mover rapidamente na areia.



Exemplo de computação bio-inspirada 1: “Tabbot”, o robô aranha



Exemplo de computação bio-inspirada 2: “Boids”, os pássaros virtuais

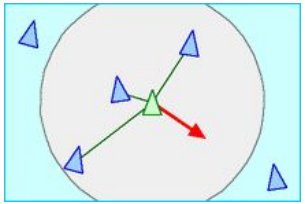


Craig Reynolds (um pesquisador em computação gráfica) criou um sistema, chamado “Boids”, que busca reproduzir os movimentos naturais de pássaros.

Em Boids, o movimento não é calculado de maneira centralizada, mas sim calculado individualmente para cada pássaro usando 3 regras simples.

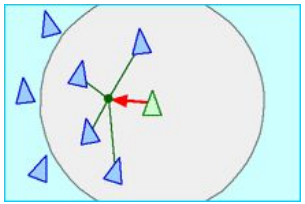
Como os Boids se movem?

- Pássaros reais nao sabem a posição e direção exata do floco como um todo, mas tem uma noção de sua distância para outros pássaros próximos.
- Cada boid usa apenas essa informação para calcular seu movimento.



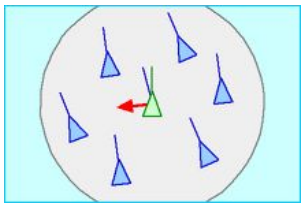
Regra 1: Se afastar de boids que estejam próximos

$F_{avoid} = \sum(Pos_{Me} - Pos_{boid\_i}) * W_{avoid}$



Regra 2: Se aproximar do centro do floco próximo

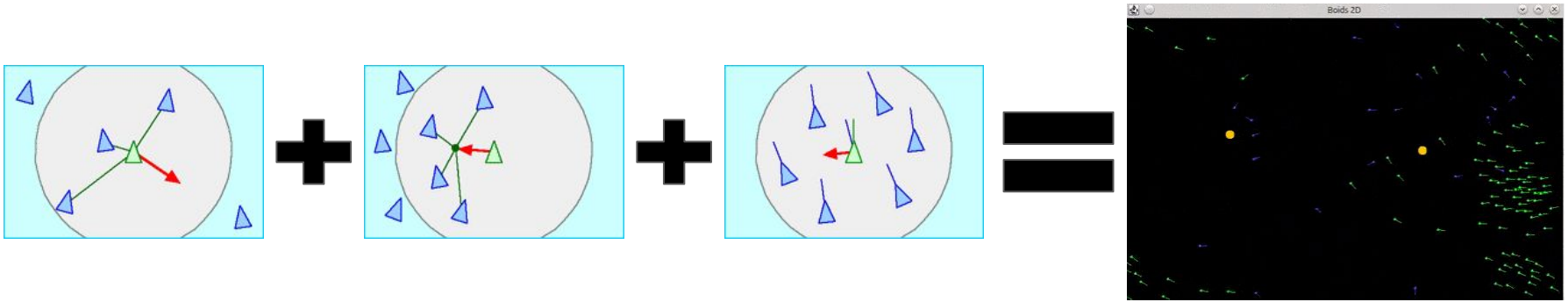
$F_{group} = Mean(Pos_{boid\_i}) * W_{group}$



Regra 3: ajustar a direção baseado nos boids próximos

$F_{align} = Mean(Dir_{boid\_i}) * W_{align}$

Como os Boids se movem?



Calculamos o vetor de velocidade do boid usando as três regras anteriores:

$$Speed_{New} = Speed_{Old} + F_{avoid} + F_{group} + F_{align}$$

O surgimento de sistemas complexos a partir da combinação de regras simples é chamado “emergência”. O conceito de emergência é central na pesquisa de sistemas bio-inspirados.

Sistema Boid usado para Computação Gráfica

Cena do filme “Lord of The Rings: The Two Towers (2002)”.
Os soldados nesta cena são animados usando um sistema similar ao Boids.

