Algoritmo de Enxame de Partículas Híbrido Aplicado a Clusterização de Dados

Gustavo H. A. Sousa[1], **Ahmed A. A. Esmin**[1]

[1] Departamento de Ciência da Computação – Universidade Federal de Lavras (UFLA)
Caixa Postal 3037 – 37.200-000 – Lavras – MG – Brazil

{sousagh,aesmin}@gmail.com

Abstract. *Clustering is an important data mining task and has been explored extensively by a number of researchers for different application areas such as text application and bio-informatics data. The Particle Swarm Optimization (PSO) is a technique based on social behavior that has been successfully applied to several types of problems. In this paper we propose the use of a novel algorithm for clustering data that we call Hybrid Particle Swarm Optimization with Mutation (HPSOM) which is based on PSO. The HPSOM basically uses PSO and by incorporating the mutation process often used in Genetic Algorithms to allow the search to escape from local optima. It is shown how the PSO/HPSOM can be used to find the centroids of a user specified number of clusters. The new algorithm is evaluated on real clustering benchmark data. The proposed method is compared with k-means clustering technique and standard Particle Swarm clustering algorithm. The results show that the algorithm is efficient and produces compact clusters.*

Resumo. *Clusterização é uma importante ferramenta de mineração de dados e tem sido extensivamente explorada por diversos pesquisadores de diferentes áreas de aplicação, tais como mineração de texto e bio-informática. O Particle Swarm Optimization (PSO) é uma técnica baseada no comportamento social que vem sendo aplicada com sucesso a diversos tipos de problemas. Neste trabalho, propomos a utilização de um novo algoritmo para agrupamento de dados que chamamos de Hybrid Particle Swarm Optimization with Mutation (HPSOM), que é baseado no PSO. O HPSOM utiliza basicamente PSO e incorpora o processo de mutação, muitas vezes usados em Algoritmos Genéticos, para escapar de ótimos locais. É mostrado como o PSO/HPSOM pode ser usado para localizar o centroides de um número de clusters especificado pelo usuário. A nova proposta de método é avaliada em dados reais de referência clustering (benchmarks). O método proposto é comparado com a técnica de agrupamento K-means, com o algoritmo Particle Swarm Optimization padrão, e com um algoritmo híbrido que usa o algoritmo K-means para preencher o enxame inicial do PSO. Os resultados mostram que o algoritmo é eficiente e produz clusters compactos.*

1. Introdução

Clustering, também conhecido como Agrupamento ou Clusterização, é uma das técnicas mais utilizadas na descoberta de conhecimento a partir de bancos de dados. Este método

consiste em encontrar grupos nos quais os seus elementos são similares entre si e diferentes dos elementos de outros grupos [Cohen and Castro 2006]. Clusterização de dados é o processo que visa reunir vetores multidimensionais em grupos, tendo em vista obter a maior similaridade dentro de cada grupo [Merwe and Engelbrecht 2003].

O *Particle Swarm Optimization* (PSO) é uma técnica de otimização estocástica proposta inicialmente por [Kennedy and Eberhart 1995] inspirado pelo comportamento social de cardumes de peixes e bandos de aves. Kennedy e Eberhart descobriram o algoritmo através da simulação de um modelo social simplificado, onde a intenção original foi simular a graciosa, porém imprevisível, coreografia de um bando de pássaros.

O *Particle Swarm Optimization* é um algoritmo relativamente novo e muitas vezes é aplicado com sucesso a diversos problemas de otimização. Recentemente surgiram na literatura algumas abordagens do PSO para resolver o problema de Clusterização de dados com resultados satisfatórios. Já que a aplicação do PSO em *Clustering* é recente, suas abordagens ainda podem ser trabalhadas a fim de encontrar melhores resultados.

Hybrid Particle Swarm Optimization with Mutation é um algoritmo proposto por [Esmin et al. 2006] que se que integra ao PSO com mutação, muitas vezes utilizada em Algoritmos Genéticos, objetivando evitar os mínimos locais e solução do problema de estagnação apresentado pelo PSO padrão.

Este artigo tem como objetivo o estudo de *Hybrid Particle Swarm Optimization with Mutation* para resolver o problema de Clusterização de dados, além de propor um novo algoritmo baseado em HPSOM no qual o algoritmo *K-means* é usado para preencher o enxame inicial.

Este trabalho está organizado da seguinte forma: a Seção 2 apresenta o algoritmo *K-means*. O algoritmo PSO é descrito na Seção 3. A Seção 4 explica como o PSO pode ser utilizado para Clusterização de dados. O algoritmo *Hybrid Particle Swarm Optimization with Mutation* é apresentado na Seção 5. Duas técnicas de Clusterização baseadas em PSO são discutidas na Seção 6. Na Seção 7 são expostos os resultados e a Seção 8 apresenta as conclusões deste artigo.

