# Heurística *Particle Swarm Optimization* – PSO

**Pedro E. Melha Lemos**[1]

[1]Pontifícia Universidade Católica do Rio Grande do Sul (PUCRS)
Faculdade de Informática -– Bacharelado em Ciência da Computação
Av. Ipiranga, 6681 – Bairro Partenon – CEP 90619-900 – Porto Alegre – RS – Brazil

pedro.elpidio@acad.pucrs.br

***Resumo.*** *Este relatório apresenta exemplificadamente a heurística de Otimização por Enxame de Partículas, sendo o termo em inglês Particle Swarm Optimization, o seu algoritmo, exemplos de execuções do algoritmo implementado e aplicações práticas.*

## 1. Introdução

Considere uma partícula como um barco em um lago, sendo este a função de custo. Dados dois barcos, ambos têm o objetivo de encontrar o ponto mais fundo de um lago. A técnica para encontrá-lo é a que segue:

- O barco que tiver o segundo ponto mais fundo conhecido avança em direção ao outro até encontrar um ponto mais fundo do que o já conhecido.
- A partir disso, o outro barco avança em direção ao primeiro até encontrar um outro ponto mais fundo que o encontrado anteriormente.
- Dessa forma o ponto recém encontrado é o novo ponto mais fundo conhecido.
- Então o primeiro barco volta à avançar em direção ao outro até encontrar um ponto mais fundo que o recém encontrado.
- Caso o ponto mais fundo que o primeiro barco encontrar seja coincidentemente o mesmo que o recém encontrado, significa que ambos os barcos estão no mesmo ponto e encontraram o ponto mais fundo do lago.

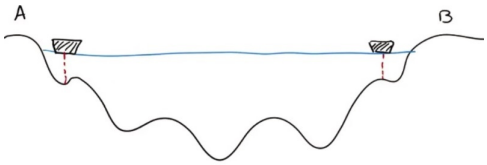


Figura 1. Estado inicial dos barcos A e B.

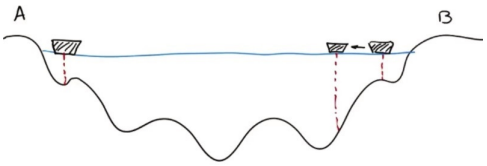


Figura 2. Barco B em direção ao barco A.

Considere a Figura 1 o estado inicial em que ambos os barcos estão parados. Cada barco sabe qual é o ponto mais fundo conhecido do lago (mínimo global) e qual é o ponto mais fundo que o próprio descobriu (mínimo local). O primeiro barco à avançar é o barco B, por que o barco A conhece um ponto mais fundo que o ponto que o barco B conhece. Dessa forma o barco B move-se em direção ao barco A para que então tentar encontrar um ponto mais fundo do que o ponto descoberto pelo barco A.

Como ilustrado na Figura 2, o barco B descobre um ponto que é mais fundo que o ponto descoberto por A, então este se torna o novo ponto conhecido mais fundo do lago.

Agora é a vez do barco A iniciar sua busca por um ponto do lago mais fundo que o ponto recém encontrado por B. Na Figura 3 o barco A move-se em direção ao barco B, já na Figura 4 o barco A encontra um ponto mais fundo que o encontrado por B.

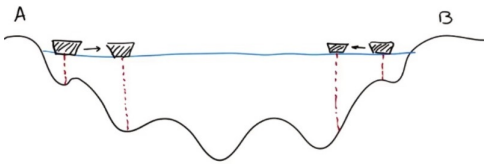


Figura 3. Barco A em direção ao barco B.

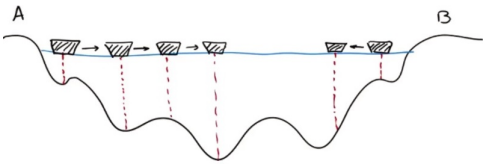


Figura 4. Barco B em direção ao barco A.

Agora é novamente a vez do barco B tentar encontrar um ponto mais fundo que o recém descoberto por A. Então este barco começa a se mover em direção ao barco A. Como o ilustrado na Figura 5, o barco B acaba parando em frente ao barco A, por que todos os pontos que ele encontrou não são mais fundos que o ponto encontrado por A e se ele continuasse avançando iria estar na mesma posição que A.