Resumo da Ópera: Parte 1

- Sistemas bioinspirados são sistemas computacionais criados a partir de uma inspiração em um sistema natural.
  - Exemplo 1: Imitando o movimento da aranha marroquina, o robô “Tabot” se move na areia;
  - Exemplo 2: Sistema de CG multi-agente “boids” que imita o movimento dos pássaros;
- Na parte 2, explicaremos os “Algoritmos Genéticos”, um sistema bioinspirado para a resolução de problemas.
- Na parte 3, mostraremos um sistema bioinspirado para programação: “Programação Genética”.

De onde vêm as “idéias” de sistemas naturais?

Charles Darwin

Na parte 1, mostramos sistemas computacionais que imitam sistemas complexos naturais, como aranhas e pássaros.

Mas, se quisermos não apenas imitar certos sistemas naturais, mas sim criar novos sistemas, como a natureza faz?

Em outras palavras, não queremos simplesmente usar os pontos fortes de cada sistema natural, mas sim usar a força que criou esses sistemas em primeiro lugar.

Essa força é a evolução, e a computação evolutiva é a resposta que estamos procurando aqui.

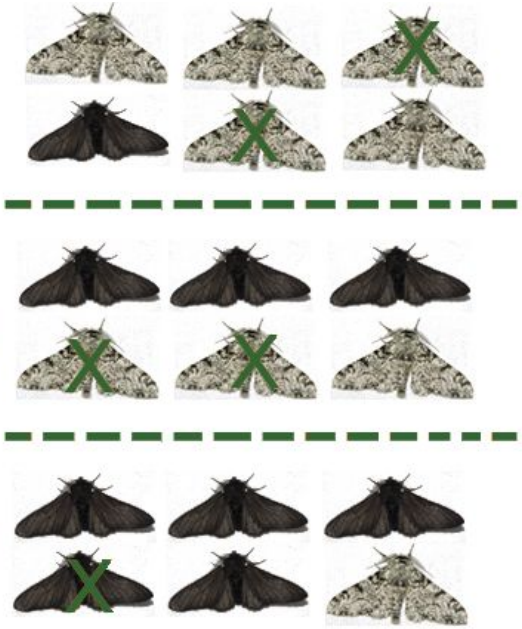
Lembrando a Teoria da Evolução: (1) Seleção Natural

- Criaturas adaptadas ao ambiente vivem por mais tempo, e tem mais descendentes. Assim, o número dessas criaturas aumenta.
- Criaturas não adaptadas ao ambiente não sobrevivem, reproduzem-se menos, e o seu número no ambiente diminui.

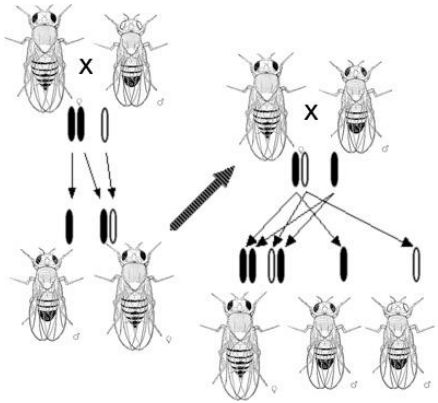
Exemplo: Na Inglaterra, o numero de mariposas brancas e negras mudou antes e depois da revolução industrial. A poluição fez as árvores esconderem melhor as mariposas negras.

Unpolluted Environment

Polluted Environment



Lembrando a Evolução: (2) Mutação e (3) Genes



De onde vem a diferença entre mariposas brancas e negras?

As mudanças de uma criatura para outra vêm pela mutação.

A mutação introduz novas características. Se essas características forem benéficas, a seleção natural faz com que a criatura floresça no ambiente.

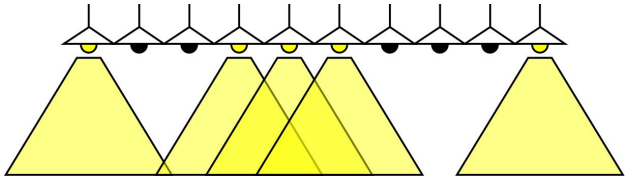
Como uma criatura passa suas características para seus descendentes? Os "genes" carregam informação de uma criatura para a próxima.