2. *K-means*

K-means (KM) é uma técnica de Clusterização de Particionamento proposta inicialmente por [MacQueen 1967] e tem o propósito de particionar uma população n-dimensional em k grupos em uma dada base de dados.

De acordo com [Carlantonio 2001], o algoritmo KM toma um parâmetro de entrada, k, e particiona um conjunto de n objetos em k *clusters* de modo que a similaridade *intracluster* resultante é alta, mas a similaridade entre *clusters* diferentes é baixa. A similaridade de *clusters* é medida em respeito ao valor médio dos objetos em um *cluster*, que pode ser visto como o centro de gravidade do *cluster*.

O algoritmo *k-means* trabalha da seguinte maneira. Inicialmente, o método, aleatoriamente, seleciona k objetos, cada um dos quais, inicialmente, representa os centroides, ou centros dos *clusters*. Para cada um dos objetos da base de dados, é feita a atribuição ao *cluster* ao qual o objeto é mais similar, baseado na distância entre o objeto e a média do *cluster*. Ele, então, computa o novo valor dos centroides, que é tido pela média dos valores dos objetos que estão atribuídos à cada *cluster* [Carlantonio 2001].

O método KM padrão pode ser resumido da seguinte maneira:

1. Escolha aleatoriamente k objetos da base de dados como os centros iniciais dos *clusters*;
2. **Repetir**;
   - (a) (re)atribua cada objeto ao *cluster* ao qual o objeto é mais similar, de acordo com o calor médio dos objetos no *cluster*;
   - (b) atualize as médias dos *clusters*, ou seja, calcule o calor médio dos objetos para cada cluster;

   até que se atinja o critério de parada;

O algoritmo *k-means* pode parar se exceder o máximo de iterações pré estabelecido, se a alteração nos centroides após uma iteração for insignificante, ou se não houver alteração nos membros de cada centroide.

A performance do algoritmo KM é sensível à partição inicial, gerada pela escolha aleatória dos centros iniciais [Zhang et al. 1999].

A necessidade de o usuário ter que especificar k, o número de *clusters* com antecedência é uma desvantagem. O método *k-means* não é adequado para descobrir *clusters* com formas não convexas ou grupos de dimensões muito diferentes. Além disso, ele é sensível a ruídos, visto que pequeno número de tais dados pode influenciar, substancialmente, o valor do resultado final do algoritmo [Carlantonio 2001].

3. *Particle Swarm Optimization*

Particle Swarm Optimization (PSO) é um método de otimização baseado em população inicialmente proposto por [Kennedy and Eberhart 1995] inspirado no comportamento social de bando de pássaros. A busca por alimento ou pelo ninho e a interação entre os pássaros ao longo do vôo são modelados como um mecanismo de otimização [Saramago and Prado 2005].

De acordo com [Merwe and Engelbrecht 2003], dado um problema, o algoritmo mantém uma enxame de partículas de tamanho s, em que cada partícula representa um candidato potencial para a solução do problema. Cada partícula encontra-se em uma posição em um espaço de busca multidimensional e mantém as seguintes informações:

- f – *fitness* da partícula. Este valor indica quão bem a partícula está posicionada no espaço de busca. Em outras palavras, o *fitness* de cada partícula significa quão boa ela é para resolver o problema. Quanto mais próximo de zero, melhor.
- x_i – Posição atual da partícula.
- v_i – Velocidade Atual da partícula.
- y_i – Melhor posição pessoal da partícula (melhor posição em que a partícula já esteve).

Assim como na natureza, para que as partículas circulem no espaço de busca, é necessário que cada partícula possua uma velocidade. Assim, a cada iteração a partícula tem sua velocidade ajustada de acordo com a seguinte equação:

$$v_{i,k}(t+1) = wv_{ik}(t) + c_1 r_{1,k}(t)(y_{i,k}(t) - x_{i,k}(t)) + c_2 r_{2,k}(t)(\hat{y}_k(t) - x_{i,k}(t)) \qquad (1)$$

Onde $v_{i,k}$ denota a $k^{-ésima}$ dimensão do vetor velocidade associado à $i^{-ésima}$ partícula. O valor w é um peso fornecido pelo usuário, que pode-se manter fixo ao longo de todo o algoritmo ou variar entre um w_{min} e um $w_{máx}$ de acordo com a seguinte equação:

$$\frac{w = w_{min} + iw_{máx} - w_{min}}{t} \tag{2}$$

Onde i é o numero da iteração e t é o número máximo de iterações do algoritmo.

