# Otimização por nuvem de partículas aplicada ao problema de particionamento de grafos

**Eduardo Max Amaro Amaral**[1]

[1]Programa de Pós-Graduação em Informática – Disciplina Otimização Combinatória e Metaheurísticas – Professora D.Sc Maria Cristina Rangel – Universidade Federal do Espírito Santo (UFES) – Vitória, ES – Brazil

eduardomax@ifes.edu.br

***Abstract.*** *The metaheuristic Particle Swarm Optimization (PSO) has been widely used in solving nonlinear continuous problems and little explored in discrete problems. This paper presents this metaheuristic, with new adaptations, to its application in the problem of graph partitioning. At the end, results are presented computational experiments, for some graphs, in order to demonstrate the efficiency of the method in solving problems of this category.*

***Resumo.*** *A metaheurística Otimização por Enxame de Partículas (PSO) tem sido muito utilizada na resolução de problemas contínuos não-lineares e pouco explorada em problemas discretos. Este artigo apresenta essa metaheurística, com novas adaptações, para sua aplicação no problema de particionamento de grafos. Ao final, são apresentados resultados de experimentos computacionais, para alguns grafos, a fim de demonstrar a eficiência do método na resolução de problemas desta categoria.*

## 1. Introdução

Métodos de otimização e busca estocástica baseados nos princípios e modelos da evolução biológica natural têm recebido crescente interesse nas últimas décadas, devido principalmente a sua versatilidade para a resolução de problemas complexos, nas áreas de otimização e aprendizado de máquina. O desenvolvimento de modelos computacionais, inspirados nos mecanismos evolutivos, caracteriza-se pela configuração de algoritmos de otimização robustos e sistemas adaptativos.

Os Algoritmos Evolucionários (AEs) [Fogel, Owens and Walsh 1966], [Schwefel 1995], metodologias da área computação evolucionária ou evolutiva (CE), não são algoritmos computacionais em seu significado usual, mas formam uma classe de métodos regidos por princípios similares. Estes princípios oriundos do "mundo biológico" são baseados na teoria da evolução Darwiniana e tentam abstrair e imitar alguns dos mecanismos evolutivos à resolução de problemas que requerem adaptação, busca e otimização [Oliveira, Silva and Aloise 2004].

Otimização por Nuvem de Partículas (*Particle Swarm Optimization - PSO*) [Kennedy and Eberhart 1995] é uma técnica de computação evolucionária baseada em dinâmica de populações. Foi desenvolvida por James Kennedy, um psicólogo social, e por Russell Eberhart, um engenheiro elétrico, em 1995, inspirada na simulação de um

sistema social simplificado. A intenção original era simular graficamente o comportamento de um bando de pássaros em vôo com seu movimento localmente aleatório, mas globalmente determinado. Este método consiste na otimização de uma função objetivo através da troca de informações entre indivíduos (partículas) de uma população (enxame).

A Seção 2 deste trabalho apresenta o problema de particionamento de grafos. A Seção 3 detalha o algoritmo clássico do PSO assim como discute algumas abordagens para problemas discretos. A Seção 4 descreve o algoritmo PSO utilizado neste trabalho, e o ajuste dos parâmetros do algoritmo. A Seção 5 apresenta os resultados para diferentes dimensões do problema e parâmetros utilizados, e a Seção 6 apresenta as conclusões.

## 2. Problema de Particionamento de Grafos

Neste trabalho, também chamaremos esse problema de clusterização. Temos um grafo $G = (V,E)$ com pesos nas arestas que indicam a similaridade entre os pares de vértices. O objetivo é encontrar uma partição de vértices, de forma que cada parte, ou cluster, tenha arestas entre vértices internos com peso alto e o peso das arestas entre vértices de clusteres diferentes seja baixo.

Em outras palavras, podemos definir como, seja um grafo $G=(V,E)$ com $n$ vértices, o problema de particionamento de gráfico em $k >= 2$ componentes de tamanho aproximadamente igual, minimiza o corte do grafo (número de aresta que ligam as componentes ou soma dos pesos das arestas para grafo com arestas valoradas). Para um parâmetro $m >= 1$, nenhuma componente contém mais de $m * (n/k)$ vértices do gráfico.

Um fato importante sobre as funções de avaliação encontradas na literatura é que não há um consenso sobre qual a melhor função para o problema de particionamento. Entretanto, dado um particionamento $C = \{C_1, ..., C_k\}$ de $V$, dizemos que ele é um “bom" particionamento se cada subgrafo induzido por cada $C_i$ é denso, para $1 \leq i \leq k$, e, existe pouca conectividade entre os vértices desse subgrafo e os vértices no resto do grafo.