Mutações causam mudanças nos genes, e alguns animais trocam genes através da reprodução sexual.

Resumindo: Os genes guardam as características de uma criatura, e se essas características forem favoráveis, a criatura será selecionada pelo ambiente. Novas características vem pelas mutações. Isso esta parecendo um algoritmo!

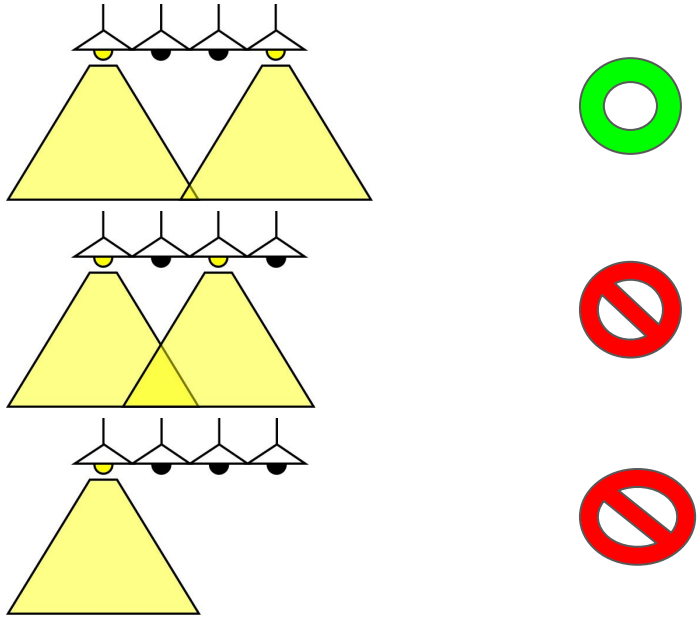
Um algoritmo usando a teoria da evolução

Um “Algoritmo Genético” usa os mecanismos da evolução para resolver problemas em geral.



Vamos resolver um problema simples usando um algoritmo genético:

- Queremos iluminar um corredor acendendo e apagando lâmpadas;
- Nenhum lugar pode ficar escuro;
- Mas, não queremos desperdiçar energia;

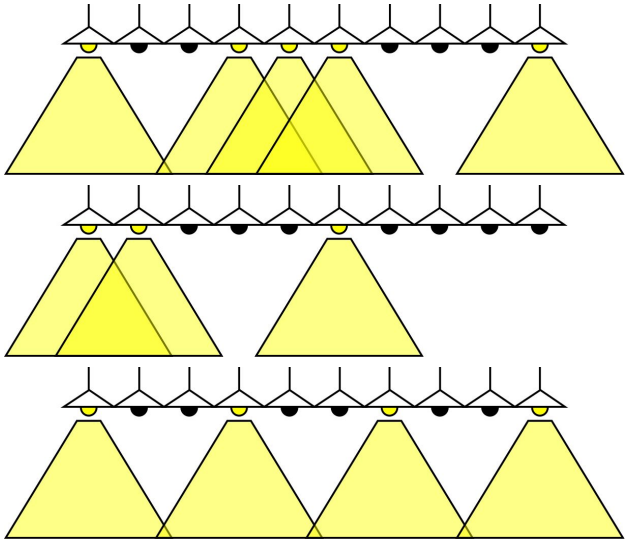


Vamos calcular quais lâmpadas acender

Algoritmo Genético 1: Estrutura Genética

O algoritmo genético começa representando uma solução para o problema como um “gene”. Esse gene é a estrutura de dados do algoritmo genético. A decisão sobre como especificar uma solução para o problema é chamada de *“encoding”* (codificação) do problema.

Para este problema, uma possível codificação é listada abaixo. Primeiro marcamos o estado do corredor como uma lista de 0s e 1s. O tamanho da lista é o número de lâmpadas. Uma lâmpada acesa é representada como 1, e uma lâmpada apagada como 0. O uso de representação binária deixa a coisa mais fácil de programar.