A velocidade é modificada separadamente em cada dimensão $k \in i \ldots n$. r_1 e r_2 são valores aleatórios, $r_1 \sim U(0,1)$ e $r_2 \sim U(0,1)$, ou seja, entre zero e um, que contribuem para a natureza estocástica do algoritmo. c_1 e c_2, $0 < c_1, c_2 \leq 2$, são coeficientes de aceleração, c_1 regula o passo máximo na direção da melhor posição pessoal e c_2 na direção da posição global (*gbest*, ou *global best*) ou da vizinhança (*lbest*, ou *local best*) [van der Bergh 2001]. O termo $c_1 r_{1,k}(t)(y_{i,k}(t) - x_{i,k}(t))$ leva em conta as experiências passadas da partícula e é associado a cognição. O termo $c_2 r_{2,k}(t)(\hat{y}_k(t) - x_{i,k}(t))$ é o termo social, pois a partícula se inspira na melhor solução ao seu redor [van der Bergh 2001, Cohen and Castro 2006].

Na versão *gbest* do PSO, $\hat{y}$ representa a melhor posição que foi encontrada dentre todas as partículas do enxame. Esta versão fornece uma taxa de convergência mais rápida, mas que, no entanto, é menos robusta. Na versão *lbest*, a população é dividida em vizinhanças, e $\hat{y}_j$ representa a melhor posição encontrada na vizinhança da partícula j , para uma população de tamanho s e vizinhança de tamanho l [van der Bergh 2001]. A definição de $\hat{y}$, como é utilizado na versão *gbest*, é representada pela Equação (3).

$$\hat{y}_i(t) \in \{y_0(t), y_1(t), \ldots, y_s(t)\} \mid f(\hat{y}_i(t+1)) = min\{y_0(t), y_1(t), \ldots, y_s(t)\} \tag{3}$$

Onde f é a função objetivo do problema. A melhor posição global é aquela que possui menor valor da função objetivo.

A posição da partícula é atualizada usando o novo vetor velocidade de acordo com a seguinte equação:

$$x_i(t+1) = x_i(t) + v_i(t+1) \tag{4}$$

A melhor posição pessoal da partícula i é calculada como:

$$x(t+1) = \begin{cases} y_i(t) & \text{se } f(x_i(t+1)) \geq f(y_i(t)) \\ x_i(t+1) & \text{se } f(x_i(t+1)) < f(y_i(t)) \end{cases}$$

(5)

O algoritmo consiste em repetidas aplicações das equações de atualização de posição apresentadas (Figura 1). A ação de alterar a posição de cada partícula simula o movimento de um pássaro no espaço, que é influenciado pela própria memória da partícula, ou melhor posição pessoal (*personal best position*), e pela partícula que está melhor posicionada, ou melhor posição global (*global best position*). O algoritmo segue a heurística que a melhor resposta sempre se encontra próxima à melhor posição global encontrada por uma partícula, por isso, todas as outras tendem a procurar a melhor solução próxima ao *gbest*.

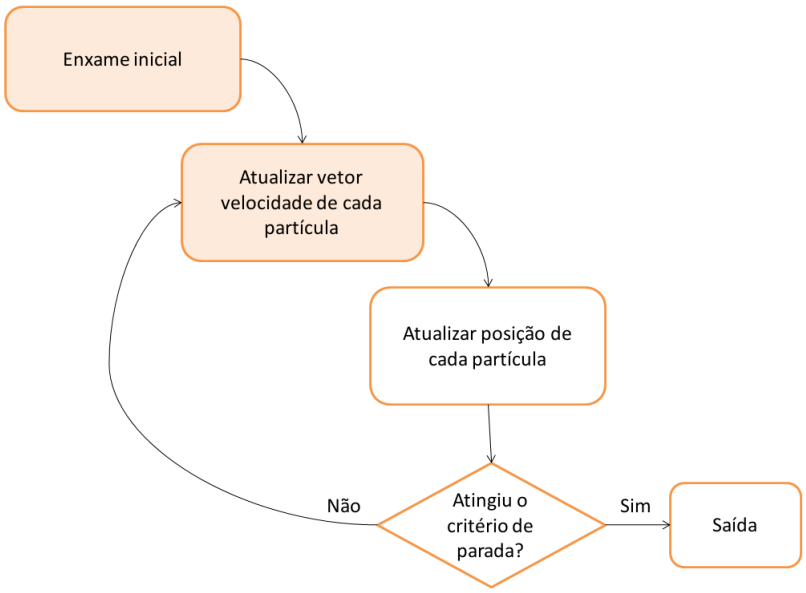

Figure 1. Fluxograma para o algoritmo PSO básico (Saramago and Prado 2005)

A inicialização da população de colônia normalmente é obtida com as partículas dispostas aleatoriamente sobre o espaço de projeto, $[-x_{max}, x_{max}]$. As velocidades $v_{i,j}$ geralmente também são inicializadas aleatoriamente em um intervalo $[-v_{max}, v_{max}]$. O critério de parada pode ser um número determinado de iterações ou outro critério dependendo do problema [van der Bergh 2001].