Sendo assim, devemos minimizar o corte sempre respeitando que nenhuma componente contém mais de $m * (n/k)$ para $n$, $m$ e $k$ definidos. Caso existam partições com mesmo corte, a melhor será aquela que possuir o tamanho das componentes mais equilibradas. Então, solução viável é uma partição cuja interseção das componentes seja vazia, a união seja V e nenhuma componente contenha mais de $m * (n/k)$.

## 3. Algoritmo PSO

O algoritmo de otimização por enxame de partículas (PSO) foi introduzido por James Kennedy e Russell Elberhart em 1995 para tratar problemas no domínio contínuo. O PSO emergiu de experiências com algoritmos que modelam o “comportamento social” observado em muitas espécies de pássaros e cardumes de peixes, e até mesmo do comportamento social humano.

Sendo assim, o algoritmo de otimização por enxame de partículas é um tipo de inteligência de enxame inspirado no comportamento de bandos de pássaros. A busca por

alimentos e a interação entre aves ao longo do vôo são modeladas como um mecanismo de otimização.

Fazendo uma analogia, o termo **partícula** é adotado para simbolizar os pássaros e representar as possíveis soluções do problema a ser resolvido. A área sobrevoada pelos pássaros é equivalente ao **espaço de busca** e encontrar o local com comida, ou o ninho, corresponde a encontrar a **solução ótima**. Para que o bando de pássaros sempre se aproxime do objetivo, ao invés de se perder ou nunca alcançar o alvo focado, utiliza-se o indicador denominado **fitness**, função que irá avaliar o desempenho das partículas. Para alcançar o alvo focado, sejam os alimentos ou os ninhos, os pássaros fazem uso de suas experiências e da experiência do próprio brando. O termo indicador da experiência ou conhecimento individual de cada partícula, isto é, seu histórico de vida, é o **pbest**. Em uma abordagem mais simples, o responsável por representar o conhecimento do enxame como um todo é o **gbest**. A (Tabela 1) apresenta de forma resumida as nomenclaturas descritas acima:

**Tabela 1. Identificação dos termos do PSO.**

| Termo | Significado |
|---|---|
| Partícula | Pássaro |
| Enxame | Bando de pássaros |
| Espaço de Busca | Área sobrevoada pelos pássaros |
| Posição | Localização de cada pássaro durante o vôo |
| Solução ótima | Localização do pássaro onde ele encontrou o alimento ou o ninho |
| Fitness | Função de avaliação |
| pbest | Melhor posição conhecida pelo pássaro (Experiência individual) |
| gbest | Melhor posição conhecida pelo enxame (Experiência coletiva) |

No algoritmo PSO, os indivíduos da população são representados por pontos, denominados de partículas, que voam em um espaço de busca $\Re^d$, onde d é a dimensão do espaço. As variações nos atributos desses pontos levam a novos pontos no espaço, ou seja, correspondem a movimentações no espaço. Uma ideia inspirada em sistemas cognitivos é a de que essas partículas tenderão a mover-se em direção umas às outras e irão influenciar umas às outras.

A maior parte dos algoritmos de PSO empregam dois princípios sócio-métricos, que representam dois tipos de informação importante no processo de decisão. O primeiro princípio ($g_{Best}$) conecta conceitualmente todos os membros de uma população entre si. Como consequência, o comportamento de cada partícula é influenciado pelo comportamento de todas as outras partículas. A segunda métrica ($p_{Best}$) cria uma vizinhança para cada indivíduo composta por ele próprio e seus vizinhos mais próximos. Ambas as métricas são medidas por uma função de avaliação ($f(p)$), também chamada

função objetivo ou de aptidão (*fitness*), que corresponde à optimalidade da solução do problema.

Uma partícula $p_i$ irá se mover em uma determinada direção que é função da posição atual da partícula $x_i(t)$, de uma velocidade $v_i(t+1)$, da posição da partícula que levou ao seu melhor desempenho até o momento ($p_{Best}$), e do melhor desempenho global do sistema até o momento ($g_{Best}$). A velocidade da partícula será dada por (1):

$$v_i(t+1) = v_i(t) + \varphi_1 * rand_1(.) * (p_B - x_i(t)) + \varphi_2 * rand_2(g_B - x_i(t)) \quad (1)$$

onde: $\varphi_1$ e $\varphi_2$ são constantes limitadas a um intervalo finito, em que Kennedy denomina-os como sendo respectivamente os componentes "cognitivo" (coeficiente de individualidade) e "social" (coeficiente de sociabilidade). $rand_1(.)$ e $rand_2(.)$ são duas funções aleatória no intervalo [0,1].