1	0	0	1	1	1	0	0	0	1

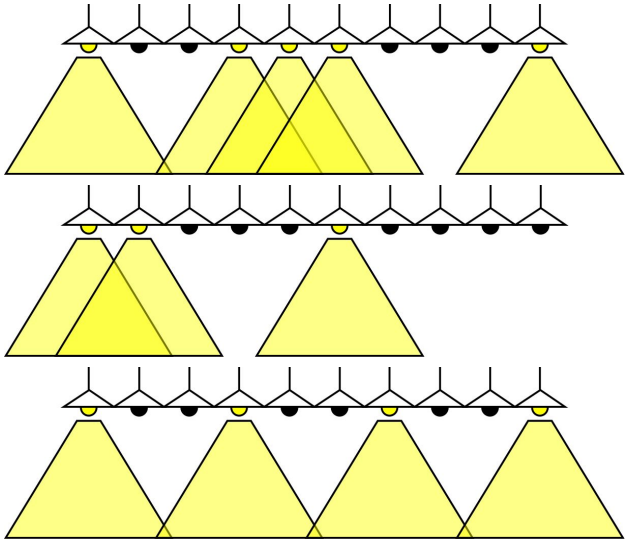
1	1	0	0	0	1	0	0	0	0

1	0	0	1	0	0	1	0	0	1

Algoritmo genético 2: Seleção

Para selecionar um gene (uma solução), precisamos avaliar sua qualidade. Chamamos a função que faz essa avaliação de “função de objetivo”, ou *“fitness”*.

Para este problema, uma fitness é “somar os espaços claros no corredor, e subtrair o número de lâmpadas acesas”. Assim, soluções melhores têm valores maiores.



1	0	0	1	1	1	0	0	0	1

Fitness: +9 - 5 = **+4**

1	1	0	0	0	1	0	0	0	0

Fitness: +6 - 3 = **+3**

1	0	0	1	0	0	1	0	0	1

Fitness: +10 - 4 = **+6**

Algoritmo genético 3: Mutação

Para criarmos novas soluções, usamos a mutação e cruzamento do gene.

Pensando na codificação que escolhemos (lista de 0s e 1s), o cruzamento pode ser a troca direta de partes da lista, e a mutação e a troca de um bit.

Parents:

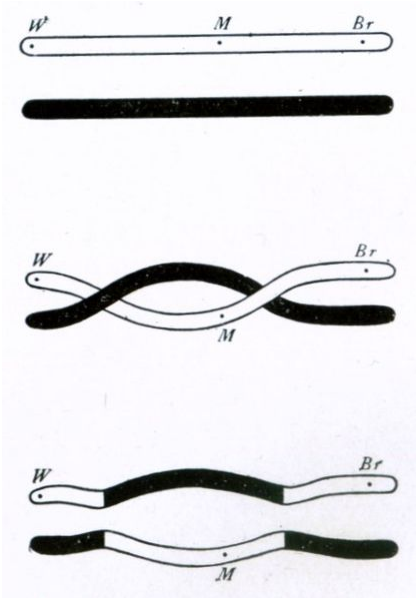
1	0	0	1	1	1	0	0	0	1
1	1	0	0	0	1	0	0	0	0

Crossover:

1	1	0	1	1	1	0	0	0	0

Mutation:

1	1	0	1	0	1	0	0	0	0



Algoritmo genético: juntando tudo

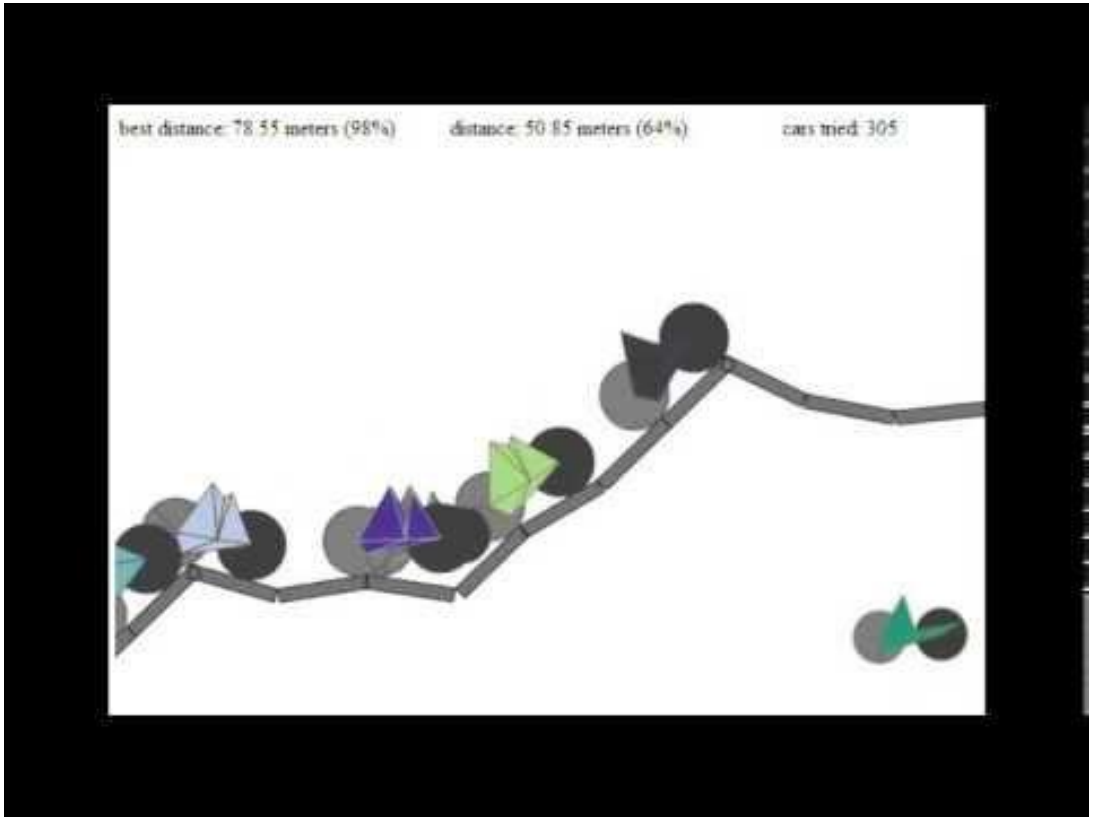
As três partes principais de um algoritmo genético são:

- Codificação da solução (gene) : Estrutura de dados do algoritmo
- Avaliação da solução (fitness): Estimar o quanto o problema foi resolvido
- Novas soluções (mutação): Função que modifica soluções existentes

Juntando essas 3 partes, o algoritmo se desenrola da seguinte maneira:

① Criamos um conjunto inicial de soluções aleatoriamente;
② Usando a função de fitness, avaliamos as soluções;
③ Eliminamos as soluções de baixa qualidade;
④ Dentre as restantes, usamos mutação para criar novas soluções;
⑤ Voltamos para o passo 2;

Exemplo de algoritmo genético: Evolução de Carros



Link: https://www.youtube.com/watch?v=uxourrlPlf8

Explicando a evolução dos carros

- **Gene :** Lista de floats com características
  - Chassis : Raio do polígono(R), ângulo(A) ;
  - Posições das rodas (W) ;
  - Raio da roda (S) ;

R_1	A_1	R_2	A_2	R_3	A_3	...	W_1	S_{W1}	W_2	S_{W2}
3.1	1.4	4.3	0.1	2.5	1.1		1	5.5	2	1.3

- **Função de Fitness :** Distância percorrida pelo carro. Maior, melhor.

- **Mutação e cruzamento:** Por usarmos floats ao invéz de binario, nao podemos usar os mesmos operadores explicados antes. Ao invéz disso, devemos usar algo como a média ou combinações lineares.

Algoritmos genéticos na prática

- Algoritmos genéticos são comumente usados em problemas de otimização;
- Em geral, eles tem bom desempenho em problemas de modelagem e design
- Um dos méritos dos algoritmos genéticos é sua capacidade de gerar soluções “criativas”, que desafiam os conceitos comuns de designers humanos.