4. PSO Usado para Clusterização de Dados

No método proposto por [Merwe and Engelbrecht 2003], cada partícula do PSO é composta por um vetor de centroides de tamanho N_c, onde N_c é o número de grupos a serem criados, e representa uma solução completa para o problema. A cada iteração, os dados são atribuídos ao centroide ao qual estão mais próximos. As soluções são avaliadas e então atualizadas de acordo com sua melhor posição no espaço de busca e a melhor posição já encontrada por alguma partícula.

Cada partícula x_i é tido como um conjunto de *clusters*, pois, só assim, cada partícula representaria uma resposta completa da Clusterização [Merwe and Engelbrecht 2003]. Em outras palavras, no enxame do PSO, cada partícula possui um vetor de *k* centroides que são construídos da seguinte forma:

$$x_i = (m_{i1}, m_{i2}, \ldots, m_{ij}, \ldots, m_{iN_c},)$$

onde N_c é o número de *clusters* a serem criados em $m_{i,j}$ e corresponde ao $j^{-ésimo}$ centroide da $i^{-ésima}$ partícula. Assim, uma única partícula representa um candidato à solução do problema de Clusterização.

Cada partícula é avaliada através da seguinte equação denominada Função Objetivo:

$$J_e = \frac{\sum_{j=1}^{N_c}[\sum_{\forall Z_p \in c_{i,j}} d(z_p.m_{i,j})/ \mid C_{i,j} \mid]}{N_c} \tag{6}$$

Onde Z_p é o $p^{-ésimo}$ dado, $\mid C_{ij} \mid$ é o número de dados pertencentes ao *cluster* C_{ij} e d é a distância euclidiana entre Z_p e m_{ij}.

O algoritmo proposto por [Merwe and Engelbrecht 2003] pode ser descrito da seguinte forma:

1. Inicializar o valor dos centroides de cada partícula aleatoriamente
2. Para t = 1 até t_{max} faça
   - (a) Para cada partícula *i* faça
   - (b) Para cada dado z_p
     - *i*. calcule a distância Euclidiana $d(z_p, m_{ij})$ para todos os centroides m_{ij}
     - *ii*. atribua z_p ao *cluster* C_{ij}tal que $d(z_p, m_{ij}) = min_{\forall k=1...N_c}\{d(z_p, m_{ij})\}$
     - *iii*. Calcule o *fitness* da partícula utilizando a função objetivo
   - (c) Atualize o melhor global e os melhores locais
   - (d) Atualize os centroides dos *clusters*

5. *Hybrid Particle Swarm Optimization with Mutation*

O trabalho de [Esmin et al. 2006] propõe um novo motodo chamado *Hybrid Swarm Optimizer with Mutation* (HPSOM), que tem como objetivo resolver o problema de estagnação apresentado pelo PSO padrão e, desta forma impedir que as partículas fiquem presas em mínimos locais. Para resolver este problema, o HPSOM integra o processo de mutação muitas vezes utilizado em GA no PSO. A cada iteração, as partículas tem uma probabilidade, que é escolhida pelo usuário, de sofrer mutação. O processo de mutação é empregado utilizando a seguinte equação:

$$mut(p[k]) = p([k]x - 1) + \omega \tag{7}$$

Onde $p[k]$ é uma partícula escolhida aleatoriamente do enxame e ω é um valor, também aleatório, que está no intervalo $[0, 0.1x(x_{max} - x_{min})]$, que representa 10% do espaço de busca.

Simulações em uma série *benchmarks* e testes mostram que o algoritmo híbrido proposto apresenta uma melhor capacidade de encontrar o ótimo global do que algoritmo PSO padrão.

Este artigo apresenta o estudo de aplicação do método HPSOM para resolver o problema de Clusterização, o qual ainda não existe na literatura.

6. Algoritmos Híbridos

De acordo com [Merwe and Engelbrecht 2003], o método *k-means* converge para o resultado mais rapidamente que o *Particle Swarm Optimization*. Em outras palavras, o algoritmo *k-means* alcança a convergência em poucas iterações. Entretanto, na maioria dos casos, o PSO encontra grupos que possuem maior *fitness*.