Uma vez que a velocidade da partícula é calculada, a posição da partícula *i* na próxima iteração é estabelecida como uma influência aditiva da posição antiga e da velocidade calculada, sendo expressa por (2):

$$x_i(t+1) = x_i(t) + v_i(t+1) \quad (2)$$

O algoritmo básico de otimização por enxame de partículas pode ser descrito brevemente utilizando os seguintes passos: dada uma população inicial de partículas, atualiza-se o vetor posição a partir do vetor velocidade de cada partícula até que se atinja o critério de parada pré-definido.

O pseudocódigo do algoritmo, em sua forma original, é descrito no Algoritmo 1.

**Algoritmo 1**: Pseudocódigo do PSO.

1. Determine o número de partículas *P* da população.
2. Inicialize aleatoriamente a posição inicial (*x*) de cada partícula *p* de *P*.
3. Atribua uma velocidade inicial (*v*) igual para todas as partículas.
4. Para cada partícula *p* em *P* faça:
   (a) Calcule sua aptidão *fp = f (p)*.
   (b) Calcule e melhor posição da partícula p até o momento ($p_{Best}$).
5. Descubra a partícula com a melhor aptidão de toda a população ($g_{Best}$).
6. Para cada partícula *p* em *P* faça:
   (a) Atualize a velocidade da partícula pela fórmula:
   $$v_i(t+1) = v_i(t) + \varphi_1 * rand_1(.) * (p_B - x_i(t)) + \varphi_2 * rand_2(g_B - x_i(t))$$
   (b) Atualize a posição da particular pela fórmula:
   $$x_i(t+1) = x_i(t) + v_i(t+1)$$

Mover Particulas

7. Se condição de término não for alcançada, retorne ao passo 4.

### 3.1. Otimização Discreta por Enxame de Partículas

O PSO foi originalmente desenvolvido para ser utilizado na solução de problemas de natureza contínua. Para tornar possível sua utilização em problemas discretos foi preciso realizar algumas adaptações nos algoritmos e na representação das partículas e suas velocidades.

Essas adaptações transformam cada componente das equações (antes simplesmente operações matemáticas) em operações discretas [Clerc 2000].

As adaptações realizadas são:

1) **Posição da partícula**: a posição da partícula passou a ser representada como um vetor (ou lista) com valores discretos. No PSO contínuo a partícula era representada por um vetor de valores contínuos. Por exemplo, no caso do Problema de Particionamento de Grafos, os valores discretos representariam as partições da solução;

2) **Velocidade da partícula**: a velocidade passou a ser vista como uma lista de transposições (swaps) realizadas no vetor posição das partículas, enquanto que no PSO contínuo era caracterizada simplesmente por um valor aritmético para cada uma das N dimensões do espaço de busca;

3) **Aplicação da velocidade a uma posição**: no PSO contínuo os valores armazenados no vetor posição das partículas eram somados algebricamente aos valores do vetor velocidade. No PSO discreto a aplicação da velocidade a uma posição é realizada a partir da troca dos valores do vetor posição de acordo com os valores da velocidade.

## 4. Algoritmo PSO Aplicado ao Problema de Particionamento

Nesta seção, são descritos os detalhes do algoritmo PSO que trata o problema de particionamento em grafos. Algumas estratégias foram desenvolvidas e implementadas na tentativa de melhorar os resultados obtidos.

Na construção das partículas iniciais, neste trabalho utilizou o proposto por [Faleiros e Xavier 2011]. Foi elaborada uma classificação gulosa aleatória com o intuito de construir soluções iniciais viáveis. Seja $R$ um subconjunto de $V$ com $k$ centros escolhidos aleatoriamente (centróides). Cada cluster será formado incrementalmente, onde a seleção do próximo elemento é guiada por uma função que mede o benefício que o mais recente elemento escolhido concede para a solução. Inicialmente, serão criados $k$ clusteres $C_1, ..., C_k$ com apenas um vértice de $R$ em cada um (centróides). A cada passo do algoritmo, escolhemos a aresta de maior peso que liga um vértice $v$ ainda não associado a um vértice de algum cluster $C_i$, $1 \leq i \leq k$, e adicionamos $v$ a $C_i$.