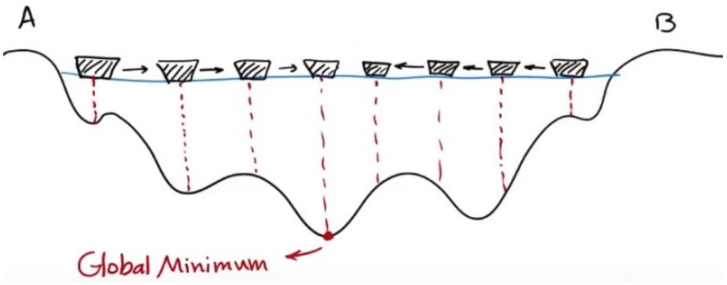


Figura 5. Ambos os barcos encontram o mínimo global.

Portanto, o barco A encontrou o ponto mais fundo de todo o lago, sendo este o mínimo global.

## 2. Propriedades

Matematicamente pode ser definido como segue:

**Posição da partícula**: $x^i_{k+1} = x^i_k + v^i_{k+1}$, onde [Rooy ]:

$x^i_k$: Posição da partícula $i$ na iteração $k$.
$v^i_k$: Velocidade da partícula $i$ na iteração $k$.

**Velocidade da partícula**: $v^i_{k+1} = w_k v^i_k + c_1 r_1 (p^i_k - x^i_k) + c_2 r_2 (p^g_k - x^i_k)$, onde [mat ]:

$p^i_k$: Melhor posição individual da partícula $i$ na iteração $k$.
$p^g_k$: Melhor posição global de todas as melhores posições individuais na iteração $k$.
$w_k$: Constante de inércia.

$c_1$, $c_2$: Peso do comportamento cognitivo (local) e social (global) da partícula respectivamente.
$r_1$, $r_2$: Números aleatórios no intervalo $[0; 1]$.

É importante destacar que na fórmula da velocidade da partícula o termo $c_1 r_1 (p_k^i - x_k^i)$ é cognitivo (local) e o termo $c_2 r_2 (p_k^g - x_k^i)$ é social (global). A Figura 6 ilustra bem as propriedades apresentadas. Na mesma figura, $p_k^i$ é o mínimo local que a partícula $i$ já encontrou e $p_k^g$ é o mínimo global de encontrado por alguma de todas as partículas. $v_k^i$ é o vetor em que a partícula $i$ está atualmente indo.

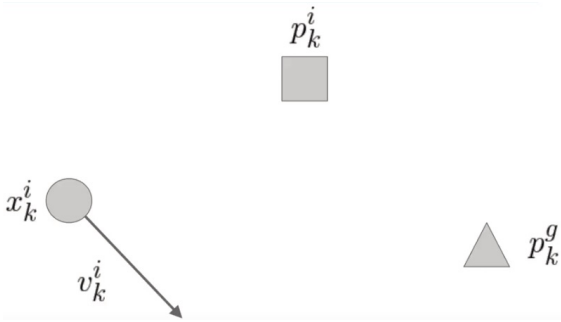


Figura 6. Partícula $i$ na iteração $k$.

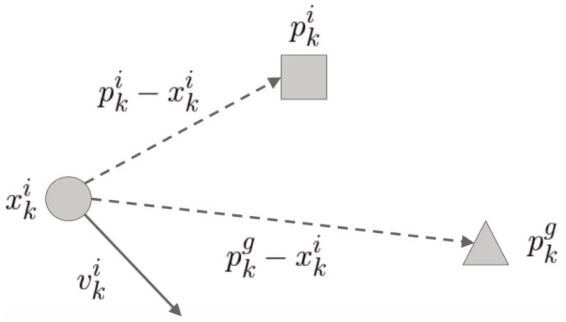


Figura 7. Vetores da partícula $i$.

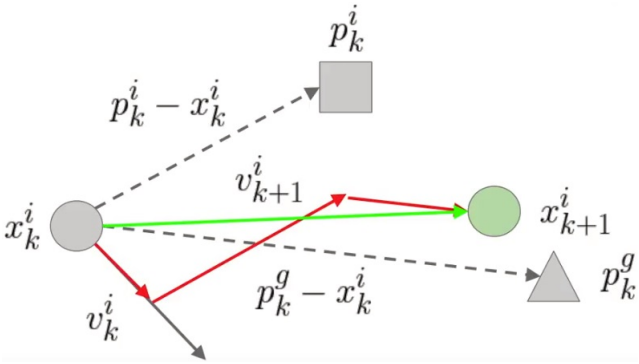


Figura 8. Combinação dos vetores da partícula $i$.