Em seu trabalho, [Merwe and Engelbrecht 2003] mostram que o desempenho do algoritmo PSO para Clusterização de dados pode ser melhorado. Isto pode ser conseguido fornecendo no enxame inicial o resultado do algoritmo *k-means* a fim de obter pelo menos uma partícula próxima ao melhor particionamento. O resultado do método *k-means* é utilizado como uma das particulas, enquanto o resto do enxame é inicializado aleatoriamente e a partir daí, buscam refinar o resultado trazido pelo *k-means*. O método híbrido utilizando PSO e *k-means* foi chamado de PSOKM.

O presente trabalho faz uso da proposta de [Merwe and Engelbrecht 2003], chamando-o de PSOKM. Além disso, é proposta uma nova abordagem para a solução do problema de Clusterização, um algoritmo que usa a mesma ideia de hibridização do trabalho de [Merwe and Engelbrecht 2003] (PSOKM), mas que, no entanto, utiliza o HPSOM ao invés do PSO clássico. Esta abordagem busca a união de duas abordagens, tanto a proposta por [Esmin et al. 2006] quanto a proposta por [Merwe and Engelbrecht 2003], a fim de se obter melhores resultados que ambos separadamente. Esta combinação de métodos foi chamada neste trabalho de HPSOMKM (*Hybrid Particle Swarm Optimizatiom with Mutation and K-Means*) e é ilustrada na Figura 2.

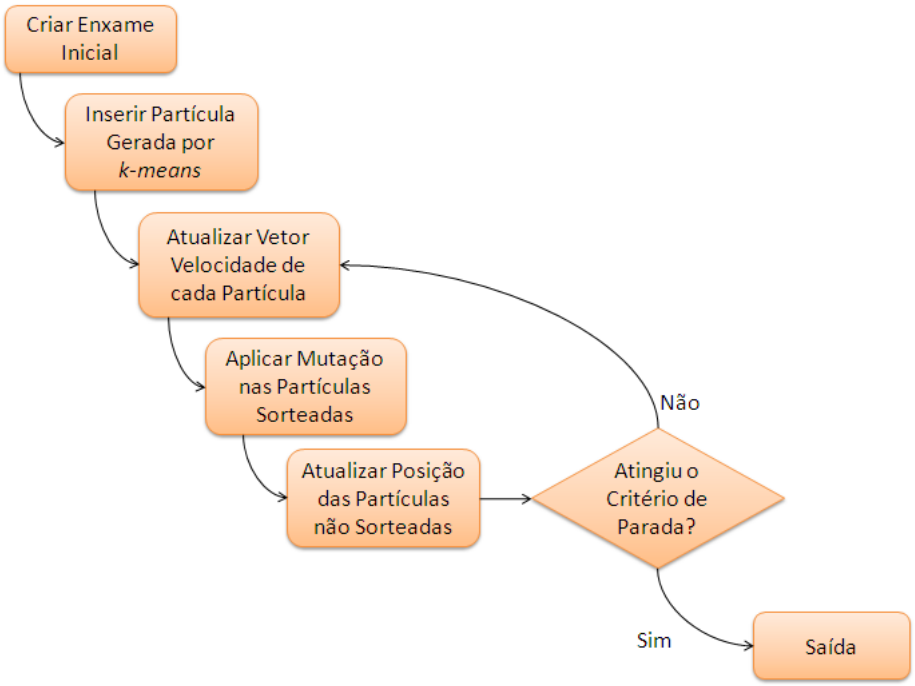

Figure 2. Fluxograma para o algoritmo HPSOMKM

Se compararmos a Figura 1 com a Figura 2 é perceptível que o algoritmo HPSOMKM utiliza a mesma ideia do PSO, com algumas diferenças básicas. No segundo passo, é introduzido uma ultima partícula que é o resultado do algoritmo *kmeans*, que

benckmark	Número de Instâncias	Número de Atributos	Número de Classes
Iris	150	4	3
Wine	178	13	3
Glass	214	9	7

Table 1. Características dos *benchmarks*

servirá como guia para as outras partículas, otimizando o algoritmo. O quarto passo, como descrito na Seção 5, executa a mutação nas partículas do enxame. Cada partícula tem uma probabilidade estabelecida arbitrariamente de sofrer mutação, que está descrita na Equação 7.

O algoritmo HPSOMKM é apresentado na Figura 2, podemos observar a introdução de uma partícula gerada a partir do resultado do *kmeans* e esta partícula tem um papel fundamental na melhoria do resultado global do enxame. A mutação também é aplicada em algumas partículas aleatoriamente selecionadas após a atualização das suas velocidades como é descrito na seção 5.

