Otimização por Enxame de Partículas

Osniel Lopes Teixeira

Universidade Presbiteriana Mackenzie, São Paulo, SP, Brasil
osnielteixeira@icloud.com

Abstract. Esse relatório demonstra o uso de algoritmo de otimização por enxame de partícular (*Particle Swam Optimization, PSO*) em função contínua. O estudo foi realizado no âmbito da disciplina de Computação Natural, ministrada pelo professor Leandro Nunes de Castro.

Keywords: Particle Swarm Optimization · Otimização · Computação Natural.

1 Introdução

O PSO realiza a tarefa de otimização por meio de diversos agentes, chamados de partículas, cada qual constituindo uma solução candidata para o problema. Cada partícula é caracterizada como um ponto num espaço n-dimensional. Nesse trabalho, apresenta-se a hiperparametrização do algoritmo quando aplicado para otimizar a função

$$f(x) = 2^{-2((x-0.1)/0.9)^2} \left(sin\left(5\pi x\right)\right)^6 \tag{1}$$

Diversas execuções foram realizadas com diferentes hiperparametros e seus resutados são demonstrados. O algortimo se mostrou bastante robusto para encontrar o ponto de máxima global em grande parte dos casos.

2 Particle Swarm Optimization

O algoritmo de otimização por enxame de partículas (Particle Swarm Optimization, PSO) utiliza diversas partículas para realizar a exploração de um espaço vetorial n-dimensional. No contexto desse algorítmo, "partícula" é uma metáfora para um vetor. Cada partícula representa uma solução candidata para o problema. As partículas são dotadas de memória que indica qual foi a posição já ocupada na sua história que melhor resolveu o problema, ou seja, gerou a melhor solução candidata. Para se moverem no espaço (gerando novas soluções) as partículas usam o conceito de velocidade. A velocidade $\mathbf{v}$ da partícula i no tempo $t+1$ pode ser calculada segundo a equação:

$$\mathbf{v}_i(t+1) = \mathbf{v}_i(t) + \varphi_1 \left(\mathbf{p}_i - \mathbf{x}_i\left(t\right)\right) + \varphi_2 \left(\mathbf{p}_g - \mathbf{x}_i\left(t\right)\right) \tag{2}$$

onde $\mathbf{p}_\mathrm{i}$ é a melhor posição já ocupada pela partícula, $\mathbf{p}_\mathrm{g}$ é a melhor posição encontrada pelas partículas da vizinhança, φ_1 é um moderador para a intensidade

com a qual a partícula se locomove em direção à sua melhor posição histórica e φ_2 é um moderador para a intensidade com a qual a partícula se move em direcão a melhor posição histórica da vizinhança e $\mathbf{x}_i$ é a posição atual da partícula. A vizinhança tem um tamanho arbitrário, podendo considerar todas as outras partículas do enxame ou apenas algumas partículas.

A cada iteração do tempo a posição da partícula será atualizada ao somar sua posição atual com o vetor velocidade. Após a atualização a memória de cada partícula será requalificada com a nova melhor posição, se houver.

3 Metodologia

3.1 Descrição do Problema

O problema no qual aplicou-se o PSO foi a otimização da função da equação 1, cujo ponto de máxima é o valor 1.

3.2 Experimento 1: Número de partículas e k-vizinhos

Para testar como a mudança no número de partículas e de k-vizinhos afeta o desempenho do algoritmo, foram realizadas diversas execuções, onde apenas esses hyperparametros foram variados e os outros foram mantidos estáticos conforme a tabela a seguir.

Tabela 1. Hyperparametrização estática dos experimentos

Experimento 1		Experimento 2	
Hyperparametro	Valor	Hyperparametro	Valor
Número de iterações	100	Número de iterações	100
φ_1	0,1	Número de partículas	30
φ_2	0,9	Número de vizinhos	9

O número de partículas foi variado de 10 a 90, em intervalos de 10. O número de vizinhos foi variado como uma fração do número de partículas de 0,1 a 0,9, em intervalos de 0,1. Para cada combinação dos parâmetros o algoritmo foi executado 100 vezes. Os resultados exibibidos na Fig. 1 representam a média das melhores posições guardadas nas memórias das partículas na última execução.

Percebe-se como o número ideal k-vizinhos parece ser no mínimo 30% do número de partículas e que um número pequeno de partículas não conseguiu executar a otimização corretamente, foram necessárias pelo menos 20 partículas para existirem bons resultados (com 10 partículas e 80% ou 90% de k-vizinhos os resultados foram bons, mas com mais partículas houveram resultados ainda melhores).

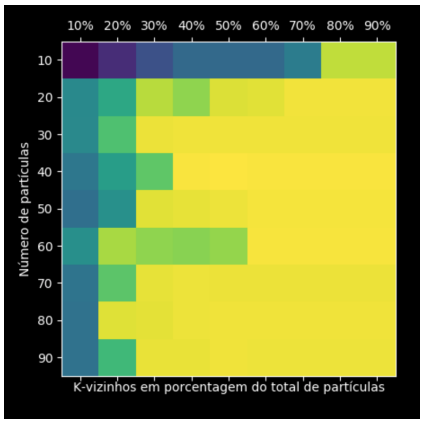

Fig. 1. Resultados do primeiro experimento. Células escuras representam valores próximos a zero, células amarelas representam valores próximos a 1.

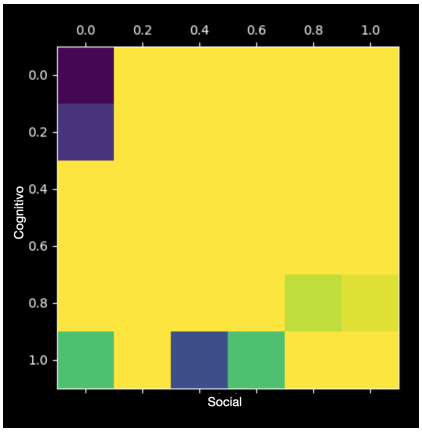

Fig. 2. Resultados do segundo experimento. Células escuras representam valores próximos a zero, células amarelas representam valores próximos a 1.

3.3 Experimento 2: Moderadores cognitivo e social

No segundo experimento testou-se como a alteração nos moderadores cognitvo (φ_1) e social (φ_2) impactariam o desempenho do algoritmo. Para isso, foram realizadas diversas execuções onde apenas esses hyperparametros foram variados e os outros foram mantidos estáticos conforme a Tabela 1. φ_1 e φ_2 foram variados de 0,0 a 1,0, em intervalos de 0,2. Para cada combinação de parâmetros o algoritmo foi executado 10 vezes. Os resultados exibidos na Fig. 2 representam a média do ponto de máxima encontrado a cada execução. Nota-se que quando os moderados são pequenos, os resultados alcançados são baixos. Também há perda de performance quando o fator cognitivo é relativamente maior que o fator social, como se pode observar nas linhas inferiores da matriz.

4 Conclusões

- O algoritmo conseguiu convergir para soluções de ótimo global em grande parte dos testes.
- Uma k-vizinhança de 30% parece o mínimo necessário para que o algoritmo realize a otimização adequadamente do problema tratado.
- Moderador cognitivo relativamente alto pode diminuir o desempenho, da mesma maneira que moderador social relativamente baixo.

Referências Bibliográficas

1. De Castro, L. N: Fundamentos de Computação Natural. https://pt.slideshare.net/lndecastro/2012-computao-natural-slides-do-curso-completo. Acesso por último em 27 de outubro de 2020.