Para a próxima iteração, isto é, em $k + 1$ é feita a combinação dos dois vetores apresentados na Figura 7; dando origem à um novo vetor que aproxima a partícula $i$ do ponto mínimo global, como apresentado na Figura 8.

## 3. Algoritmo

### 3.1. Inicialização

- Definir o valor das constantes $k_{max}$, $w_k$, $c_1$ e $c_2$.
- Inicializar aleatoriamente as posições das partículas.
- Inicializar aleatoriamente as velocidades das partículas.
- Definir o contador de iterações $k = 1$.

### 3.2. Otimização

- Avaliar a função de custo $f(x_k^i)$ para cada partícula $x_k^i$.

- Se $f(x_k^i) \leq f(x_{k-1}^i)$, então $p_k^i = x_k^i$; senão $p_k^i = x_{k-1}^i$
- Se $f(x_k^i) \leq f(x_{k-1}^g)$, então $p_k^g = x_k^i$; senão $p_k^g = x_{k-1}^g$.
- Se a condição de parada for satisfeita, pare o algoritmo.
- Atualize a velocidade de todas as partículas.
- Atualize a posição de todas as partículas.
- Incremente o contador k.
- Volte para o passo inicial.

## 4. Funções de custo

Como analogia ao apresentado na introdução, o terreno do lago seria a função de custo. Há outras funções de custo que podem ser utilizadas, especialmente matemáticas, como as apresentadas abaixo:

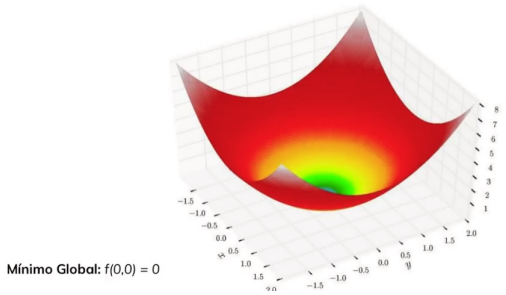


Figura 9. Função esfera.

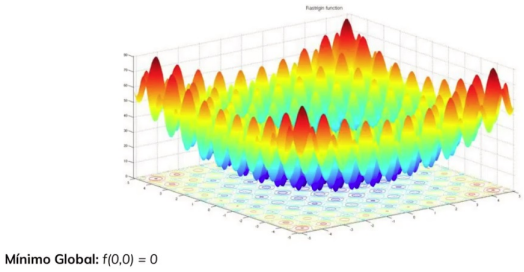


Figura 10. Função *Rastrigin*.

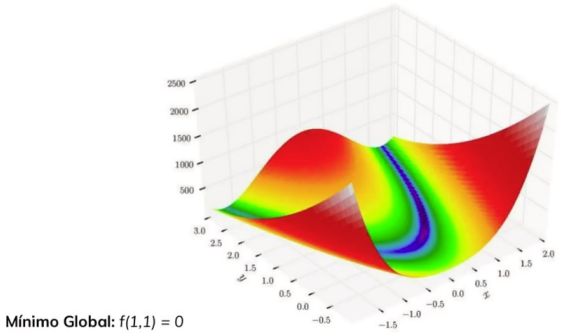


Figura 11. Função *Rosenbrock*.

## 5. Conclusão

De acordo com [Hu ], *Particle Swarm Optimization* compartilha muitas similaridades com técnicas de computação evolucionária tais como algoritmos genéticos. No entanto, as potenciais soluções seria então chamadas de partículas, como apresentado neste relatório que flutuam sobre o espaço de problema seguindo a partículas ótimas atuais.

O mesmo autor destaca que esta heurística está associada com inteligência artificial, sendo que há dois métodos populares inspirados em enxame na área: *Ant Colony Optimization* (ACO) e *Particle Swarm Optimization* (PSO). Além de também ter relação com redes neurais artificais.

## Referências

Particle swarm optimization algorithm. MathWorks. Disponível em: https://www.mathworks.com/help/gads/particle-swarm-optimization-algorithm.html?s_tid=gn_loc_drop.

Hu, X. Pso tutorial. Disponível em: http://www.swarmintelligence.org/tutorials.php.

Rooy, N. A. Particle swarm optimization from scratch with python. Disponível em: https://nathanrooy.github.io/posts/2016-08-17/simple-particle-swarm-optimization-with-python/.

Siqueira, J. G. Otimização por enxame de partículas em python. Disponível em: https://youtu.be/33hBveE7HlI.

Todas as figuras presentes nesse relatório foram retiradas de [Siqueira ].