7. Resultados

Para avaliar e comparar os métodos, foram usados três *benchmarks* amplamente abordados na literatura: *Iris*, *Glass* e *Wine* [Asuncion and Newman 2007]. Todos podem ser encontrados no UCI *Repository of Machine Learning Databases* e suas características estão descritas na Tabela 1. O *benchmark Iris* apresenta cento e cinquenta instâncias da flor Iris, divididas em três classes com cinquenta instâncias cada. A primeira classe representa o tipo Setosa, a segunda ao tipo Versicolour e a terceira Virginica. As instâncias possuem quatro atributos de valores reais, *sepal length* (comprimento da sépala), *sepal width* (largura da sépala), *petal length* (comprimento da pétala), *petal width* (largura da pétala). Uma das classes (Setosa) é linearmente separável das outras duas, que não são linearmente separáveis entre si.

O *benchmark Wine* possui cento e setenta e oito instâncias de vinho. Esses dados são resultados de uma análise química realizada em vinhos da mesma região da Itália, mas vindos de diferentes cultivares. A análise determinou as quantidades de treze constituintes encontrados em cada um dos três tipos de vinho. Os atributos são: *alcohol , malic acid, ash, alcalinity of ash, magnesium, total phenols, flavanoids, noflavanoid fenols, proanthocyanins, color intensity, hue, OD280/OD315 of diluted wines, praline*. A primeira classe contém cinquenta e nove instâncias, a segunda classe contém setenta e uma e a terceira quarenta e oito.

O *benchmark Glass* apresenta duzentas e quatorze instâncias de vidros divididas em sete tipos. Setenta instâncias em *building windows float processed*, dezessete em *vehicle windows float processed*, setenta e seis em *buiding windows non-float processed*, nenhuma em *vehicle windows non-float processed*, treze em *containers*, 9 em *tableware*, vinte e nove em *headlamps*. Os atributos dos dados são: *refrative index, sodium, magnesium, aluminium, silicon, potassium, calcium, barium, iron*.

Para cada conjunto de dados, cada algoritmo foi executado por 100 vezes, com 400 iterações, 30 partículas, w variando entre 0.5 e 0.005, $c1 = 0.2$ e $c2 = 0.2$. Após uma grande quantidade de testes, foram escolhidos estes valores por obterem bons resultados

Método	*Iris*	*Wine*	*Glass*
PSO	0.08696 ± 0.00484	0.05903 ± 0.00153	0.01911 ± 0.00108
KM	0.08307	0.06155	0.01393
PSOKM	0.08210 ± 0.00203	0.05820 ± 0.00069	0.01382 ± 0.00026
HPSOM 10%	0.08203 ± 0.00289	0.00289 ± 0.00148	0.01442 ± 0.00123
HPSOM 5%	0.08295 ± 0.00320	0.05838 ± 0.00141	0.01446 ± 0.00116
HPSOMKM 10%	$0.08222 \pm$ **0.00114**	$0.05810 \pm$ **0.00074**	$0.01385 \pm$ **0.00022**
HPSOMKM 5%	**0.08164** ± 0.00250	**0.05801** ± 0.00075	**0.01377** ± 0.00037

Table 2. Resultados dos testes experimentais

em todos os métodos.

Cada execução é avaliada de acordo com o *fitness* da melhor partícula, ou seja, é calculada a média da distância entre cada elemento do conjunto de dados e o centroide ao qual ele pertence.

Para saber se determinado elemento foi agrupado de maneira correta, é preciso saber a que classe do *benchmark* o *cluster* encontrado pelo algoritmo corresponde [Esmin et al. 2008, Pereira 2007]. Dado que:

- $G = \{g_1, g_2, \ldots, g_k\}$ é o conjunto dos grupos gerados pelo algoritmo.
- $B = \{b_1, b_2, \ldots, b_k\}$ é o conjunto de classes do *benchmark*.
- r_i é a classe representada pelo grupo i.
- n_{ij} é o grupo de dados da classe *j* no grupo *i*.

O algoritmo de mapeamento de *clusters* pode ser descrito da seguinte forma:

Enquanto $G \neq$
  Encontrar maior $n_{ij} \mid g_i \in G$ e $b_j \in B$
  $r_i = j$
  $G = G - g_i$
  $B = B - b_j$

Dessa maneira, todos os dados que estiverem em c_i e também pertencerem a classe r_i estão agrupados corretamente, os outros estão agrupados incorretamente.

A Tabela 2 resume os resultados obtidos nos seis algoritmos de agrupamento para os *benchmarks* já descritos na questão da distância intracluster, ou seja, a média do valor do *fitness* da melhor partícula em todas as execuções e seu desvio padrão ($f \pm \sigma$) para cada um dos métodos descritos neste trabalho. Vale destacar que nos algoritmos que possuem mutação, foram testados dois tipos de porcentagem de mutação, 5% e 10%. Em outras palavras, os algoritmos HPSOM 5%, HPSOM 10%, HPSOMKM 5% e HPSOMKM 10% na Tabela 2 são os testes dos algorítmos HPSOM e HPSOMKM em 5% e 10% de mutação, respectivamente.