Antena para satélites da NASA

Trem bala N700

Resumo da Ópera: Parte II

- Algoritmo Genético é um método que usa idéias da teoria da evolução para resolver problemas genéricos de maneira robusta.

- As principais partes de um algoritmo genético são:
  - ① Função de Fitness ;
  - ② Estrutura de dados genética ;
  - ③ Mutação e Crossover;

- Exemplos de algoritmos genéticos :
  - “Problema do corredor”: Que lâmpadas acender para iluminar um corredor?
  - “Evocars”: Gerar um carro atravéz da evolução para resolver uma pista.

- Algoritmos genéticos são muito usados para resolver problemas de otimização e design.

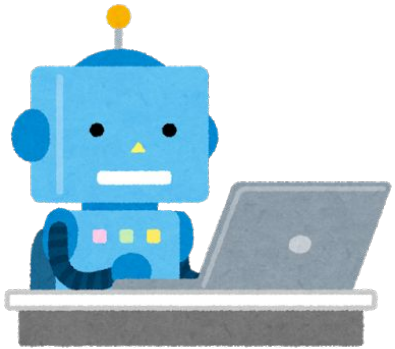
Programação Genética

- Na parte 2, apresentamos o algoritmo genético, que usa idéias da teoria da evolução para resolver “problemas arbitrários”.

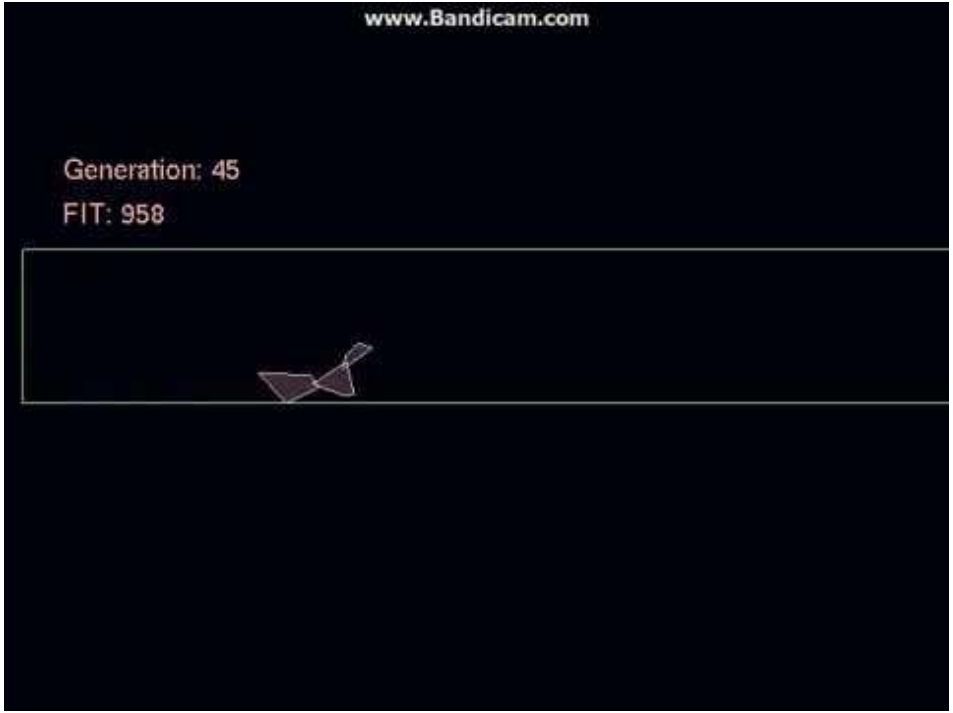
- “Problema arbitrário” aqui significa “qualquer problema”, mas até onde vai essa palavra “qualquer”? O que podemos resolver com a evolução?

- A programação genética usa da teoria da evolução para criar programas e algoritmos em si.

- O que consegue fazer um programa que seja capaz de evoluir?