Nos componentes "cognitivo" ($\varphi_1$ - coeficiente de individualidade) e "social" ($\varphi_2$ - coeficiente de sociabilidade), e nas duas funções aleatórias *rand1 (.)* e *rand2 (.)*, executou-se uma estratégia similar as utilizadas em [Abraham et. al 2008] e [Fuchs et. al 2012].

Dois fatores probabilísticos foram gerados, $P_1$ e $P_2$, onde $P_1 = \varphi_1 * rand1\ (.)$, e $P_2 = \varphi_2 * rand2\ (.)$. Esses fatores definem qual componente, "cognitivo" ou "social", irá

influenciar a nova partícula. Caso $P_1 < P_2$, o fator cognitivo prevalecerá, e as novas centróides, da nova partícula, irão se basear nas centróides que originaram a partícula $p_{Best}$. Caso contrário, o fator social prevalecerá, e as novas centróides, da nova partícula, irão se basear nas centróides que originaram a partícula $g_{Best}$.

Já as atualizações da velocidade e da posição da partícula são similares ao utilizado em [Fuchs et. al 2012]. Apesar dessas atualizações serem bem definidas em aplicações com espaço de busca contínuo, a forma como as partículas são atualizadas no PSO discreto ainda carece de formalização e muitas vezes fica restrita ao tipo de problema tratado.

Neste caso, a estratégia escolhida foi utilizar a velocidade como um fator de escolha das centróides geradoras da nova partícula. As novas partículas são geradas com a mesma função de classificação gulosa, mas não mais com centróides aleatórias. As novas partículas são geradas com centróides pertencentes à própria partícula e centróides da partícula $g_{Best}$ ou $p_{Best}$, dependendo do valor de $P_1$ e $P_2$. A velocidade é iniciada utilizando-se 90% das centróides da própria partícula e 10% das centróides da partícula $g_{Best}$ ou $p_{Best}$, e atualizada a cada iteração, até a nova partícula ser gerada com 90% das centróides da partícula $g_{Best}$ ou $p_{Best}$ e 10% das centróides da própria partícula. Assim, é possível convergir a boas soluções e manter a aleatoriedade da heurística.

## 5. Resultados

Os testes foram realizados em 70 grafos, sendo 10 grafos de 4, 6, 12, 20, 50, 100 e 500 vértices cada. Todos os grafos foram particionados seguindo o padrão, onde *k* é número de partições:

4 – 2 partições;

6 – 2 e 3 partições;

12 – 2, 3 e 4 partições;

20 – 2, 4 e 5 partições;

50 – 2, 5 e 10 partições;

100 – 2, 5 e 10 partições;

500 – 2, 5 e 10 partições;

Utilizou-se 10 partículas iniciais aleatórias e o algoritmo executou por 10 iterações. Os valores utilizados nas constantes $\varphi_1$ e $\varphi_2$ foram os mesmo sugeridos por [Shi e Eberhart 1998], sendo $\varphi_1 = \varphi_2 = 2$.

A Tabela 2 traz os resultados com a média de cinco diferentes execuções do algoritmo rodando sobre os mesmos grafos.

A Tabela 3 traz um comparativo entre os algoritmos PSO, Guloso e Tabu Search para o problema de particionamento.

**Tabela 2. Identificação dos termos do PSO.**

| Heurística PSO | | |
|---|---|---|
| Tamanho | k | Custo médio |
| 4 | 2 | 12 |
| 6 | 2 | 17,6 |
| 6 | 3 | 31,2 |
| 12 | 2 | 51,6 |
| 12 | 3 | 69 |
| 12 | 4 | 97,4 |
| 20 | 2 | 112,6 |
| 20 | 4 | 213,8 |
| 20 | 5 | 252 |
| 50 | 2 | 378,2 |
| 50 | 5 | 1192,2 |
| 50 | 10 | 1830,2 |
| 100 | 2 | 1448,4 |
| 100 | 5 | 3230,2 |
| 100 | 10 | 5329,8 |
| 500 | 2 | 36205,8 |
| 500 | 5 | 71571,4 |
| 500 | 10 | 92247,4 |