O tempo de execução médio de cada método, em 100 execuções, foi medido em milissegundos e é apresentado na Tabela 3, juntamente com o desvio padrão, para a melhor análise dos dados.

Observando a Tabela 3, pode-se perceber que o método *k-means* executa mais rapidamente que todos os outros algoritmos testados neste trabalho. Na Tabela 2, KM

Método	*Iris*	*Wine*	*Glass*
PSO	4516.5 ± 145.44	11943.43 ± 115.73	20274.5 ± 254.18
KM	**6.75 ± 7.91**	**33.18 ± 7.83**	**51.81 ± 9.29**
PSOKM	4370.06 ± 200.37	11157.18 ± 246.25	18771.5 ± 580.02
HPSOM 10%	4514.75 ± 223.71	11244.25 ± 264.12	19143.56 ± 288.55
HPSOM 5%	4526.25 ± 119.73	11217.81 ± 417.12	18730.5 ± 222.78
HPSOMKM 10%	4426.81 ± 36.77	11103.37 ± 122.08	21744.25 ± 452.76
HPSOMKM 5%	4434.62 ± 49.96	11372.06 ± 213.71	21022.56 ± 179.47

Table 3. Tempo de execução dos testes experimentais

obteve bons resultados, muito competitivos em relação ao PSO em todos os *benchmarks* testados, e sempre chega ao mesmo resultado devido a não ser um método estocástico.

O PSO obteve resultados competitivos nas duas bases *Iris* e *Wine* na comparação com o KM. No entanto, PSO apresentou resultados inferiores que o KM na base *Glass*.

Os métodos HPSOM 5% e HPSOM 10% obtiveram melhores resultados que o PSO em todos os testes e valores muito próximos entre si, com exceção dos testes feitos com *Wine*, onde o HPSOM 10% obteve resultados largamente melhores que HPSOM 5%. Também, conseguiram melhores resultados que KM no *Iris* e *Wine*. Além disso, HPSOM 5% e HPSOM 10% conseguiram menor desvio padrão que o PSO, o que significa maior confiabilidade no método.

PSOKM, como dito por [Merwe and Engelbrecht 2003], de fato, refinou o resultado do KM, ou seja, nenhum dos resultados do método *k-means* foi melhor que PSOKM. Além disso, PSOKM foi melhor que os métodos HPSOM 5% e HPSOM 10% nos testes com *Wine* e *Glass* e , também, apresentou menor desvio padrão.

Por fim, o método HPSOMKM obteve os melhores resultados deste trabalho tanto na distância intracluster quanto no desvio padrão. HPSOMKM 5% conseguiu os melhores *clusters* (Figura 3) e HPSOMKM 10% obteve o melhor desvio padrão.

8. Conclusões

Nesse trabalho, foi apresentado um estudo do problema de clusterização, uma das principais tarefas de descoberta de conhecimento em bancos de dados e aplicado em diversas áreas.

Inicialmente, foi apresentado o PSO, *Particle Swarm Optimization*, um algoritmo baseado em comportamento social que tem sido aplicado com sucesso em diversos problemas. Foram apresentados alguns métodos de resolução do problema de clusterização baseados no PSO que tem surgido na literatura nos últimos anos. Dentre eles, o método desenvolvido por [Merwe and Engelbrecht 2003]. Então, foram propostos e testados dois algoritmos híbridos, que usam a ideia de [Merwe and Engelbrecht 2003] em conjunto com *k-means* e um esquema de mutação proposto por [Esmin et al. 2006], para a clusterização de dados.

Três *benchmarks* conhecidos da área foram usados para comparar a eficiência desses métodos. O HPSOMKM demonstrou-se capaz de encontrar boas soluções para o problema, melhores que todas as outras testadas neste trabalho, principalmente em *clus-*