Exemplo de programação genética 1: Evolução de um robo (2 minutos)



Neste video, o programa que controla os movimentos do robô é evoluído usando programação genética.

Da mesma maneira que um algoritmo genético, os programas candidatos são avaliados, reproduzidos e mutados, de acordo com sua capacidade de controlar o robô.

No exemplo deste vídeo, o programa apenas calcula o período e amplitude do movimento de cada parte do robô. Mas o resultado final até parece natural, não?

Video “Evolving to Walk” by AKUKamil: https://www.youtube.com/watch?v=GKUvZ9oD_o4

Exemplo de programação genética 2: Evolução de robôs jogadores de futebol (2 min)

Neste video, novamente vemos a evolução de um programa que controla robôs.

Porém, este exemplo é um pouco mais interessante:

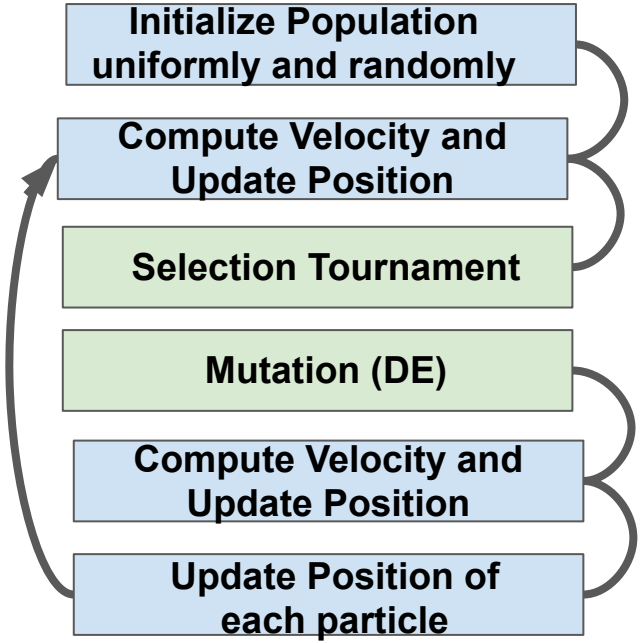
- Os robôs aqui podem observar e ouvir ao seu redor (input)
- Os robos podem chutar, girar e correr (output)
- É necessário que o time colabore entre si.

Video: “Evolution of a self-organizing robot soccer team”, by Wilfried Elmenreich: https://www.youtube.com/watch?v=cP035M_w82s

Resumo da programação genética

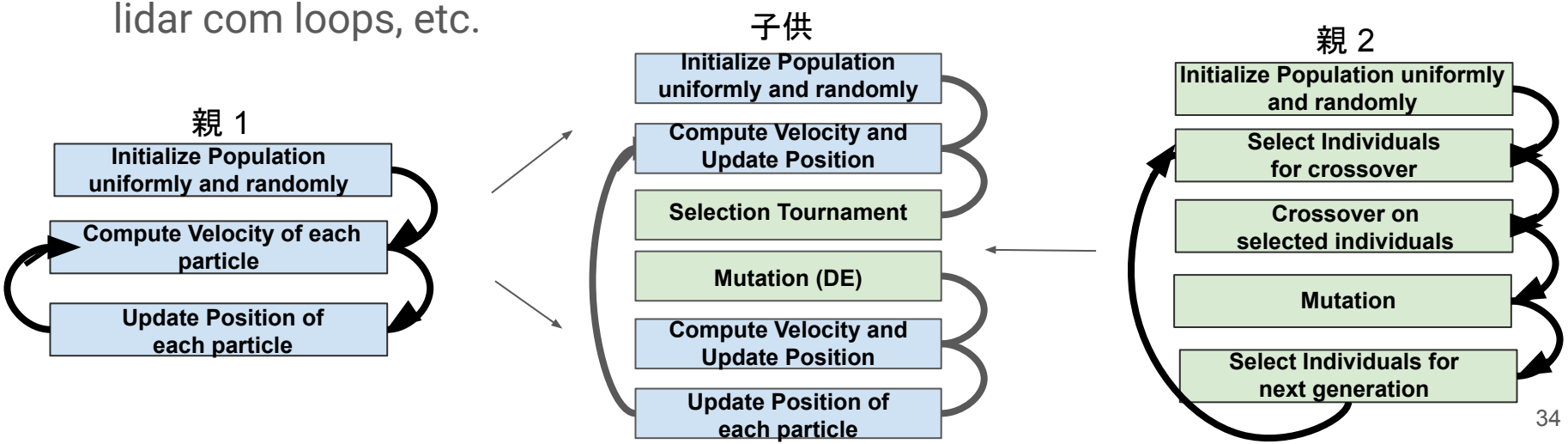
- A estrutura da programação genética é em grande parte igual ao algoritmo genético (função de fitness, loop, etc)
- Por outro lado, a codificação é mais complicada (condicionais, loops, etc).

```
while Solution.nfe < max_nfe:
    U = X
    #Round 1
    S1 = op.select_current(U)
    U = op.w_pso(S1, w=0.75, c1=0.25, c2=1.00)
    X  = op.replace_if_random(X, U)
    #Round 2
    S1 = op.select_tournament(U, n=1, k=int(n*0.50))
    S2 = op.select_random(X, 1)
    S3 = op.select_current(X)
    U  = op.w_mut_de(S1, S2, S3, beta=0.50)
    X  = op.replace_if_best(X, U)
    #Round 3
```