**Tabela 3. Identificação dos termos do PSO.**

| | | PSO | Guloso | Tabu Search |
|---|---|---|---|---|
| Tamanho | k | Custo médio | | |
| 4 | 2 | 12 | 12,2 | 12 |
| 6 | 2 | 17,6 | 17,2 | 17,2 |
| 6 | 3 | 31,2 | 32,6 | 30,2 |
| 12 | 2 | 51,6 | 52 | 41,6 |
| 12 | 3 | 69 | 74,6 | 64,2 |
| 12 | 4 | 97,4 | 97,4 | 80,4 |
| 20 | 2 | 112,6 | 90,6 | 79,8 |
| 20 | 4 | 213,8 | 196 | 167,4 |
| 20 | 5 | 252 | 256,6 | 209 |
| 50 | 2 | 378,2 | 390,8 | 364,6 |
| 50 | 5 | 1192,2 | 905,8 | 688,6 |
| 50 | 10 | 1830,2 | 1382 | 1061,8 |
| 100 | 2 | 1448,4 | 1612,8 | 1238 |
| 100 | 5 | 3230,2 | 3137,6 | 2496,4 |
| 100 | 10 | 5329,8 | 4301,4 | 3250,6 |
| 500 | 2 | 36205,8 | 38473,8 | 38598,8 |
| 500 | 5 | 71571,4 | 73807 | 67295,4 |
| 500 | 10 | 92247,4 | 89615,4 | 79287,6 |

## 6. Considerações Finais

Este trabalho apresentou uma análise experimental do desempenho de uma abordagem baseada na otimização por enxame de partículas (PSO) aplicada a um problema de particionamento de grafos.

Os algoritmos de enxame, de modo geral, têm mostrado resultados extremamente promissores em termos de robustez de descoberta de soluções, da velocidade da solução e precisão numérica [Serapiao 2009], sobretudo na resolução de problemas contínuos.

Por outro lado, esses métodos ainda enfrentam dificuldades na aplicação de problemas de otimização de alta dimensão, principalmente por causa da carga computacional. Além disso, sua aplicação em problemas discretos demanda um grande esforço para modelar corretamente os parâmetros do algoritmo.

Os resultados, apesar de satisfatórios, mostraram que tanto a sequência quanto a forma com que as componentes da velocidade são aplicadas na atualização da posição da partícula precisam ser adequadamente caracterizadas. Isso deve ainda ser devidamente investigado.

Sendo assim, trabalhos futuros podem ser feitos na análise e melhoria das estratégias aqui adotadas, principalmente nos parâmetros e sequências de atualizações das partículas, o que provavelmente deve melhorar o desempenho geral deste algoritmo adaptado para a resolução do problema de particionamento de grafos. Propostas de códigos paralelos para o uso em multiprocessadores podem reduzir consideravelmente o tempo computacional e viabilizar o uso em grafos com muitos vértices.

## 9. Referências


Ajith Abraham, Swagatam Das, Sandip Roy. (2008) “Swarm Intelligence Algorithms for Data Clustering”, *Soft Computing for Knowledge Discovery and Data Mining*, pp 279-313, Publisher Springer US, ISBN 978-0-387-69934-9.

Clerc, M. (2000) “Discrete Particle Swarm Optimization Illustrated by the Traveling Salesman Problem”, http://www.mauriceclerc.net, February.

Fogel, L. J., Owens, A. J. and Walsh, M. J. (1966) “Artificial Intelligence Through Simulated Evolution”, New York, NY: John Wiley & Sons.

Fuchs, S. C., Delgado, M. R., Luders, R. (2012) “Pso para Otimização Combinatória: Uma Análise da Equação de Atualização da Velocidade das Partículas”, Simpósio Brasileiro de Pesquisa Operacional, Rio de janeiro, Brasil.

Kennedy, J. and Eberhart, R.C. (1995) “Particle Swarm Optimization”, In *Proceedings of the IEEE International Conference on Neural Networks*, pp. 1942–1948.

Oliveira, M. C. S. de, Silva, T. L. and Aloise, D. J. (2004) “Otimização por Nuvem de Partículas: Diferença entre Aplicações a Problemas Contínuos e Discretos”, In *XXXVI SBPO*, São João Del-Rei, MG.

Schwefel, H. P. (1995) “Evolution and Optimum Seeking” New York: John Wiley & Sons.

Serapiao, A. B. de S. (2009) “Fundamentos de otimização por inteligência de enxames: uma visão geral”, Sba Controle & Automação, vol.20, n.3, pp. 271-304, ISSN 0103-1759.

Shi, Y., Eberhart, R.C. (1998) “A Modified Particle Swarm Optimizer”, In: *IEEE World Congress on Computational Intelligence*, Anchorage, Alaska, pp.69-73.