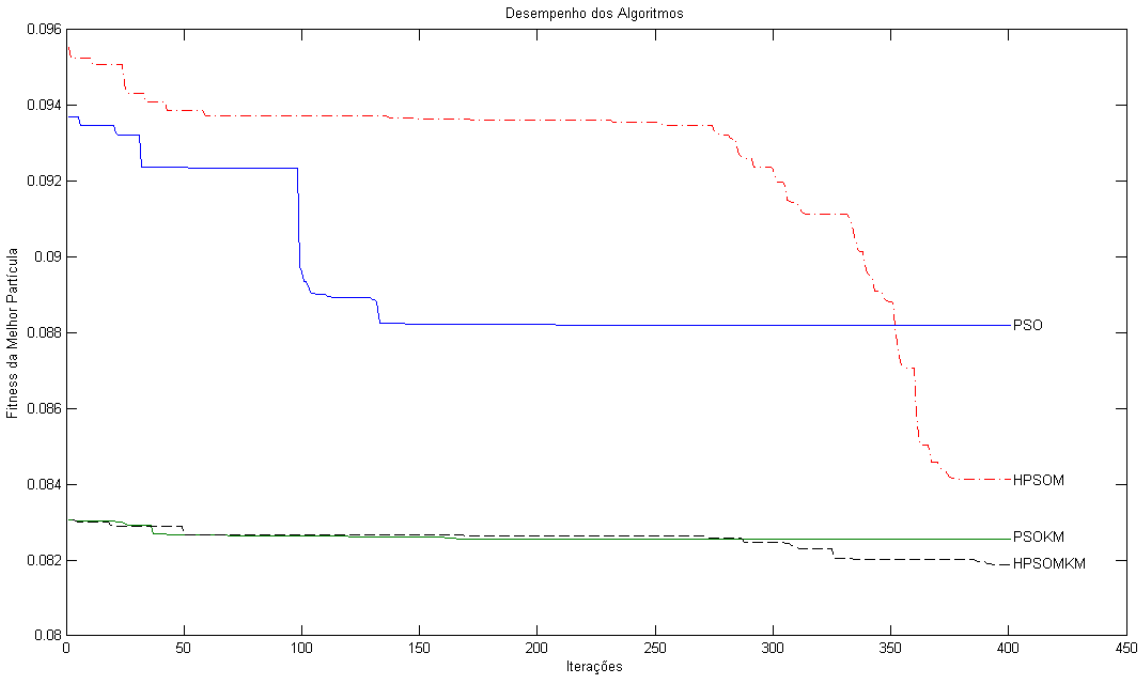

Figure 3. *Fitness* da melhor partícula ao longo das iterações (valores multiplicados por 100, para se obter melhor visibilidade)

ters com formato circular. Os resultados demonstram que é interessante utilizar técnicas híbridas para resolver problemas de Clusterização.

Em trabalhos futuros, deseja-se o desenvolvimento de um método de Clusterização, baseado no PSO, capaz de lidar com *clusters* de forma complexa e seja capaz de determinar o número ideal de clusters.

9. Agradecimentos

Os autores agradecem ao CNPq e à FAPEMIG pelo auxílio dado a esta pesquisa.

References

Asuncion, A. and Newman, D. J. (2007). Uci machine learning repository [http://www.ics.uci.edu/ mlearn/mlrepository.html]. *Irvine, CA: University of California, Department of Information and Computer Science.*

Carlantonio, L. M. (2001). Novas metodologias para clusterização de dados, coordenação de programas de pós-graduação em engenharia. Master's thesis, COPPE/UFRJ.

Cohen, S. C. M. and Castro, L. N. (2006). Data clustering with particle swarms. *Congress on Evolutionary Computation*, Proceedings of IEEE Congress on Evolutionary Computation 2006:1792–1798.

Esmin, A. A. A., Pereira, D. L., and Araújo, A. F. R. (2008). Study of different approach to clustering data by using the particle swarm optimization algorithm. In *IEEE Congress on Evolutionary Computation*, pages 1817–1822.

Esmin, A. A. A., Torres, G. L., and Alvarenga, G. B. (2006). Hybrid evolutionary algorithm based on pso and ga mutation. *6th International Conference on Hybrid Intelligent*

Systems & 4th Conference on Neuro-Computing and Evolving Intelligence. Auckland - Nova Zelândia, pages 57–60.

Kennedy, J. and Eberhart, R. C. (1995). Particle swarm optimization. *IEEE International Conference on Neural Networks, 1995, Perth. Proceedings of IEEE International Conference on Neural Networks*.

MacQueen, J. (1967). Some methods for classification and analysis of multivariate observations. In *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability.*, volume 1. University of California Press.

Merwe, D. W. and Engelbrecht, A. P. (2003). Data clustering using particle swarm optimization. *Congress on Evolutionary Computation, 2003. Proceedings of IEEE Congress on Evolutionary Computation*, pages 215–220.

Pereira, D. L. (2007). Estudo do pso na clusterização de dados. Monografia apresentada na UFLA (Universidade Federal de Lavras).

Saramago, S. F. and Prado, J. R. (2005). Otimização por colônia de partículas. *FAMAT em Revista (UFU)*, pages 87–103.

van der Bergh, F. (2001). *An Analysis of Particle Swarm Optimizers*. PhD thesis, (PhD in the Faculty of Natural and Agricultural Science) – University of Pretoria.

Zhang, B., Hsu, M., and Dayal, U. (1999). K-harmonic means -a data clustering algorithm. *Hewlett-Packard Research Laboratory*.