Codificação na programação genética

- A parte difícil de preparar na programação genética é a codificação e os operadores de mutação e crossover.
- Normalmente, este problema é resolvido dividindo o programa em módulos, e tratando esses módulos como parte do gene.
- Problemas de pesquisa: definição de “módulo”, método de crossover, como lidar com loops, etc.



Da programação genética à vida artificial

- Nós vimos exemplos da programação genética usada para criar programas para o controle de robôs que andam e jogam futebol.
- Mas até onde podemos levar a complexidade de sistemas gerados pela programação genética? Podemos criar algo próximo de uma criatura natural?
- Esse tipo de pergunta leva ao campo de pesquisa “Vida Artificial”
- Na vida artificial, tentamos fazer simulações mais e mais próximas de sistemas vivos naturais.

Exemplo de vida artificial: The bibites

Na simulação “Bibites”, vemos a evolução de:

- Diferentes velocidades e padroes de movimento;
- Frequência de reprodução;
- Carnívoros e Herbívoros;
- Sistema de imunidade, agrupamento social;

A evolução de vida artificial via programação genética pode ser bem divertida!

https://leocaussan.itch.io/the-bibites

Resumo da aula,

- **Computação bioinspirada:**
  - Sistemas computacionais criados a partir de idéias da natureza;
  - Exemplo 1: Robô que pode andar na areia baseado em movimentos de aranhas (Tabbot)
  - Exemplo 2: Flocos de passáros em sistemas de computação gráfica (Boids)

- **Algoritmo genético:**
  - Algoritmo que usa a teoria da evolução para resolver problemas arbitrários;
  - Composto de 3 partes principais:
    ① Seleção (Função de fitness), ② Gene (Estrutura de dados), ③ Mutação e Crossover
  - Especialmente útil em problemas de design, capaz de soluções criativas;

- **Programação genética:**
  - Algoritmo genético aplicado à geração de programas;
  - A codificação e o crossover requerem atenção especial (problemas abertos);
  - Evolução de criaturas vivas simuladas: Vida artificial;

Para saber mais:

- **Algoritmos Genéticos:**
  - “An Introduction to Genetic Algorithms” by Melaine Mitchell (Book)
    https://mitpress.mit.edu/books/introduction-genetic-algorithms
- **Programação Genética:**
  - “A Field Guide to Genetic Programming” by Riccardo Poli, Willian B. Langton and Nicholas F. McPhee (Book)
    http://www.gp-field-guide.org.uk/
- **Vida Artificial:**
  - 『作って動かすALife』岡瑞起、池上高志、ドミニク・チェン、青木竜太、丸山典宏
    https://www.oreilly.co.jp/books/9784873118475/